σύγκρισης είναι σύμφωνο με το σύμβολο, η λογική τιμή της παράστασης είναι αληθής, διαφορετικά είναι ψευδής. Τα σύμβολα σύγκρισης καθώς και η σημασία τους περιέχονται στον πίνακα 4-1.1

| Τελεστής | Σχέση |
|---|---|
| = | Ισότητα |
| <> ή >< | Ανισότητα |
| < | Μικρότερο από |
| > | Μεγαλύτερο από |
| <= ή =< | Μικρότερο ή ίσο από |
| >= ή >= | Μεγαλύτερο ή ίσο από |

***Πίνακας 8-1.*** *Τελεστές σύγκρισης*

**Παραδείγματα 8-2.**

α) Έστω η λογική παράσταση:

```
SinusMaxValue > TriangleMaxValue
```

Αν `SinusMaxValue=2` και `TriangleMaxValue=1`, η τιμή της λογικής παράσταση είναι αληθής.
Όμως, αν `SinusMaxValue=1.5` και `TriangleMaxValue=2.75`, η τιμή της λογικής παράσταση είναι ψευδής.

β) Η τιμή της λογικής παράστασης:

```
B^2 > 4*A*C
```

είναι αληθής για `B=10.5 A=3` και `C=2.93` ενώ
είναι ψευδής για `B=1.6 A=8` και `C=2`

γ) Η τιμή της λογικής παράστασης:

```
Voltage.Enabled = True
```

είναι αληθής, αν το αντικείμενο ελέγχου `Voltage` είναι ενεργοποιημένο και ψευδής, αν το αντικείμενο ελέγχου `Voltage` είναι απενεργοποιημένο.

Σύγκριση μεταξύ αλφαριθμητικών ποσοτήτων.

Η σύγκριση αλφαριθμητικών ποσοτήτων γίνεται χαρακτήρα - χαρακτήρα και ελέγχεται αν ένας χαρακτήρας προηγείται ή έπεται του άλλου στον πίνακα που χρησιμοποιείται για την παράσταση των χαρακτήρων από το υπολογιστικό σύστημα. Ας σημειωθεί, ότι σε αυτόν τον πίνακα τα ψηφία προηγούνται των λατινικών γραμμάτων και τα λατινικά γράμματα προηγούνται των ελληνικών. Επίσης, τα κεφαλαία γράμματα κάθε αλφαβήτου είναι διαφορετικά από τα μικρά και προηγούνται.

**Παραδείγματα 8-3.**

Η τιμή της λογικής παράστασης:

```
Word1 > Word2
```

είναι αληθής για:

```
Word1= "ÌÅÃÉÓÔÏÓ" και  Word2="ÌÅÃÁËÏÓ"
       "ÄÙÄÅÊÁ"             "12"          οι αριθμοί προηγούνται των γραμμάτων
       "maximum"            "MINIMUM"     τα κεφαλαία προηγούνται των μικρών
       "ÌÅÃÉÓÔÏ"            "MAXIMUM"     τα ελληνικά έπονται των λατινικών
       "ÊÅÍÏ"               "ÊÅÍÏ "       περισσότεροι χαρακτήρες στον β' όρο
       "ÊÅÍÏ"               " ÊÅÍÏ"       το διάστημα προηγείται των γραμμάτων
```

και ψευδής για:

```
Word1="Kbit"      και  Word2= "KBYTE"
      "19/12/1970"          "19/08/1970"
      "Kgr!"                "Kgr#"
      "GYM"                 "GYMNASIUM"
```



Λάθη σύνταξης στις λογικές παραστάσεις

Οι συγκρίσεις πρέπει να γίνονται μεταξύ παραστάσεων του ίδιου τύπου. Δηλαδή, ή μεταξύ αριθμητικών παραστάσεων, ή μεταξύ αλφαριθμητικών παραστάσεων. Η σύγκριση μεταξύ αριθμητικής και αλφαριθμητικής παράστασης δεν είναι αποδεκτή. Επίσης, σε μια απλή λογική παράσταση απαγορεύεται να τοποθετήσουμε δύο σύμβολα σύγκρισης.

**Παραδείγματα 8-4.**

Οι παραστάσεις:

```
New.Value + 5 = "4212 + 5"     σύγκριση αριθμητικής με αλφαριθμητική ποσότητα
Maximum >> Minimum             δεν υπάρχει τελεστής >>
LimitDown < X < LimitDown      δύο τελεστές σύγκρισης στη σειρά
```

δεν αποτελούν αποδεκτές λογικές παραστάσεις.

Σύνθετες λογικές παραστάσεις

Οι σύνθετες λογικές παραστάσεις δημιουργούνται από λογικές παραστάσεις που συνδέονται μεταξύ τους με τη βοήθεια λογικών συνδέσμων. Οι πιο κοινοί λογικοί σύνδεσμοι είναι:

## Λογικό ΚΑΙ (And) ή σύζευξη

Η σύνθετη λογική παράσταση που προκύπτει από το ***λογικό ΚΑΙ*** (σύζευξη) δύο απλών λογικών παραστάσεων είναι αληθής, μόνον αν και οι δύο λογικές παραστάσεις είναι αληθείς. Η κωδική λέξη που χρησιμοποιείται στην VB για τη σύζευξη δύο λογικών παραστάσεων είναι η `And`.

**Παράδειγμα 8-5.**

Έστω, ότι τα θερμοκρασιακά όρια για τη σωστή λειτουργία ενός ολοκληρωμένου κυκλώματος είναι -55° C, το κατώτερο, και 125° C, το ανώτερο. Η λογική παράσταση που κωδικοποιεί στην VB αυτήν την πρόταση είναι:

```
(-55 < Temperature) And (Temperature < 125)
```

Αν `Temperature=23`, δηλαδή έχει τιμή μεγαλύτερη από `-55` και ταυτόχρονα μικρότερη από `125`, η λογική παράσταση είναι αληθής (ΚΑΙ οι δύο απλές λογικές παραστάσεις αληθείς). Αν `Temperature=-70`, δηλαδή μικρότερη από `-55`, η λογική παράσταση είναι ψευδής, αφού μια από τις απλές λογικές προτάσεις (η πρώτη) είναι ψευδής.
Αν `Temperature=150`, δηλαδή μεγαλύτερη από `125`, η λογική παράσταση είναι ψευδής, αφού μια από τις απλές λογικές προτάσεις (η δεύτερη) είναι ψευδής.

## Λογικό ΄Η (Or) ή διάζευξη

Η σύνθετη λογική παράσταση που προκύπτει από το ***λογικό ΄Η*** (διάζευξη) δύο απλών λογικών παραστάσεων είναι αληθής, αν μια τουλάχιστον των λογικών παραστάσεων είναι αληθής. Η κωδική λέξη που χρησιμοποιείται στην VB για να τη διάζευξη δύο λογικών παραστάσεων είναι η `Or`.

**Παράδειγμα 8-6.**

Η λογική παράσταση:

```
(Voltage.Enabled=False) Or (Current.Enabled=False)
```

είναι αληθής αν ένα από τα αντικείμενα ελέγχου `Voltage` ή `Current` είναι απενεργοποιημένο.

## Λογικό ΟΧΙ (Not) ή άρνηση

Το ***λογικό ΟΧΙ*** δε συνδέει δύο απλές λογικές παραστάσεις αλλά τίθεται μπροστά από μια λογική παράσταση δημιουργώντας κατ' αυτόν τον τρόπο λογική παράσταση αντίθετης τιμής αλήθειας. Η κωδική λέξη που χρησιμοποιείται στην VB για να την άρνηση είναι η λέξη `Not`.

**Παράδειγμα 8-7.**

Η λογική παράσταση:

```
Not (Voltage.Locked = False)
```

ισοδυναμεί με τη λογική παράσταση:

```
Voltage.Locked = True
```

Συνδυασμός σύνθετων λογικών παραστάσεων.

Οι σύνθετες λογικές παραστάσεις μπορεί να περιέχουν και περισσότερες από δύο απλές λογικές παραστάσεις που να τις συνδέουν με περισσότερους του ενός λογικούς συνδέσμους. Ο έλεγχος αλήθειας της σύνθετης λογικής παράστασης γίνεται με βάση τις ακόλουθες αρχές:

1. Αν στη λογική παράσταση δεν υπάρχουν παρενθέσεις, ακολουθείται η εξής σειρά προτεραιότητας:
   α) Αριθμητικές πράξεις
   β) Συγκρίσεις
   γ) Λογικές πράξεις με την εξής σειρά: `Not`, `And`, `Or`.
2. Αν υπάρχουν παρενθέσεις η σειρά των πράξεων καθορίζεται από τις παρενθέσεις.

Συνηθίζεται να χρησιμοποιούμε πάντα παρενθέσεις μεταξύ των απλών λογικών παραστάσεων που συνδέονται με λογικούς συνδέσμους, ακόμα και αν δε χρειάζεται, για να είναι ευκολότερα αναγνώσιμες μιας και έτσι διακρίνονται καλύτερα και ο τρόπος δράσης των λογικών συνδέσμων είναι πιο προφανής.

**Παραδείγματα 8-8.**

Οι παραστάσεις:

```
((K<L) And (L<M)) Or (M<K)
(B<C) And ((A<B) Or (A<C))
Not ((A>B) Or (Not B>C))
```

είναι σύνθετες λογικές παραστάσεις.

Οι λογικές παραστάσεις μπορούν να γραφούν στο δεξί μέλος εντολών εκχώρησης.

Μια λογική παράσταση μπορεί να γραφεί στο δεξί μέλος μιας εντολής εκχώρησης, αρκεί στο αριστερό της να υπάρχει μεταβλητή ή ιδιότητα του τύπου Boolean. Σ' αυτήν την περίπτωση, συνηθίζεται να γράφουμε τη λογική παράσταση πάντα μεταξύ παρενθέσεων.

**Παραδείγματα 8-9.**

Στις πιο κάτω εντολές, εκχωρείται στη μεταβλητή του αριστερού μέλους η τιμή της λογικής παράστασης του δεξιού μέλους:

```
A_higherthan_B = (A>B)
Number.Visible = (IsNumeric(TextBoxA) And IsNumeric(TextBoxB))
UserPress = Check1 Or Check2 Or Check3
```

## Η δομή επιλογής If...Then ...Else

Η προγραμματιστική δομή επιλογής `If...Then...Else` χρησιμοποιείται για την εκτέλεση μιας ομάδας εντολών, αν η τιμή μιας λογικής παράστασης είναι αληθής (ικανοποιείται κάποια συνθήκη) και για την εκτέλεση μιας άλλης ομάδας εντολών, αν η τιμή μιας λογικής παράστασης είναι ψευδής (δεν ικανοποιείται η συνθήκη). Η γενική μορφή της δομής είναι:

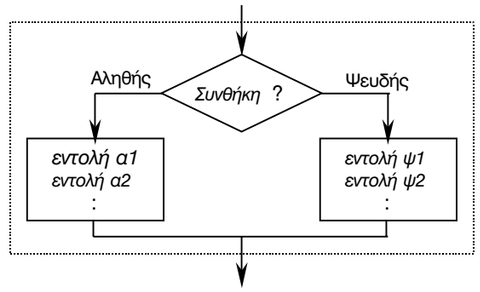
```
If συνθήκη Then
    εντολή α1   } ομάδα εντολών για
    εντολή α2   } συνθήκη αληθή
       :
Else
    εντολή ψ1   } ομάδα εντολών για
    εντολή ψ2   } συνθήκη ψευδή
       :
End If
```

όπου *συνθήκη*, μια λογική ή αριθμητική παράσταση.



Η δομή λειτουργεί ως εξής:

1. Ελέγχεται η *συνθήκη* (υπολογίζεται η αλήθεια της λογικής παράστασης).
2. Αν η *συνθήκη* είναι αληθής εκτελείται η πρώτη ομάδα εντολών (εντολές α) μεταξύ γραμμής `If` ...`Then` και γραμμής `Else`. Η ομάδα εντολών (εντολές ψ) μεταξύ της γραμμής `Else` και της γραμμής `End If` αγνοείται. Διαφορετικά, αν η *συνθήκη* είναι ψευδής, αγνοείται η πρώτη ομάδα εντολών (εντολές α) και εκτελείται η δεύτερη ομάδα εντολών (εντολές ψ).
3. Η εκτέλεση συνεχίζεται με την εντολή που ακολουθεί την `End If`




***Σχήμα 8-1.*** *Σχηματική παράσταση της δομής* `If...Then...Else`

**Παράδειγμα 8-10.**

Σε μια εφαρμογή θέλουμε, όταν το απόθεμα ενός υλικού σε μια αποθήκη γίνει μικρότερο ή ίσο των 10 τεμαχίων, να τονίζεται με κόκκινο χρώμα μια ετικέτα στο παράθυρο αναφοράς των αποθεμάτων και να αλλάζει το μήνυμά της σε "ÊÜôù áðü ôï üñéï áóöáëåßáò". Όταν πάλι το απόθεμα ξεπεράσει τα 10 τεμάχια, θέλουμε η ετικέτα να επανέρχεται σε κανονικό τρόπο εμφάνισης και να παρουσιάζει το μήνυμα "ÅðÜñêåéá".

```
If Stock <= 10 Then
   StockIndicator.Caption = "ÊÜôù áðü ôï üñéï áóöáëåßáò"
   StockIndicator.BackColor = vbRed
   StockIndicator.ForeColor = vbWhite
Else
   StockIndicator.Caption = "ÅðÜñêåéá"
   StockIndicator.BackColor = vbWhite
   StockIndicator.ForeColor = vbBlack
End If
```

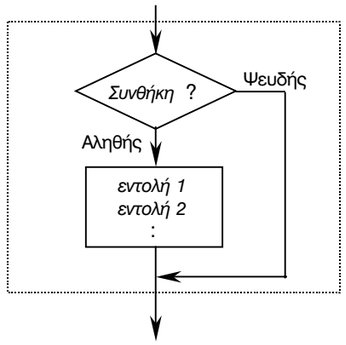
If...Then
Μια πιο απλή μορφή της δομής επιλογής

Αν η κατάσταση που εντιμετωπίζουμε είναι πιο απλή και μας αρκεί η εκτέλεση μιας ομάδας εντολών, στην περίπτωση που ικανοποιείται η συνθήκη και δε θέλουμε να γίνει κάποια λειτουργία στην περίπτωση που δεν ικανοποιείται, χρησιμοποιούμε μια πιο απλή μορφή την:

```
If συνθήκη Then
    εντολή 1    } ομάδα εντολών για
    εντολή 2    } συνθήκη αληθή
       :
End If
```

Η λειτουργία της δομής γίνεται ως εξής:

1. Ελέγχεται η *συνθήκη*.
2. Αν η *συνθήκη* είναι αληθής εκτελείται η ομάδα εντολών μεταξύ `If...Then` και `End If`. Διαφορετικά, αν η *συνθήκη* είναι ψευδής, αγνοείται η ομάδα εντολών.
3. Η εκτέλεση συνεχίζεται με την μετά την `End If` εντολή.




***Σχήμα 8-2**. Σχηματική παράσταση της δομής* `If`...`Then`

**Παράδειγμα 8-11.**

Έστω, ότι στη μεταβλητή `Max` θέλουμε να βρίσκεται ο μεγαλύτερος δύο αριθμών και στη μεταβλητή Min, ο μικρότερος τους. Μετά την εκτέλεση των εντολών:

```
If Min > Max Then
   Temporary = Max
   Max = Min
   Min = Temporaty
End If
```

θα ικανοποιείται οπωσδήποτε η αρχική απαίτηση. Πράγματι, αν στη μεταβλητή `Min` καταχωριστεί αριθμός μεγαλύτερος από αυτόν της μεταβλητής Max, θα εκτελεστούν οι εντολές στο εσωτερικό της δομής, οι οποίες θα αντιμεταθέσουν το περιεχόμενο των μεταβλητών. Αν στη μεταβλητή `Min` καταχωριστεί αριθμός μικρότερος ή ίσος από αυτόν της μεταβλητή Max δε θα εκτελεστεί καμιά εντολή από το εσωτερικό της δομής και οι μεταβλητές μετά από την `End If` θα διατηρήσουν τις επιθυμητές τιμές.

Δυνατότητες γραφής σε μια μόνο γραμμή

Αν μετά τις κωδικές λέξεις `Then` και `Else` έχουμε να γράψουμε από μια μόνο εντολή μπορούμε να συμπτύξουμε τον τρόπο γραφής της εντολής σε μια γραμμή.

**Παράδειγμα 8-12.**

Έστω, ότι στις μεταβλητές A και B έχουν καταχωριστεί δύο αριθμοί. Η εντολή:

```
If A > B Then Max = A Else Max = B
```

καταχωρίζει τον μεγαλύτερο από τους δύο αριθμούς στη μεταβλητή MAX. Πράγματι, αν `A > B` εκτελείται η εντολή μεταξύ των `Then` και `Else`, ενώ αν `A £ B`, εκτελείται η εντολή που υπάρχει μετά την κωδική λέξη `Else`.

**Παράδειγμα 8-13.**

Το τμήμα προγράμματος που ακολουθεί υπολογίζει την απόλυτη τιμή ενός αριθμού.

```
AbsoluteValue = Number
If < 0 Then AbsoluteValue = -Number
```

Στην πρώτη εντολή το πρόγραμμα "θεωρεί" ότι η απόλυτη τιμή του αριθμού `Number` είναι ο ίδιος ο αριθμός. Αυτό βέβαια, καλύπτει μόνον τις περιπτώσεις που ο αριθμός είναι θετικός ή μηδέν, γι' αυτό και στη συνέχεια ελέγχεται κατά πόσον η αρχική "υπόθεση" είναι σωστή. Αν ο αριθμός `Number` βρεθεί αρνητικός, το πρόγραμμα κάνει διόρθωση, καταχωρίζοντας στην απόλυτη τιμή `AbsoluteValue` τον αντίθετο του αριθμού.

Το αντίστροφο ισχύει κάπως διαφορετικά. Λογική παράσταση ψευδής ισοδυναμεί με αριθμητική παράσταση με τιμή 0 και λογική παράσταση αληθής ισοδυναμεί με αριθμητική παράσταση με τιμή -1.

Τελειώνοντας, υπενθυμίζουμε ότι σαν συνθήκη μπορεί να χρησιμοποιηθεί και αριθμητική παράσταση. Αριθμητική παράσταση με τιμή μηδέν ισοδυναμεί με ψευδή λογική παράσταση και αριθμητική παράσταση με τιμή διάφορη του μηδενός ισοδυναμεί με αληθή λογική παράσταση.



**Παράδειγμα 8-14.**

Οι εντολές:

```
If Temperature Then Message = "Èåñìïêñáóßá äéÜöïñç ôùí Ï°"
If Á-Â Then Message="Ïé áñéèìïß åßíáé Üíéóïé"
If CMonth Or CMonth>12 Then Message = "ÌÞíáò åêôüò ïñßùí"
```

είναι ισοδύναμες με τις:

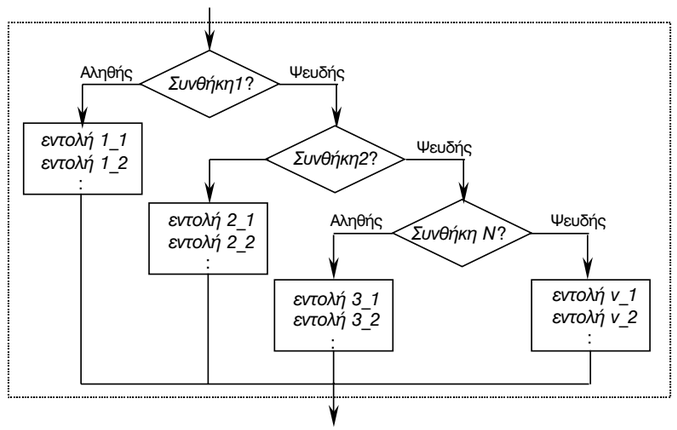
```
If Temperature <> 0 Then Message = "Èåñìïêñáóßá äéÜöïñç ôùí Ï°"
If Á <> Â Then Message = "Ïé áñéèìïß åßíáé Üíéóïé"
If (CMonth<=0) Or (CMonth>12) Then Message = "ÌÞíáò åêôüò ïñßùí"
```

## Πολλαπλές επιλογές. Η δομή `If...Then...ElseIf`

Υπάρχουν περιπτώσεις στις οποίες πρέπει να ελεγχθούν περισσότερες από μια συνθήκες στη σειρά και ο έλεγχος να σταματήσει στην πρώτη που ικανοποιείται για να εκτελεστούν οι εντολές της αντίστοιχης ομάδας. Για να γίνει αυτό μπορούν να χρησιμοποιηθούν διαδοχικές δομές της μορφής `If`...`Then`...`Else`, ή να χρησιμοποιηθεί μια πιο σύνθετη μορφή της, η δομή `If`...`Then`...`ElseIf` με γενική μορφή:

```
If συνθήκη1 Then
    εντολή 1_1 }
    εντολή 1_2 }  ομάδα εντολών για συνθήκη1 αληθή
       :
ElseIf συνθήκη2 Then
    εντολή 2_1 }
    εντολή 2_2 }  ομάδα εντολών για συνθήκη2 αληθή
       :
Else
    εντολή v_1 }
    εντολή v_2 }  ομάδα εντολών για όλες τις συνθήκες ψευδείς
       :
End If
```

Στην αρχή ελέγχεται η *συνθήκη1*. Αν είναι αληθής εκτελείται η πρώτη ομάδα εντολών, όπως ακριβώς γίνεται και με την απλή `If`...`Then`...`Else`. Αν η *συνθήκη1* είναι ψευδής ελέγχεται η επόμενη συνθήκη, δηλαδή η *συνθήκη2*. Αν η *συνθήκη2* είναι αληθής, εκτελείται η δεύτερη ομάδα εντολών, ενώ αν είναι ψευδής ελέγχεται η επόμενη συνθήκη κ.ο.κ. έως ότου εξαντληθούν όλες οι συνθήκες, οπότε εκτελούνται οι εντολές της ομάδας `Else` (αν υπάρχουν). Μετά την εκτέλεση μιας ομάδας εντολών, όλες οι άλλες ομάδες που ακολουθούν αγνούνται.




***Σχήμα 8-3.** Σχηματική παράσταση της δομής* `If`...`Then`...`ElseIf`

**Παράδειγμα 8-15.**

Έστω ότι στις μεταβλητές a, b, c έχουν καταχωριστεί τα μήκη των πλευρών ενός τριγώνου. Η δομή:

```
If (a=b) And (b=c) Then
   TriangleType = "Éóüðëåõñï"
ElseIf (a=b) Or (b=c) Or (c=a) Then
   TriangleType = "Éóïóêåëýò"
Else
   TriangleType = "Óêáëåíü"
End If
```

καταχωρίζει στη μεταβλητή TriangleType το είδος του τριγώνου.

**Παράδειγμα 8-16.**

Ο κώδικας:

```
If Voltage < 110 Then
   Message = "Ðïëý ×áìçëÞ ÔÜóç"
ElseIf Voltage < 3000 Then
   Message = "×áìçëÞ ÔÜóç"
ElseIf Voltage < 60000 Then
   Message = "ÌÝóç ÔÜóç"
Else
   Message = "ÕøçëÞ ÔÜóç"
End If
```

καταχωρίζει στη μεταβλητή Message το λεκτικό που αντιστοιχεί στην τιμή της τάσης.

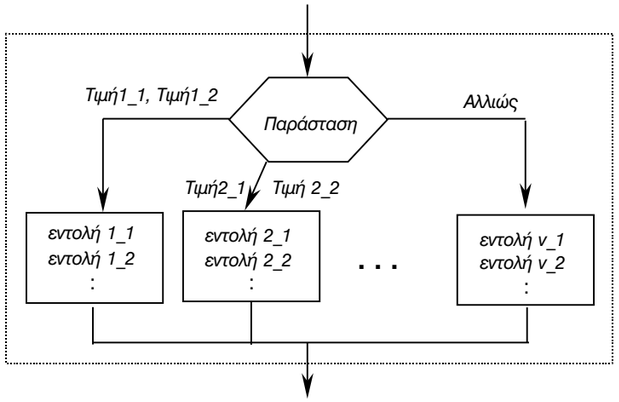
## Η δομή πολλαπλής επιλογής Select Case

Πολλές φορές θέλουμε να συγκρίνουμε την τιμή μιας αριθμητικής ή αλφαριθμητικής παράστασης με πολλές διαφορετικές τιμές και ανάλογα με την τιμή με την οποία είναι ίση να εκτελέσουμε και μια αντίστοιχη ομάδα εντολών. Η λειτουργία αυτή μπορεί να κωδικοποιηθεί όπως είδαμε με δομή If ... Then ... ElseIf. Όμως η VB διαθέτει άλλη μια δομή πολλαπλής επιλογής, τη δομή Select Case, με τη βοήθεια της οποίας οι κωδικοποιήσεις, στην ειδική αυτή περίπτωση, γίνονται πιο ευανάγνωστες. Η γενική μορφή της δομής είναι:

```
Select Case παράσταση
   Case τιμή1_1, τιμή1_2, ...
         εντολή 1_1
         εντολή 1_2        } 1η ομάδα εντολών
            :
   Case τιμή2_1, τιμή2_2,...
         εντολή 2_1
         εντολή 2_2        } 2η ομάδα εντολών
            :
  Case Else
         εντολή v_1
         εντολή v_2        } Ν-ιοστή ομάδα εντολών
            :
End Select
```

Στην αρχή υπολογίζεται η τιμή της παράστασης και στη συνέχεια συγκρίνεται με κάθε τιμή της λίστας τιμών που ακολουθεί την πρώτη Case. Αν η τιμή της παράστασης βρεθεί ίση με μια από αυτές τις τιμές εκτελείται η πρώτη ομάδα εντολών και οι υπόλοιπες ομάδες αγνοούνται. Αν η τιμή της παράστασης δε βρεθεί ίση με κάποια από αυτές τις τιμές αγνοείται η πρώτη ομάδα εντολών και γίνεται έλεγχος με τις τιμές της δεύτερης Case κ.ο.κ. Αν καμιά από τις τιμές των Case δεν είναι ίση με την τιμή της παράστασης εκτελούνται οι εντολές της ομάδας Case Else. Η ύπαρξη της Case Else μέσα στη δομή δεν είναι υποχρεωτική.






***Σχήμα 8-4.*** *Σχηματική παράσταση της δομής* Select Case

**Παραδείγματα 8-17.**

α) Ο κώδικας:

```
Select Case Operator
     Case "+"
          Result = A + B
     Case "-"
          Result = A - B
     Case "*"
          Result = A * B
     Case "/"
          Result = A / B
End Select
```

δρα όπως ακριβώς και μια αριθμομηχανή. Ανάλογα με το σύμβολο που έχει για τιμή η μεταβλητή Operator εκτελεί και την αντίστοιχη πράξη μεταξύ των A και B.

β) Αν CurrentDay πλαίσιο κειμένου στο οποίο γράφουμε ημερομηνίες και DayType ετικέτα στην οποία θέλουμε να εμφανίζεται κατάλληλο λεκτικό ανάλογα με την ημέρα της εβδομάδας που πέφτει η ημερομηνία, ο κώδικας:

```
Select Case Weekday(DateValue(CurrentDay))
    Case 1, 7
        DayType.Caption = "Ìç åñãÜóéìç çìÝñá"
    Case 2, 3, 4, 5, 6
        DayType.Caption = "ÅñãÜóéìç çìÝñá"
End Select
```

Εμφανίζει το λεκτικό "Ìç åñãÜóéìç çìÝñá" για την Κυριακή και τη Δευτέρα και το λεκτικό "ÅñãÜóéìç çìÝñá" για τις υπόλοιπες μέρες της εβδομάδας.

Στη δομή Select Case χρησιμοποιείται μόνο μια παράσταση της οποίας η τιμή υπολογίζεται στην κορυφή της δομής. Αντίθετα, στη δομή If...Then...ElseIf υπάρχει η δυνατότητα χρήσης περισσοτέρων από μια διαφορετικών παραστάσεων, μιας στο If και μιας σε κάθε ElseIf. Αντικατάσταση της δομής If...Then...ElseIf με δομή Select Case μπορεί να γίνει μόνον αν χρησιμοποιείται η ίδια παράσταση μετά το If και μετά από κάθε ElseIf.

Δίπλα από την κωδική λέξη Case, εκτός από τιμές που χωρίζονται με κόμματα, είναι δυνατόν να αναγράφονται και περιοχές τιμών. Αν η περιοχή έχει άνω και κάτω όριο, μεταξύ των ορίων γράφεται η κωδική λέξη To. Αν όμως η περιοχή έχει μόνο κάτω ή μόνο άνω όριο γράφεται η κωδική λέξη Is, το σύμβολο σύγκρισης >, <, >=, <= και το όριο. Δίπλα από την κωδική λέξη Case μπορεί να γραφεί και σύνθετη λογική παράσταση, στην οποία υπάρχει ένας από τους λογικούς συνδέσμους Or, And.

**Παράδειγμα 8-18.**

```
Select Case InValue
    Case Is < 0
         Message = "Áñíçôéêþ ôéìþ"
    Case 1 To 10
         Message = "Ìåôáîý 1 êáé 10"
    Case 11, 12, 17 Ôï 25
         Message = "11, 12 þ ìåôáîý 17 êáé 25"
    Case Is >= 50 And InValue <= 100
         Message = "Ìåôáîý 50 êáé 100"
    Case 101, 102, Is > 110 Message = "101, 102 þ ìåãáëýôåñï ôïõ 110"
End Select
```

Παρατηρούμε ότι η κωδική λέξη `Is` χρησιμοποιείται ως υποκατάστατο της μεταβλητής μόνο την πρώτη φορά. Ας σημειωθεί ότι στην τέταρτη περίπτωση που χρησιμοποιήθηκε ο λογικός σύνδεσμος `And` πρέπει να επαναληφθεί το όνομα της μεταβλητής. Αυτό ισχύει και στην περίπτωση που χρησιμοποιείται λογικός σύνδεσμος `Or`.

## Ένθεση προγραμματιστικών δομών

Στο εσωτερικό μιας προγραμματιστικής δομής μπορούμε να γράψουμε οποιεσδήποτε εντολές ή προγραμματιστικές δομές της VB. Έτσι δημιουργούνται ***ένθετες δομές (nested structures).***

**Παράδειγμα 8-19.**

Το τμήμα κώδικα:

```
If Á > Â Then
   ToPrint = "A>B"
Else
   If Á < Â Then
      ToPrint = "A<B"
   Else
      ToPrint = "A=B"
   End if
End If
```

αποθηκεύει στη μεταβλητή `ToPrint` το λεκτικό αποτέλεσμα της σύγκρισης μεταξύ Á και Â.

**Παράδειγμα 8-20.**

Το τμήμα κώδικα:

```
Select Case CMonth
    Case 1, 3, 5, 7, 8, 10, 12
         LastDay = 31
    Case 4, 6, 9, 11
         LastDay = 30
    Case 2
         If CYear mod 4 <> 0 Then
            LastDay = 28
         Else
            LastDay = 29
         End If
End Select
```

υπολογίζει την τελευταία μέρα του μήνα `CMonth`, τη χρονιά `Cyear`, αν `CYear` είναι μεταξύ 1901 και 2099.



## Ανακεφαλαίωση

Οι λογικές παραστάσεις συντίθενται από σταθερές, μεταβλητές, παραστάσεις, σύμβολα σύγκρισης (=, >, <, <>, >= κ.ά.) και λογικούς συνδέσμους ΚΑΙ (`And`), Ή (`Or`), ΟΧΙ (`Not`).
Η δομή `If...Then...Else` χρησιμοποιείται για την εκτέλεση μιας ομάδας εντολών αν ικανοποιείται κάποια συνθήκη και την εκτέλεση μιας άλλης ομάδας εντολών αν δεν ικανοποιείται η συνθήκη. Όταν πρέπει να ελεγχθούν περισσότερες από μια συνθήκες στη σειρά και ο έλεγχος να σταματήσει στην πρώτη που ικανοποιείται για να εκτελεστούν οι εντολές της αντίστοιχης ομάδας, χρησιμοποιείται η δομή `If...Then...ElseIf`.
Όταν θέλουμε να συγκρίνουμε την τιμή μιας αριθμητικής ή αλφαριθμητικής παράστασης με πολλές διαφορετικές τιμές και ανάλογα με την τιμή με την οποία είναι ίση να εκτελέσουμε και μια αντίστοιχη ομάδα εντολών, χρησιμοποιούμε τη δομή `Select Case`.

## Εργαστηριακές Ασκήσεις

1. Σε ένα παιχνίδι, ο ένας παίκτης σκέφτεται έναν αριθμό. Ο άλλος παίκτης προσπαθεί να τον μαντέψει με μόνη βοήθεια απαντήσεις της μορφής "Μεγαλύτερος" ή "Μικρότερος" του πρώτου παίκτη. Γράψτε πρόγραμμα, το οποίο να δέχεται από τον πρώτο παίκτη τον αριθμό, να τον εξαφανίζει από την οθόνη και να δίνει τις απαντήσεις στο δεύτερο παίκτη. Το πρόγραμμα να μετρά τις προσπάθειες που έγιναν μέχρι ο δεύτερος παίκτης να πετύχει τον αριθμό.

2. Δημιουργήστε φόρμα που να περιέχει πεδίο κειμένου στο οποίο ο χρήστης να πληκτρολογεί ένα ***συνθηματικό (password).*** Με κώδικα να ελέγχεται κατά πόσο το συνθηματικό είναι αποδεκτό ή όχι και να παρουσιάζεται κατάλληλο μήνυμα. Μετά από 3 αποτυχημένες πληκτρολογήσεις το πεδίο να κλειδώνεται και ο χρήστης να μην έχει τη δυνατότητα άλλης προσπάθειας.
   Υπόδειξη: Για να μη φαίνεται το συνθηματικό να γράφονται άσπρα γράμματα σε άσπρο φόντο.

3. Γράψτε πρόγραμμα που να τοποθετεί τρεις αριθμούς, διάφορους ανά δύο, σε αύξουσα σειρά.
   Υπόδειξη: Σε φόρμα να τοποθετήσετε τρία πεδία κειμένου. Το πρώτο πεδίο να είναι πεδίο για τον αριθμό X, το δεύτερο για τον αριθμό Y και το τρίτο για τον αριθμό Z. Ο χρήστης αφού πληκτρολογήσει τιμές μέσα στα πεδία να πατά κατάλληλο πλήκτρο και σε τέταρτο πεδίο να παρουσιάζονται οι αριθμοί ταξινομημένοι. Μεταξύ των αριθμών να τοποθετούνται κόμματα.

4. Δημιουργήστε φόρμα στην οποία όταν γράφεται ένας αριθμός, από 1 έως 7 σε ένα πεδίο κειμένου, να εμφανίζει σε ένα άλλο πεδίο κειμένου το όνομα της ημέρας της εβδομάδος, που αντιστοιχεί σε αυτόν το αριθμό. Να θεωρηθεί ότι η πρώτη μέρα της εβδομάδας είναι η Κυριακή.

5. Γράψτε πρόγραμμα που όταν του δίνουμε μια ημερομηνία μεταξύ 1/3/1900 και 27/02/2100, να υπολογίζει την επόμενη της μέρα, χωρίς να κάνει χρήση των συναρτήσεων ημερομηνίας.
   Υπόδειξη: Σε φόρμα να τοποθετήσετε τρία πεδία κειμένου. Το πρώτο πεδίο να είναι πεδίο για το έτος σε μορφή εεεε, το δεύτερο για το μήνα και το τρίτο για τη μέρα. Ο χρήστης, αφού πληκτρολογήσει τιμές μέσα στα πεδία, να πατά κατάλληλο πλήκτρο και σε τέταρτο πεδίο να παρουσιάζεται η επόμενη ημερομηνία σε μορφή ηη/μμ/εεεε. Αν οι τιμές που έχει πληκτρολογήσει ο χρήστης δεν είναι αποδεκτές στο τέταρτο πεδίο, να εμφανίζεται κατάλληλο μήνυμα (π.χ. μη αποδεκτή ημέρα, μη αποδεκτός μήνας κ.ά. ).

6. Γράψτε πρόγραμμα που όταν του δίνετε μια ημερομηνία να εμφανίζει σε ποια ημέρα της εβδομάδος "πέφτει", π.χ. Δευτέρα, Τρίτη κλπ. 7.

7. Βελτιώστε το πρόγραμμα της εφαρμογής με το ημερολόγιο-χρονόμετρο του μαθήματος 7, ώστε εκτός από την ημερομηνία να εμφανίζει όλογράφως και την ημέρα της εβδομάδας και το μήνα.

8. Η κλίμακα Beaufort χρησιμοποιείται για την εκτίμηση της έντασης του ανέμου. Η κλίμακα αυτή έχει 18 βαθμίδες από τις οποίες χρησιμοποιούνται μόνο οι 13 πρώτες που είναι:

| Βαθμοί | Ονομασία | Μίλια/ώρα |
|---|---|---|
| 0 | Νηνεμία | 1 |
| 1 | Υποπνέων | 1-3 |
| 2 | Ασθενής | 4-7 |
| 3 | Λεπτός | 8-12 |
| 4 | Μέτριος | 13-18 |
| 5 | Λαμπρός | 19-24 |
| 6 | Ισχυρός | 25-31 |
| 7 | Σφοδρός | 32-38 |
| 8 | Ορμητικός | 39-46 |
| 9 | Θύελλα | 47-54 |
| 10 | Ισχυρά θύελλα | 55-63 |
| 11 | Σφοδρά θύελλα | 64-73 |
| 12 | Τυφών | 74-82 |

Γράψτε πρόγραμμα που να δέχεται την ταχύτητα του ανέμου σε μίλια/ώρα και να τυπώνει την ονομασία και την έντασή του σε κλίμακα Beaufort.

## Σημειώσεις:



# Μάθημα 9
# Δομές Επανάληψης

Τελειώνοντας αυτό το μάθημα οι μαθητές θα μπορούν:

- Να επεξηγούν τη χρησιμότητα των δομών επανάληψης.
- Να περιγράφουν τις διαφορές μεταξύ της δομής `Do...Loop` και της δομής `For... Next`.
- Να χρησιμοποιούν κατά περίπτωση την κατάλληλη δομή επανάληψης.
- Να δημιουργούν προγράμματα με ένθετες προγραμματιστικές δομές.

Ένα από τα χαρακτηριστικά των υπολογιστικών συστημάτων είναι ότι μπορούν να εκτελούν πολλές φορές ένα σύνολο από πράξεις και λειτουργίες, οι οποίες τους περιγράφονται μόνο μια φορά. Η περιγραφή μιας επαναληπτικής εκτέλεσης γίνεται με την ένθεση ομάδων εντολών μέσα στις κατάλληλες προγραμματιστικές δομές επανάληψης. Οι προγραμματιστικές δομές επανάληψης είναι δομές οι οποίες μας βοηθούν να δημιουργήσουμε ***βρόχους (loops),*** δηλαδή τμήματα προγράμματος των οποίων οι εντολές εκτελούνται ξανά και ξανά για έναν προκαθορισμένο αριθμό επαναλήψεων, ή ενόσω ικανοποιείται κάποια συνθήκη. Η VB διαθέτει δύο βασικές δομές επανάληψης, τη δομή `Do...Loop` και τη δομή `For...Next`. Τόσο η μια δομή όσο και η άλλη είναι αρκετά ευέλικτες στη σύνταξη τους και μπορούν να καλύψουν κάθε ανάγκη κωδικοποίησης που είναι δυνατόν να προκύψει.

Βρόχος=Θηλιά

## Η δομή `Do . . . Loop`

Στην προγραμματιστική δομή επανάληψης `Do...Loop` χρησιμοποιείται μια λογική ή αριθμητική παράσταση για να καθοριστεί, αν θα εκτελεστούν ξανά οι εντολές που έχουν γραφεί μεταξύ της γραμμής που περιέχει την κωδική λέξη `Do` και της γραμμής που περιέχει την κωδική λέξη `Loop`. Υπάρχουν τέσσερεις παραλλαγές αυτής της δομής. Η γενική μορφή της πρώτης παραλλαγής είναι:

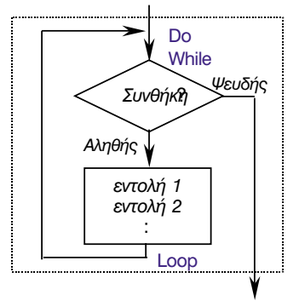
*Φανερόν ότι κύκλω τε έσται η γένεσις τοις απλοίς σώμασι.*
Αριστοτέλης

```
Do While συνθήκη
    εντολή 1
    εντολή 2
    :
Loop
```

ομάδα εντολών βρόχου

όπου *συνθήκη*, η λογική ή αριθμητική παράσταση. Σε αυτή την παραλλαγή γίνονται ανακυκλώσεις μέσα στο βρόχο όσο διάστημα η *συνθήκη* είναι αληθής. Συγκεκριμένα, η δομή λειτουργεί ως εξής:

1. Ελέγχεται η *συνθήκη*.
2. Αν η *συνθήκη* είναι αληθής, εκτελείται η ομάδα των εντολών του βρόχου. Όταν τελειώσει η εκτέλεση και της τελευταίας εντολής, γίνεται επιστροφή στην εντολή `Do While`, όπου ελέγχεται εκ νέου η *συνθήκη* κ.ο.κ.
3. Αν η συνθήκη είναι ψευδής παρακάμπτονται οι εντολές του βρόχου, και η εκτέλεση του προγράμματος συνεχίζεται με την εντολή που ακολουθεί την `Loop`.




***Σχήμα 9-1.*** *Σχηματική παράσταση της δομής* `Do While...Loop`

**Παραδείγματα 9-1.**

Οι τόκοι Τ, που δίνει ένα κεφάλαιο Κ τοκιζόμενο για ένα έτος, με ετήσιο επιτόκιο Ε% είναι:

$$T = \frac{K \times E\%}{100}$$

Στο τμήμα προγράμματος που ακολουθεί, υπολογίζονται τα χρόνια Years που πρέπει να περάσουν για να γίνει το κεφάλαιο NewAmount διπλάσιο του αρχικού Amount, για σταθερό επιτόκιο NperCent το έτος:

```
Years = 0
NewAmount = Amount                        'Áñ÷éêü íÝï êåöÜëáéï
Do While (NewAmount < 2*Amount)
   Years = Years + 1                       'Áëëáãþ Ýôïõò
   Interest = NewAmount * NperCent / 100   'Ôüêïò
   NewAmount = NewAmount + Interest        'ÍÝï êåöÜëáéï
Loop
```

Πράγματι, την πρώτη φορά που θα ελεγχθεί η συνθήκη της εντολής Do, δεν έχει περάσει κανένα έτος (Years=0) και το νέο κεφάλαιο είναι ίσο με το αρχικό (NewAmount=Amount), οπότε θα εκτελεστούν οι εντολές στο εσωτερικό του βρόχου. Οι εντολές αυτές αυξάνουν κατά 1 το έτος, υπολογίζουν τον τόκο Interest για το τρέχον έτος και αυξάνουν το κεφάλαιο. Στην επόμενη ανακύκλωση το αυξημένο κεφάλαιο συγκρίνεται με το αρχικό και αυτό επαναλαμβάνεται μέχρι να παραβιαστεί η συνθήκη του While και να βρεθεί ψευδής. Τότε η εκτέλεστη του προγράμματος συνεχίζεται με τις μετά το Loop εντολές.

Είναι προφανές ότι η ομάδα των εντολών του βρόχου δεν εκτελείται καμία φορά στην περίπτωση που η συνθήκη είναι εξαρχής ψευδής. Επίσης, στο εσωτερικό του βρόχου πρέπει να αλλάζει η τιμή μιας μεταβλητής που συμμετέχει στη συνθήκη. Στην αντίθετη περίπτωση εκτελούνται συνεχώς ανακυκλώσεις και έχουμε ***ατέρμονα βρόχο (endless loop)***.

Οι ατέρμονες βρόχοι αποτελούν παγίδες. Αν το πρόγραμμα πέσει σε ατέρμονα βρόχο, μπορεί να διακοπεί μόνο με εξωτερική επέμβαση.

**Παραδείγματα 9-2.**

Η πράξη της διαίρεσης με ακεραίους αριθμούς μπορεί να προσομοιωθεί με διαδοχικές αφαιρέσεις. Αν Number είναι ο αριθμός, τον οποίο διαιρούμε με τον αριθμό Divisor, Quotient το πηλίκο και Remainder το υπόλοιπο της διαίρεσης, οι εντολές που προσομοιώνουν τη διαίρεση είναι:

```
Quotient = 0
Do While (Number >= Divisor)
  Quotient = Quotient + 1      'Ï äéáéñÝôçò ÷ùñÜ áêüìá ìéá öïñÜ óôï äéáéñåôÝï
  Number = Number - Divisor    'ÄëÛôùóç ôïõ äéáéñåôÝïõ
Loop
Remainder = Number            'Ôåëåêåü õðüëïéðï
```

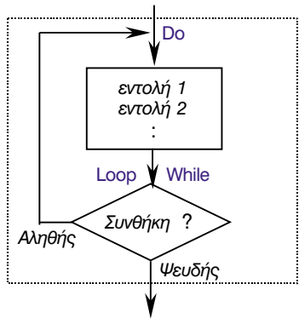
Πράγματι, αν υποθέσουμε ότι ο διαιρετέος Number είναι μεγαλύτερος ή ίσος του διαιρέτη Divisor, την πρώτη φορά που θα ελεγχθεί η συνθήκη της εντολής Do θα βρεθεί αληθής και θα εκτελεστούν οι εντολές του βρόχου. Οι εντολές αυτές καταμετρούν πόσες φορές χωρά ο διαιρέτης στο διαιρετέο και υπολογίζουν το νέο διαιρετέο. Όταν μετά από κάποιες επαναλήψεις ο νέος διαιρετέος βρεθεί μικρότερος του διαιρέτη, η συνθήκη του While θα βρεθεί ψευδής και η εκτέλεστη του προγράμματος θα συνεχιστεί με τις μετά το Loop εντολές. Αν υποθέσουμε ότι ο διαιρετέος Number είναι μικρότερος του διαιρέτη, Divisor, την πρώτη φορά που θα ελεγχθεί η συνθήκη της εντολής Do θα βρεθεί ψευδής και δε θα εκτελεστεί καμιά από τις εντολές του βρόχου. Το υπόλοιπο θα είναι ο διαιρετέος.

Στους πρώτους υπολογιστές η πράξη της διαίρεσης γινόταν με διαδοχικές αφαιρέσεις.

Μια άλλη παραλλαγή της δομής Do...Loop εκτελεί πρώτα τις εντολές του βρόχου και μετά πραγματοποιεί τον έλεγχο της συνθήκης. Αυτό συνεπάγεται ότι οι εντολές του βρόχου θα εκτελεστούν τουλάχιστον μια φορά. Η γενική μορφή της δομής είναι:

```
Do
   εντολή 1
   εντολή 2        ομάδα εντολών βρόχου
   .
Loop While συνθήκη
```






***Σχήμα 9-2.*** *Σχηματική παράσταση της δομής* Do...Loop While.

**Παραδείγματα 9-3.**

Σε μια σκακιέρα στο πρώτο τετραγωνάκι βάζουμε έναν κόκκο στάρι, στο δεύτερο διπλάσιους κόκκους από ότι στο πρώτο, δηλαδή 2, στο τρίτο διπλάσιους κόκκους από ότι στο δεύτερο, δηλαδή 2x2=4, στο τέταρτο διπλάσιους κόκκους από ότι στο τρίτο, δηλαδή 2x4=8 και συνεχίζουμε για να καλύψουμε με αυτή την αντιστοίχιση και τα 64 τετραγωνάκια της σκακιέρας, διπλασιάζοντας σε κάθε νέο τετραγωνάκι τους κόκκους του προηγούμενου.
Το ερώτημα είναι σε πιο τετραγωνάκι "χωρά" η ετήσια παραγωγή σταριού της Ελλάδας!!! Ένα τόνος στάρι έχει περίπου 20.000.000 κόκκους και η Ελλάδα παράγει το χρόνο περίπου 2.100.000 τόνους στάρι;

Ο μύθος λέει ότι τόσο στάρι ζήτησε ο εφευρέτης του σκακιού Sessa από το βασιλιά των Περσών, ως ανταμοιβή του για την εφεύρεση του σκακιού.

```
SeedCroop = Tons * SeedsPerTon          'Ðáñáãùãþ óå êüêêïõò
Square = 1                              'Áñ÷Þ áðü ôï 1ï ôåôñÜãùíï
SeedsOnSquare = 1                       'Êüêêéé óôï ôåôñÜãùíï
Do
   Square = Square + 1                  'Áëëáîå ôåôñÜãùíï
   SeedsOnSquare = 2 * SeedsOnSquare    'Áéðëáóßáóå êüêêïõò
Loop While (SeedsOnSquare >= SeedCroop)
```

Ας σημειωθεί ότι σε αυτό το παράδειγμα προτιμήθηκε ο τρόπος κωδικοποίησης Do...Loop While έναντι του Do While ...Loop, αφού θέλουμε να εκτελεστεί οπωσδήποτε μια φορά ο βρόχος (η ετήσια παραγωγή δεν υπάρχει περίπτωση να είναι 1 κόκκος!).

**Παράδειγμα 9-4.**

Το τμήμα προγράμματος το οποίο υπολογίζει το πλήθος των εμφανίσεων της συμβολοσειράς Word μέσα στη συμβολοσειρά Sentence είναι:

```
Count = 0                'Áñ÷éêÞ ôéìÞ óôïí ìåõñçôÞ
Start = 1                'Áñ÷éêÞ èÝóç áíáæÞôçóçò
Do
    Pos = Instr(Start, Sentence, Word)
    If Pos<>0 Ôhen
       Count = Count + 1            'Áðáñßèìçóç åìöáíßóåùí
       Start = Pos + 1              'Áñ÷Þ áíáæÞôçóçò
    End If
Loop While Pos<>0
```

Αρχικά στο μετρητή Count δίνεται η τιμή 0 και η αναζήτηση αρχίζει από τη θέση Start=1. Λόγω της μορφής του βρόχου η αναζήτηση πραγματοποιείται τουλάχιστον μια φορά. Αν μετά τη θέση Start, μέσα στη Sentence, βρεθεί η Word, καταχωρίζεται η θέση της στην τιμή μεταβλητή Pos, αυξάνεται ο απαριθμητής κατά 1 και η νέα αναζήτηση αρχίζει μια θέση μετά. Αν όμως η αναζήτηση αποβεί άκαρπη, η συνάρτηση Instr αποδίδει την τιμή 0 στη μεταβλητή Pos και διακόπτονται οι ανακυκλώσεις.

Οι άλλες δύο παραλλαγές της δομής χρησιμοποιούνται όταν θέλουμε να εκτελούνται οι εντολές του βρόχου για όσο διάστημα η συνθήκη είναι ψευδής. Εμφανισιακά δε διαφέρουν από τις προηγούμενες παραλλαγές παρά μόνο στο ότι αντί της κωδικής λέξης `While` χρησιμοποιείται η κωδική λέξη `Until` που είναι "αντίθετή" της. Η γενική τους μορφή είναι:

```
Do Until συνθήκη
    εντολή 1
    εντολή 2
      :
Loop
```

```
Do
    εντολή 1
    εντολή 2
      :
Loop Until συνθήκη
```

**Παράδειγμα 9-5.**

Το τμήμα προγράμματος που ακολουθεί προκαλεί μια χρονική καθυστέρηση διάρκειας `Delay` δευτερολέπτων τουλάχιστον. Σε αυτό το χρονικό διάστημα παρουσιάζει το πλήθος των δευτερολέπτων, που έχουν περάσει από τα μεσάνυχτα, στο πλαίσιο κειμένου `Display`.

```
Start = Timer                       'Áñ÷Þ ìÝôñçóçò ÷ñüíïõ
Do Until Timer - Start > Delay      'Óôéãìþ-Áñ÷Þ > Êáèõóôýñçóç ?
    Display = Timer
    DoEvents
Loop
```

Πράγματι, η τιμή της συνάρτησης `Timer` ενημερώνεται από το υπολογιστικό σύστημα και ισούται με το πλήθος των δευτερολέπτων που έχουν περάσει από τα μεσάνυχτα. Στην πρώτη εντολή λαμβάνεται η έναρξη του χρόνου μέτρησης. Η τιμή αυτή αποθηκεύεται στη μεταβλητή `Start`. Στη συνθήκη του βρόχου, υπολογίζεται σε κάθε κύκλο η διαφορά των δευτερολέπτων που έχουν περάσει από την είσοδο στο βρόχο και ελέγχεται, αν αυτή η διαφορά είναι μεγαλύτερη από την τιμή της μεταβλητής `Delay`.

Τη στιγμή που εκτελείται ένας βρόχος δεν εκτελούνται οι διαδικασίες των συμβάντων που προκύπτουν και κατά συνέπεια και ενημέρωση της οθόνης. Το πρόγραμμα μένει εγκλωβισμένο στο βρόχο μέχρι τη στιγμή του τερματισμού του.
Η υπορουτίνα DoEvents είναι μια υπορουτίνα της VB. Η DoEvents επιτρέπει την εκτέλεση, παράλληλα με το βρόχο, και άλλων υπορουτινών διαχείρισης συμβάντων.

**Παράδειγμα 9-6.**

Το τμήμα προγράμματος, το οποίο αφαιρεί τα διπλά κενά διαστήματα που υπάρχουν μεταξύ των λέξεων της πρότασης `Sentence` είναι:

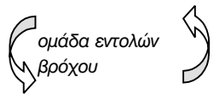
```
Do
   Pos = InStr(Sentence, "  ")    'Âñåò äéðëü êåíü äéÜóôçìá
   If Pos <> 0 Then
      Sentence = Left(Sentence, Pos) & Mid(Sentence, Pos + 2)
   End If
Loop Until Pos = 0                'ÄéÜêïøå áí äåí õðÜñ÷åé Üëëï
```

Πράγματι, η συνάρτηση `InsStr` δίνει στη μεταβλητή Pos τη θέση που έχουν ανιχνευτεί δύο συνεχόμενα κενά διαστήματα. Η συνάρτηση `Left(Sentence, Pos)` δίνει το αριστερό τμήμα της πρότασης μέχρι και το πρώτο κενό διάστημα και η συνάρτηση `Mid(Sentence, Pos+2)` το δεξί τμήμα που αρχίζει δύο θέσεις μετά το το πρώτο κενό διάστημα, επομένως η πρόταση ανασυγκολάται με ένα κενό λιγότερο. Αυτό επαναλαμβάνεται μέχρι να

## Η δομή επανάληψης `For...Next`

Σε πολλές περιπτώσεις είναι εκ των προτέρων γνωστό πόσες φορές θα εκτελεστεί μια ομάδα εντολών. Σε αυτές τις περιπτώσεις η δημιουργία των βρόχων μπορεί να γίνει με την προγραμματιστική δομή `For...Next`. Η γενική μορφή της δομής είναι:

```
For μετρητής = αρχή To τέλος Step βήμα
   εντολή 1
   εντολή 2
     :
Next μετρητής
```

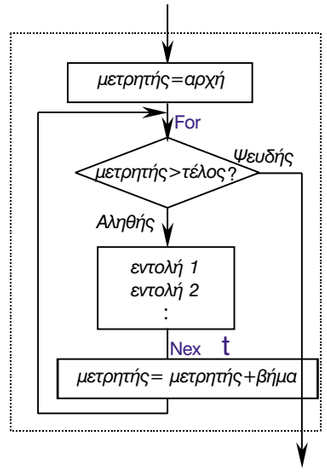



όπου *μετρητής* αριθμητική μεταβλητή και *αρχή, τέλος* και *βήμα* αριθμητικές σταθερές, μεταβλητές ή παραστάσεις.



Η δομή λειτουργεί ως εξής:

1. Υπολογίζονται οι τιμές των *αρχή*, *τέλος* και *βήμα*.
2. Η τιμή της *αρχής* δίνεται για αρχική τιμή στον μετρητή.
3. Αν *μετρητής* *τέλος* εκτελούνται οι εντολές του βρόχου, διαφορετικά το πρόγραμμα παρακάμπτει τις εντολές του βρόχου και συνεχίζει με τη μετά το `Next` εντολή.
4. Στην εντολή `Next` ο *μετρητής* αυξάνεται κατά την τιμή του *βήματος*.
5. Επαναλαμβάνονται τα βήματα 3 και 4, έως ότου να ικανοποιηθεί η συνθήκη *μετρητής* > *τέλος*.




***Σχήμα 9-3.*** *Σχηματική παράσταση της δομής* `For...Next`

**Παράδειγμα 9-7.**

Το τμήμα προγράμματος που ακολουθεί αντικαθιστά το χρώμα του υποβάθρου (ιδιότητα **`BackColor`**) του παραθύρου, σαρώνοντας όλους τους φωτεινούς τόνους του πράσινου, είναι:

```
For Colour = 0 To vbGreen Step 256
    Me.BackColor = Colour
Next Colour
```

Πράγματι η μεταβλητή `Colour` παίρνει τιμές από 0 (μαύρο) μέχρι `vbGeen` (φωτεινό πράσινο) με βήμα 256. Το βήμα 256 δεν επιλέχθηκε τυχαία. Τα χρώματα στην οθόνη δημιουργούνται από έναν αριθμό ο οποίος προκύπτει ως τριάδα δεκαεξαδικών αριθμών (από 0 έως FF=255) που αντιστοιχούν στο κόκκινο, το πράσινο και το μπλε. Η αλλαγή ανά 256 επηρεάζει μόνο το δεύτερο χρώμα, δηλαδή το πράσινο.

Όταν η τιμή του βήματος είναι 1, μπορούμε να παραλείψουμε το τμήμα της εντολής που καθορίζει το βήμα.

**Παράδειγμα 9-8.**

Στη μεταβλητή `Number` υπάρχει ένας ακέραιος γραμμένος κατά τον αμερικανικό τρόπο, σύμφωνα με τον όποίο χρησιμοποιούνται κόμματα για την ομαδοποίηση των ψηφίων σε τριάδες, αντί για τελείες. Το τμήμα του προγράμματος που ακολουθεί, αντικαθιστά τα κόμματα με τελείες.

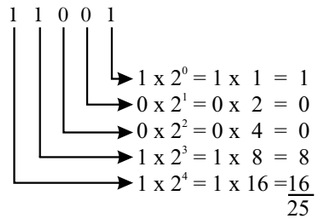
```
For i = 1 To Len(Number)
    If Mid(Number,i,1) = "," Then Mid(Number,i,1) = "."
Next i
```

Με το βρόχο σαρώνονται τα ψηφία του `Number`. Σε καθε κύκλο η συνάρτηση `Mid` απομονώνει ένα ψηφίο, που συγκρίνεται με το κόμμα. Αν το αποτέλεσμα είναι αληθές, γίνεται η αντικατάστασή του στην εντολή `Mid`.

Το *βήμα* μεταβολής του μετρητή μπορεί να είναι και αρνητικό. Είναι ευνόητο ότι σε αυτή την περίπτωση η τιμή του μετρητή δε θα αυξάνεται αλλά θα ελαττώνεται. Οι ανακυκλώσεις στο βρόχο θα σταματήσουν όταν ο *μετρητής* γίνει μικρότερος από την τιμή του *τέλους*.

**Παράδειγμα 9-9.**

Για να μετατρέψουμε έναν αριθμό από το δυαδικό σύστημα στο δυαδικό, αρχίζουμε από το τελευταίο του ψηφίο και σαρώνοντας τον αριθμό προς στην αρχή πολλαπλασιάζουμε κάθε ψηφίο με $2^0$, $2^1$, $2^2$ κλπ. Στο τέλος προσθέτουμε τους επιμέρους παράγοντες. π.χ. για τον αριθμό 11001 έχουμε:




Το τμήμα προγράμματος που ακολουθεί δίνει στη μεταβλητή `DecNum` τη δεκαδική παράσταση του δυαδικού αριθμού, που βρίσκεται στη μεταβλητή `BinNum`.

```
DecNum = 0
Power = 0
For i = Len(BinNum) To 1 Step -1        'Áñ÷Þ áðü ôï ôÝëïò
    Digit = CLng(Mid(BinNum, i, 1))     'Áðïìùíùóç øçößï
    DecNum = Digit * 2 ^ Power + DecNum 'Õðïëüãéóå ìåñéêü Üèñïéóìá
    Power = Power + 1                   'Åðùìåíïò âêèÝôçò
Next i
```

Στη μεταβλητή `DecNum` δίνεται αρχική τιμή 0. Επίσης, στη μεταβλητή `Power`, που παρισάνει τον εκθέτη του 2, δίνεται αρχική τιμή 0. Ο βρόχος σαρώνει τα ψηφία από το τέλος προς την αρχή. Στο βρόχο απομονώνεται ένα δυαδικό ψηφίο (`Digit`) και πολλαπλασιάζεται με την κατάλληλη δύναμη του 2. Επίσης, στο βρόχο προετοιμάζεται και ο επόμενος εκθέτης.

Υπάρχουν ορισμένα θέματα που πρέπει να έχουμε υπ' όψιν μας για τις τιμές των σταθερών, μεταβλητών ή παραστάσεων *αρχή, τέλος* και *βήμα*. Συγκεκριμένα:

- Οι εντολές του βρόχου μπορεί να μην εκτελεστούν καμιά φορά αν:
  - *αρχή* > *τέλος και βήμα* > *0*.
  - *αρχή* < *τέλος και βήμα* < *0*.
- Αν *αρχή* = *τέλος* και *βήμα* ΄Η 0, οι εντολές στο εσωτερικό του βρόχου θα εκτελεστούν μόνο μια φορά.
- Αν *βήμα* = 0, ο βρόχος είναι ατέρμονας.

Στην VB, αντίθετα με ότι ισχύει σε άλλες γλώσσες προγραμματισμού, ο *μετρητής*, η *αρχή,* το τέλος και το *βήμα* επιτρέπεται να παίρνουν και μη ακέραιες τιμές.

**Παράδειγμα 9-10.**

Το τμήμα προγράμματος που ακολουθεί, εμφανίζει στο εσωτερικό του παραθύρου έναν πίνακα ημιτόνων και συνημιτόνων, με βήμα τιμών ανά 0,2 ακτίνια.

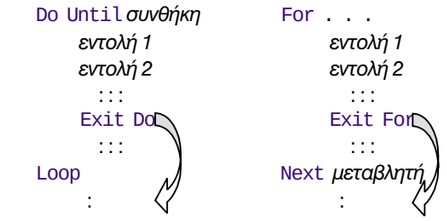
```
Dim w   As Double
Dim pi  As Double
Me.Cls
pi = 4 * Atn(1)
For w = 0 To 2 * pi Step 0.2
    Me.Print w, Sin(w), Cos(w)
Next w
```

Τελειώνοντας, έχουμε να προσθέσουμε ότι δεν πρέπει να αλλάζουμε την τιμή του *μετρητή* μέσα στο βρόχο και ότι η τιμή του *μετρητή* μετά το βρόχο είναι η τελευταία που προκάλεσε τη διακοπή του.



## Τερματισμός ανακύκλωσης και έξοδος από το βρόχο

Αρκετές φορές απαιτείται να γίνει άμεσος τερματισμός της ανακύκλωσης και έξοδος από το βρόχο από κάποιο σημείο του χωρίς να εκτελεστούν οι υπόλοιπες εντολές του ή να πραγματοποιηθούν όλες οι προσχεδιασμένες επαναλήψεις του. Η έξοδος από οποιοδήποτε σημείο του βρόχου, πλην του σημείου ελέγχου της συνθήκης εξόδου, μπορεί να γίνει με την εντολή `Exit For`, για τις δομές τύπου `For`...`Next`, και με την εντολή `Exit Do` για τις δομές τύπου `Do`...`Loop`.




**Παράδειγμα 9-11.**

Το τμήμα προγράμματος που ακολουθεί, σαρώνει τα ψηφία `Digit` του αριθμού που έχει γραφεί σε ένα πλαίσιο κειμένου με όνομα `Number` και αποφαίνεται, αν ο αριθμός είναι δυαδικός ακέραιος ή όχι.

```
IsBinary = True
For i = 1 To Len(Number)
    Digit = Mid(Number,i,1)
    If (Digit <> "0") And (Digit <> "1") Then
       IsBinary = False
       Exit For
    End If
Next Colour
```

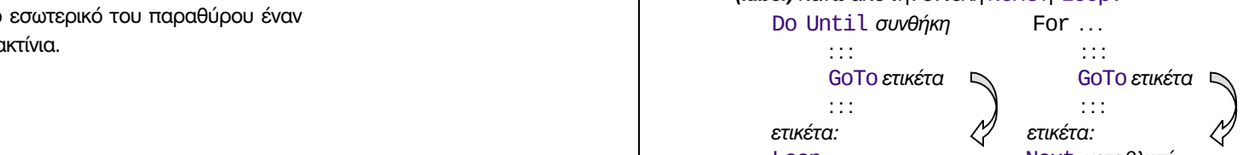
Πράγματι, αρχικά έξω από το βρόχο γίνεται υπόθεση ότι ο αριθμός είναι δυαδικός. Στο εσωτερικό του βρόχου σαρώνονται τα ψηφία του αριθμού και αν βρεθεί ψηφίο διάφορο του 0 ή του 1 η υπόθεση καταρρίπτεται και ζητείται δυναμική έξοδος από το βρόχο, αφού δεν έχει νόημα η σάρωση και των υπολοίπων ψηφίων.

Τόσο η `Exit For` όσο και η `Exit Do` συνδυάζονται σχεδόν πάντα με μια δομή διακλάδωσης `If`...`Ôhen` ή `Select Case` και μπορού να εμφανιστού στο εσωτερικό ενός βρόχου όσες φορές κρίνεται αναγκαίο.

## Άλμα στην επόμενη ανακύκλωση

Μερικές φορές χρειάζεται να διακοπεί μια ανακύκλωση χωρίς να εκτελεστούν όλες οι εντολές του βρόχου και η εκτέλεση του προγράμματος να συνεχίσει με νέα ανακύκλωση από την πρώτη εντολή του βρόχου. Αν και κάποιες γλώσσες υψηλού επιπέδου διαθέτουν κατάλληλη εντολή, η VB δεν παρέχει αυτή τη δυνατότητα άμεσα. Μπορούμε όμως να πετύχουμε έμμεσα την αλλαγή της ροής εκτέλεσης του προγράμματος με μια εντολή `GoTo`, που παραπέμπει σε μια ***ετικέτα (label)*** πάνω από την εντολή `Next` ή `Loop`.

Δεν υπάρχει καμιά σχέση μεταξύ της ετικέτας του προγράμματος και του αντικειμένου ετικέτα που τοποθετείται πάνω στις φόρμες.




Η εντολή `GoTo` για χρόνια αποτελούσε (και για πολλούς αποτελεί ακόμα) το μαύρο πρόβατο του προγραμματισμού. Αν χρησιμοποιηθεί χωρίς μέτρο και για σκοπούς διαφορετικούς από αυτόν που αναφέραμε δημιουργεί προγράμματα μπλεγμένα, τα οποία είναι δύσκολο να τα παρακολουθήσει κανείς. Η εντολή `GoTo` πρέπει να συνδυάζεται πάντα με μια δομή διακλάδωσης `If`...`Ôhen` ή `Select Case`.

**Παράδειγμα 9-12.**

Στο παιχνίδι κρεμάλα (δείτε και παράδειγμα 7-12), όταν ο δεύτερος παίκτης δώσει ένα γράμμα, πρέπει το πρόγραμμα να σαρώσει τη λέξη και όπου βρει το γράμμα να το εμφανίσει στο σκαρίφημα. Το τμήμα προγράμματος που ακολουθεί, σαρώνει τη λέξη `Word` και όπου βρει το γράμμα `Letter` το αντικαθιστά.

```
For i = 1 To Len(Word)
    If Mid(Word,i,1)<>Letter Then GoTo Next_Letter
    Mid(Crypto,i,1) = Letter
       :
       :
Next_Letter:
Next i
```

Είναι προφανές, ότι ένας άλλος προγραμματιστής θα μπορούσε να κάνει διαφορετική κωδικοποίηση. Αλλάζοντας τη συνθήκη στην εντολή `If...Then` μπορεί να δημιουργηθεί ένθεση στις εντολές που ακολουθούν και να αποφευχθεί η χρήση της εντολής `GoTo`. Αν όμως μεσολαβούν πολλές εντολές μέχρι τον τερματισμό του βρόχου και ενδιάμεσα υπάρχουν πολύπλοκες προγραμματιστικές δομές, μια επιπλέον ένθεση (και πιθανόν περισσότερες της μιας) μπορεί να δημιουργήσουν δυσκολία ανάγνωσης, οπότε η χρήση της εντολής `GoTo` βολεύει.

Για την ονομασία των ετικετών ακολουθούμε λίγο πολύ τους κανόνες ονομασίας των μεταβλητών. Χρησιμοποιούμε μόνο γράμματα και ψηφία, δεν αρχίζουμε ποτέ με ψηφίο, δε χρησιμοποιούμε δεσμευμένες λέξεις και φροντίζουμε να δίνουμε ονόματα με μικρό μήκος. Το όνομα μπορεί να περιέχει και ελληνικούς χαρακτήρες αλλά καλό είναι για λόγους γενικότητας να αποφεύγεται η χρήση τους.

Για να ξεχωρίζουν οι ετικέτες από τα άλλα δομικά στοιχεία του προγράμματος, δίπλα από το όνομά τους γράφεται υποχρεωτικά το σύμβολο `(:)`. Ας σημειωθεί ότι αυτό το σύμβολο δε γράφεται στην αναφορά της ετικέτας δίπλα από την εντολή `GoTo`.

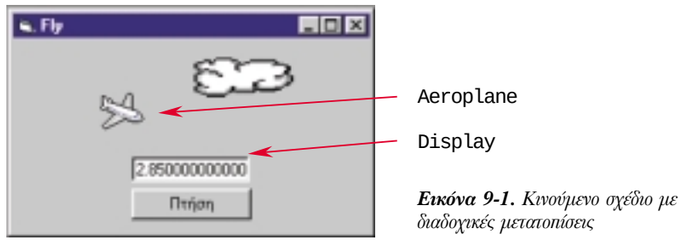
## Ένθεση προγραμματιστικών δομών

Μια δομή ελέγχου μπορεί να τοποθετηθεί στο εσωτερικό άλλης δομής ελέγχου του ίδιου ή διαφορετικού τύπου. Για παράδειγμα μια δομή `Do...Loop` μπορεί να τοποθετηθεί στο εσωτερικό μιας δομής `For...Next` ή σε ένα από τα σκέλη μιας δομής επιλογής `If...Then`. Σε κάποια από τα παραδείγματα που έχουμε δώσει υπάρχουν τέτοιες περιπτώσεις (παραδείγματα 9.4 και 9.6).

Παρατηρήστε ότι για να τονιστεί η έννοια της ένθεσης και για να γίνει πιο αναγνώσιμο το πρόγραμμα, οι εντολές των ένθετων βρόχων τοποθετούνται δεξιότερα. Αυτό παρ' όλο που δεν μας το υπαγορεύει κανένας κανόνας, αποτελεί μια καλή προγραμματιστική συνήθεια, αφού τονίζει την εμβέλεια κάθε προγραμματιστικής δομής.

**Άσκηση 9-1.**

Σε μια φόρμα τοποθετούμε στα αριστερά ένα αντικείμενο ελέγχου εικόνας με όνομα `Aeroplane` και με τη βοήθεια της ιδιότητας **`Picture`** επιλέγουμε να απεικονίζει ένα αεροπλανάκι (αρχείο `...\Common\Graphics\Icons\Industry\Plane.ico`). Να δημιουργηθεί πρόγραμμα στο οποίο το πάτημα ενός πλήκτρου διαταγής να κινεί το αεροπλανάκι από το ένα άκρο της εικόνας στο άλλο.




***Εικόνα 9-1.*** *Κινούμενο σχέδιο με διαδοχικές μετατοπίσεις*



Ο κώδικας του προγράμματος είναι:

```
Private Sub Fly_Click()
    Dim xStart          As Long    'Áñ÷éêÞ èÝóç áåñïðøËíïõ
    Dim xEnd            As Long    'Ôåëéêþ èÝóç áåñïðøËíïõ
    Dim xStep           As Long    'ÂÞìá ìåôáêßíçóçò
    Dim x               As Long    'Ìåôáâëçôþ âñü÷ïõ
    Dim tStartFly       As Double  'Áñ÷Þ ðôÞóçò
    Dim tStartDelay     As Double  'Áñ÷Þ ìÝôñçóçò ÷ñüíïõ
   'Áñ÷éêïðïßçóç ìåôáâëçôþí
    xStart = Aeroplane.Left
    xEnd = Me.Width - Aeroplane.Width
    xStep = (xEnd - xStart) / 100    'ÐôÞóç óå 100 âÞìáôá
    tStartFly = Timer
    For x = xStart To xEnd Step xStep
        Aeroplane.Left = x
       '×ñïíéêÞ êáèõóôÝñçóç 0.05 sec
        tStartDelay = Timer
        Do While Timer - tStartDelay < 0.05
            Display = Timer - tStartFly
            DoEvents
        Loop
    Next x
End Sub
```

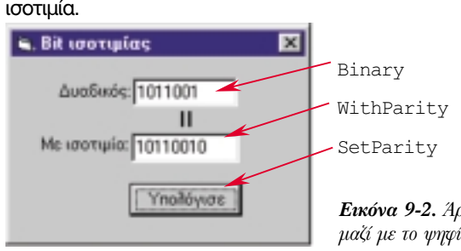
Αρχικά υπολογίζονται οι παράμετροι του βρόχου (`xStart`, `xEnd`, `xStep`) και λαμβάνεται ο χρόνος έναρξης της πτήσης (`tStartFly`). Στο εσωτερικό του βρόχου γίνεται η μετατόπιση του αεροπλάνου στη νέα του θέση (`Aeroplane.Left = x`) και χρονική καθυστέρηση, ώστε να φανεί φυσική η κίνηση του αεροπλάνου.

# Εφαρμογές

## Το bit ισοτιμίας

Οι τεχνικές ανίχνευσης σφαλμάτων στη μετάδοση δεδομένων βασίζονται στην εισαγωγή ***πλεονασμού (redundancy)*** στα προς μετάδοση δεδομένα. Ο πλεονασμός δημιουργείται με την εισαγωγή επιπλέον δυαδικών ψηφίων. Στην πιο απλή περίπτωση εισάγεται ένα μόνο bit στο τέλος ενός byte. Το πρόσθετο ψηφίο ονομάζεται ψηφίο ισοτιμίας. Το ψηφίο ισοτιμίας υπολογίζεται με την εξής λογική: Αν χρησιμοποιείται ***άρτια ισοτιμία (even parity)***, το ψηφίο ισοτιμίας παίρνει την τιμή 0 ή 1, ώστε ο συνολικός αριθμός δυαδικών ψηφίων (bits χαρακτήρα + ψηφίο ισοτιμίας) που έχουν τιμή 1 να είναι άρτιος. Στην περίπτωση της ***περιττής ισοτιμίας (odd parity)*** ο συνολικός αριθμός των δυαδικών ψηφίων πρέπει να είναι περιττός.

Θα δημιουργήσουμε πρόγραμμα που θα εμφανίζει δύο πλαίσια κειμένου και ένα πλήκτρο διαταγής. Στο ένα πλαίσιο κειμένου θα πληκτρολογείται ένας 7-ψήφιος δυαδικός αριθμός και στο πάτημα του πλήκτρου θα εμφανίζεται στο δεύτερο πλαίσιο κειμένου ο αριθμός με άρτια ισοτιμία.




***Εικόνα 9-2.*** *Άρτιο πλήθος bits 1 μαζί με το ψηφίο ισοτιμίας*

Αρχικά πρέπει να σαρώσουμε τα ψηφία του αριθμού και να απαριθμήσουμε τα ψηφία τα οποία ισούνται με 1, στη συνέχεια πρέπει να υπολογίσουμε το ψηφίο ισοτιμίας και να το τοποθετήσουμε στο τέλος του αριθμού.

Για άρτια ισοτιμία, αν το πλήθος των ψηφίων του αριθμού είναι περιττό, θέτουμε 1 το ψηφίο ισοτιμίας και αν το πλήθος των ψηφίων του αριθμού είναι άρτιο το θέτουμε 0, ώστε το συνολικό πλήθος ψηφίων να είναι άρτιο.

Ο κώδικας του προγράμματος είναι:

```
Private Sub SetParity_Click()
    Dim Digit         As String * 1
    Dim ParityBit     As String * 1
    Dim CountOnes     As Integer
    Dim i             As Integer
   'Áðáñßèìçóç øçößùí ßóùí ìå 1
    CountOnes = 0
    For i = 1 To Len(Binary)
        Digit = Mid(Binary, i, 1)
        If Digit = "1" Then CountOnes = CountOnes + 1
    Next i
   'Õðïëïãéóìùò éóïôéìßáò
    ParityBit = CStr(CountOnes Mod 2)
   'Óýíèåóç ôïõ íÝïõ áñèìïý
    WithParity = Binary & ParityBit
End Sub
```

Παρατηρήστε ότι για τον υπολογισμό του ψηφίου ισοτιμίας λαμβάνουμε το υπόλοιπο της διαίρεσης με το 2. Πράγματι το δυνατό υπόλοιπο σε αυτή την περίπτωση είναι 0 ή 1 και μάλιστα το υπόλοιπο είναι 1, αν ο αριθμός είναι περιττός και 0 αν ο αριθμός είναι άρτιος. Επίσης, η συνάρτηση `CStr` μετατρέπει τον αριθμό σε συμβολοσειρά για να είναι δυνατή η συνένωσή του με τον αρχικό αριθμό.

### Μετατροπή ακέραιου δεκαδικού αριθμού σε δυαδικού

Για να μετατρέψουμε έναν ακέραιο δεκαδικό αριθμό Ν σε δυαδικό:

1. Διαιρούμε τον αριθμό Ν με το 2 και παίρνουμε πηλίκο Π και υπόλοιπο Υ. Το υπόλοιπο Υ έχει τιμή 0 ή 1 και αποτελεί το ψηφίο του δυαδικού αριθμού.
2. Θέτουμε το πηλίκο Π στη θέση του Ν και επαναλαμβάνουμε το βήμα 1 μέχρι να βρεθεί πηλίκο ίσο με 0. Κάθε νέο υπόλοιπο το τοποθετούμε στα αριστερά του υπό σύνθεση δυαδικού αριθμού.

Θα δημιουργήσουμε πρόγραμμα που θα εμφανίζει δύο πλαίσια κειμένου και ένα πλήκτρο. Στο ένα πλαίσιο κειμένου, το `Decimal`, θα πληκτρολογείται ένας δεκαδικός αριθμός και στο πάτημα του πλήκτρου `DecToBin` θα εμφανίζεται στο δεύτερο πλαίσιο κειμένου, το `Binary`, ο αριθμός στο δυαδικό σύστημα αρίθμησης. Ο κώδικας του προγράμματος είναι:

```
Private Sub DecToBin_Click()
    Dim N     As Long
    Dim Y     As Long
    N = CLng(Decimal)
    Binary = ""
    Do
        Y = N Mod 2
        N = N \ 2              'Ôï ðçëßêï óôç èÝóç ôïõ Í
        Binary = Y & Binary    'Ôá äéáäï÷éêÜ ðñüëïéðá óôá áñéóôåñÜ
    Loop Until N = 0           'Áí ôï ðçëßêï åßíáé 0 ôåñìÜôéóå
End Sub
```

## Ανακεφαλαίωση

Οι δομές επανάληψης είναι δομές που μας βοηθούν να δημιουργήσουμε βρόχους στο εσωτερικό των οποίων οι εντολές εκτελούνται για έναν προκαθορισμένο αριθμό επαναλήψεων, ή ενόσω ικανοποιείται κάποια συνθήκη.

Στην προγραμματιστική δομή επανάληψης `Do...Loop` χρησιμοποιείται μια λογική παράσταση για να καθοριστεί, αν θα εκτελεστούν ξανά οι εντολές που βρίσκονται μεταξύ της γραμμής `Do` και της γραμμής `Loop`. Υπάρχουν τέσσερις παραλλαγές αυτής της δομής. Η προγραμματιστική δομή `For...Next` χρησιμοποιείται στις περιπτώσεις που είναι εκ των προτέρων γνωστό πόσες φορές θα εκτελεστεί μια ομάδα εντολών.

Η έξοδος από οποιοδήποτε σημείο του βρόχου, πλην του σημείου ελέγχου της συνθήκης εξόδου, μπορεί να γίνει με την εντολή `Exit For`, για τις δομές τύπου `For...Next`, και με την εντολή `Exit Do` για τις δομές τύπου `Do...Loop`.



## Εργαστηριακές Ασκήσεις

1. Γράψτε πρόγραμμα το οποίο να υπολογίζει το χρόνο διπλασιασμού του αρχικού κεφαλαίου του παραδείγματος 9-1.

2. Γράψτε πρόγραμμα που να προσομοιώνει την πράξη του πολλαπλασιασμού με διαδοχικές προσθέσεις.

3. Γράψτε πρόγραμμα το οποίο να προσομοιώνει το πρόβλημα του σταριού που περιγράφεται στο παράδειγμα 9-3.

4. Σε μια φόρμα να τοποθετήσετε ένα πλήκρο με όνομα `Fire`, ένα πλαίσιο κειμένου με όνομα `Display` και ένα πλαίσιο κειμένου με όνομα `Delay`. Στο συμβάν **`Click`** του πλήκτρου `Fire` ενσωματώστε τον κώδικα του παραδείγματος 9.5.
   α) Ζητήστε την εκτέλεση του προγράμματος και αφού δώσετε μια τιμή στο `Delay` κάντε κλικ στο πλήκτρο. Όσο τρέχουν οι τιμές στο `Display`, μετακινήστε το παράθυρο της εφαρμογής.
   β) Απομακρύνετε την υπορουτίνα `DoEvents` και εκτελέστε ξανά το πρόγραμμα. Τι παρατηρείτε; Προσπαθήστε να μετακινήσετε το παράθυρο της εφαρμογής. Τι παρατηρείτε;

5. Κάθε μεσάνυχτα η τιμή της `Timer` μηδενίζεται και η διαφορά `Timer-Start` γίνεται αρνητική. Υπάρχει λοιπόν περίπτωση να μην ικανοποιηθεί ποτέ η συνθήκη και να προκύψει ατέρμονας βρόχος. Αιτιολογήστε.
   Χρησιμοποιήστε τη συνάρτηση `Abs` (απόλυτη τιμή) για να βελτιώσετε τη συνθήκη του βρόχου, ώστε να παρακαμφθεί το πιο πάνω πρόβλημα. Αιτιολογήστε.

6. Γράψτε πρόγραμμα για την άσκηση 9-7. Προσαρμόστε το πρόγραμμα, ώστε να κάνει σάρωση στο κόκκινο και στο μπλε.

7. Γράψτε πρόγραμμα για το παράδειγμα 9-9, για τη μετατροπή δυαδικού αριθμού σε δεκαδικό αριθμό.

8. Βελτιώστε την άσκηση 9-1, με προσθήκη κατακόρυφης κίνησης (αλλαγή της ιδιότητας **`Top`**) ώστε το αεροπλανάκι:
   α) Να προσγειώνεται.
   β) Να κινείται με σκαμπανεβάσματα.
   Υπόδειξη: Με τη γεννήτρια ψευδοτυχαίων αριθμών δημιουργήστε κατακόρυφη μετατόπιση εύρους 10% του ύψους της φόρμας.

9. Υλοποιήστε το πρόγραμμα εμφάνισης ενός αριθμού με άρτια ισοτιμία. Στη συνέχεια:
   α) Βελτιώστε το πρόγραμμα, έτσι ώστε να ελέγχεται πρώτα αν ο αριθμός που δίνεται είναι δυαδικός. Αν ο αριθμός δεν είναι δυαδικός να μην προκύπτει αποτέλεσμα.
   β) Προσθέστε στη φόρμα του προγράμματος ένα πλήκτρο σημείωσης (check box), το οποίο να καθορίζει, αν η υπολογιζόμενη ισοτιμία θα είναι άρτια ή περιττή.

10. Όλοι οι φορολογούμενοι στην Ελλάδα έχουν ένα μοναδικό αριθμό φορολογικού μητρώου. Ο αριθμός αυτός είναι 9-ψήφιος. Το τελευταίο ψηφίο του φορολογικού μητρώου είναι ψηφίο ελέγχου, με την έννοια, ότι χρησιμοποιείται για να διαπιστωθεί κατά πόσο έχουν δοθεί σωστά τα υπόλοιπα ψηφία και προκύπτει ως εξής:

    α) Αρχίζουμε από το προτελευταίο του ψηφίο και σαρώνουμε τον αριθμό προς στην αρχή. Πολλαπλασιάζουμε κάθε ψηφίο με $2^0$, $2^1$, $2^2$ κλπ. Τα αποτελέσματα των γινομένων τα προσθέτουμε. Κάνουμε δηλαδή ότι ακριβώς κάναμε και στο παράδειγμα 9.9.
    β) Λαμβάνουμε το υπόλοιπο (πράξη mod) του αθροίσματος με το 11. Τα δυνατά υπόλοιπα είναι οι αριθμοί 0, 1, 2, ... 9, 10. Αν το υπόλοιπο είναι μονοψήφιος αυτός είναι το ψηφίο ελέγχου. Αν ο αριθμός είναι ο διψήφιος 10, αποκόπτουμε το 1 και ψηφίο ελέγχου είναι το 0.
    Γράψτε πρόγραμμα που σε ένα πλαίσιο κειμένου να δέχεται έναν 9-ψήφιο αριθμό και να διαπιστώνει κατά πόσο αυτός ο αριθμός είναι αριθμός φορολογικού μητρώου.

11. Υλοποιήστε το πρόγραμμα μετατροπής ενός δεκαδικού αριθμού σε δυαδικό.

## Σημειώσεις:



# Μάθημα 10
# Δομημένος προγραμματισμός
# Εφαρμογές

Τελειώνοντας αυτό το μάθημα οι μαθητές θα μπορούν:

- Να εκμεταλλεύονται τα προτερήματα του δομημένου προγραμματισμού.
- Να προγραμματίζουν χρησιμοποιώντας τις αρχές του δομημένου προγραμματισμού.

Στο ***δομημένο προγραμματισμό*** (stuctured programmining) χρησιμοποιούμε τις τρεις βασικές προγραμματιστικές δομές, τη δομή της ακολουθίας, τη δομή της επιλογής και τη δομή της επανάληψης για να συνθέσουμε τα προγράμματά μας.

Οι αρχές του δομημένου προγραμματισμού τέθηκαν από τους Bohm και Jacopini στις αρχές της δεκαετίας του 60. Το 1968 ο Edsger Dijkstra δημοσίευσε μια μελέτη με τίτλο "GoTo statement considered harmful" - η εντολή GoTo θεωρείται επιβλαβής) στην οποία έκανε επίθεση στη συνήθεια χρήσης της εντολής GoTo και θεμελίωνε το δομημένο προγραμματισμό. Οι ιδέες όμως αυτές άρχισαν να εξαπλώνονται μετά το 1973, όταν το περιοδικό Datamation έκανε ειδική αφιέρωση πάνω σε αυτό το θέμα.

Αν και ο δομημένος προγραμματισμός εμφανίστηκε ως προσπάθεια περιορισμού χρήσης των εντολών GoTo μέσα στο πρόγραμμα, εξελίχθηκε όμως σε μεθοδολογία και αποτελεί σήμερα το βασικό τρόπο δόμησης προγραμμάτων. Σε αυτό το μάθημα θα αναπτύξουμε μια σειρά από ολοκληρωμένες εφαρμογές, ώστε να δεθεί περισσότερο η θεωρία με την πράξη και να αποκτηθεί μια εξοικείωση στην ολική αντιμετώπιση των προβλημάτων από τη στιγμή που τίθενται μέχρι τη στιγμή της ολοκλήρωσης της λύσης τους με πρόγραμμα.

## ΄Ενας περίεργος υπολογισμός του αριθμού π

Για τον υπολογισμό του λόγου της περιφέρειας ενός κύκλου προς τη διάμετρό του έχουν βρεθεί πάρα πολλοί τρόποι. ΄Ενας από τους πιο περίεργους και ίσως ο πιο πρωτότυπος είναι και αυτός που βασίζεται στην τύχη!

Ας υποθέσουμε ότι έχουμε έναν τετράγωνο στόχο πλευράς a = 1. Με κέντρο τη μια κορυφή του τετραγώνου και ακτίνα ίση με την πλευρά a χαράσουμε, ένα τεταρτοκύκλιο στο εσωτερικό του. Το εμβαδόν του τεταρτοκυκλίου είναι:

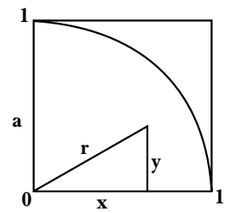
$$E_1 = \frac{\pi\alpha^2}{4} = \frac{\pi}{4}$$

ενώ, το εμβαδόν του τετραγώνου:

$$E_2 = \alpha^2 = 1$$

΄Αρα:

$$\frac{E_1}{E_1} = \frac{\pi}{4} \Leftrightarrow \pi = 4\frac{E_1}{E_1}$$




*Σχήμα 10-1.*

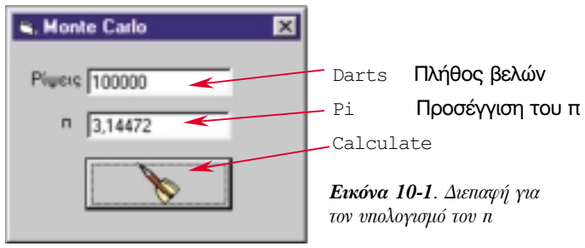
΄Εστω τώρα ότι ρίχνουμε βέλη στο στόχο, ώστε κάθε σημείο του να έχει ίση πιθανότητα να πληγεί από ένα βέλος. Θεωρητικά, αν οι ρίψεις είναι πάρα πολλές, ο λόγος των βελών που θα πλήξουν σημεία μέσα στο τεταρτοκύκλιο προς το σύνολο όλων των βελών που έχουν πλήξει οποιοδήποτε σημείο του στόχου, είναι ίσος με το λόγο των δύο εμβαδών, δηλαδή ίσος με π/4.

Αντί να κάνουμε αυτό το πείραμα εκ του φυσικού, χάνοντας χρόνο και κινδυνεύοντας από τα βέλη, μπορούμε να το προσομοιώσουμε στον υπολογιστή ως εξής: παράγουμε δύο τυχαίους αριθμούς μεταξύ 0 και 1, που τους θεωρούμε σαν τις συντεταγμένες x, y ενός σημείου που βρίσκεται κάπου μέσα στο τετράγωνο. Αν

$$x^2 + y^2 < r^2 \quad \text{ή} \quad x^2 + y^2 < 1$$

το σημείο βρίσκεται μέσα στο τεταρτοκύκλιο. Επαναλαμβάνοντας συνέχεια την πιο πάνω διαδικασία, δηλαδή παράγοντας συνεχώς τυχαία σημεία του τετραγώνου και μετρώντας το

ποσοστό εκείνων που ικανοποιούν την πιο πάνω σχέση, μπορούμε να βρούμε κατά προσέγγιση την τιμή του π/4.




***Εικόνα 10-1**. Διεπαφή για τον υπολογισμό του π*

Το πρόγραμμα για τον υπολογισμό του π με τον τρόπο που αναπτύξαμε είναι το εξής:

```
Private Sub Calculate_Click()
    Dim Count        As Long    'ÌåôñçôÞò
    Dim x            As Single
    Dim y            As Single
    Dim r2           As Single  'Ôåôñùãùíï áðüóôáóçò
    Dim DartsIn      As Single  'ÂÝëç óôï ôåôáñôïêýêëéï
    Randomize
    DartsIn = 0
    For Count = 1 To CLng(Darts)    'Ãéá üëá ôá âÝëç
       x = Rnd                      'Ôõ÷áßï x
       y = Rnd                      'Tõ÷áßï y
       r2 = x ^ 2 + y ^ 2
       If r2 <= 1 Then DartsIn = DartsIn + 1
    Next Count
    pi = 4 * DartsIn / CLng(Darts)
End Sub
```

Τρέχοντας το πρόγραμμα για διαφορετικά πλήθη ρίψεων και με την ίδια πάντα ψευδογεννήτρια, πήραμε τις εξής τιμές:

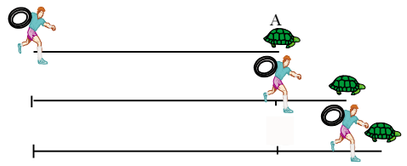
| Πλήθος ρίψεων | Προσέγγιση του π |
|---|---|
| 1.000.000 | 3.14322 |
| 10.000.000 | 3.141281 |
| 100.000.000 | 3.141608 |

Αυτή η μέθοδος έχει ευρύτερη εφαρμογή στον προσεγγιστικό υπολογισμό ποσοτήτων οι οποίες μπορούν να εκφραστούν σε μορφή εμβαδού. Το όνομα της είναι Monte Carlo, από τη συνοικία του πριγκιπάτου του Μονακό, όπου υπάρχει ένα ονομαστό καζίνο, το οποίο θεωρείται ο "εξευγενισμένος" ναός της τύχης.

## Το παράδοξο του ταχύποδα Αχιλλέα και της χελώνας

Γύρω στο 450 π.Χ. ο έλληνας φιλόσοφος Ζήνων αμφισβήτησε την αντίληψη που επικρατούσε μέχρι τότε ότι "το άθροισμα απειραρίθμων ποσοτήτων, ακόμα κι αν αυτές είναι πολύ μικρές, μπορεί να γίνει όσο μεγάλο θέλουμε", διατυπώνοντας το εξής παράδοξο:

Ο γοργοπόδαρος Αχιλλέας βαδίζει 10 φορές γρηγορότερα από μια χελώνα, η οποία προηγείται στο δρόμο κατά ένα στάδιο (περίπου 150m). Πότε θα φτάσει ο Αχιλλέας τη χελώνα;



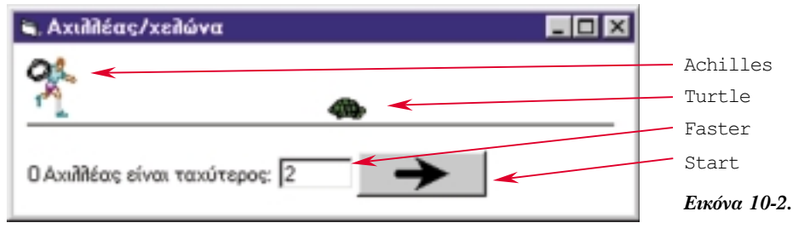

***Σχήμα 10-2.***

Ποτέ! Διότι για να φτάσει ο Αχιλλέας τη χελώνα πρέπει να διανύσει τα 150m και να περάσει από το σημείο Α στο οποίο βρισκόταν η χελώνα αρχικά. Όταν όμως βρεθεί σε αυτό το σημείο, η χελώνα θα έχει προχωρήσει κατά 15m και ο Αχιλλέας θα έχει να καλύψει τη νέα απόσταση. Διανύοντας τώρα ο Αχιλλέας τα 15m η χελώνα βρίσκεται πάλι μπροστά κατά



1,5m κ.ο.κ., ο Αχιλλέας ναι μεν θα πλησιάζει τη χελώνα αλλά δε θα την περάσει ποτέ. Από την εμπειρία μας βέβαια ξέρουμε ότι πάντα προσπερνάμε κάτι που βρίσκεται μπροστά μας και κινείται με μικρότερη ταχύτητ,α γι' αυτό και ο συλλογισμός ότι το άθροισμα απείρων χρονικών διαστημάτων δίνει άπειρο χρόνο καταρρίπτεται.

Στο πρόγραμμα που θα δώσουμε, θεωρούμε ότι υπάρχουν δύο αντικείμενα ελέγχου τύπου εικόνας, που το ένα παριστάνει τον Αχιλλέα και το άλλο παριστάνει τη χελώνα. Επίσης, θεωρούμε ότι υπάρχει ένα πλαίσιο κειμένου στο οποίο ο χρήστης πληκτρολογεί πόσες φορές είναι ταχύτερος ο Αχιλλέας από τη χελώνα και ένα πλήκτρο διαταγής για να αρχίσει η προσομοίωση της κίνησης του Αχιλλέα και της χελώνας.




***Εικόνα 10-2.***

Το πρόγραμμα που αναπτύξαμε είναι το εξής:

```
Option Explicit
Dim AchillesIni      As Integer
Dim TurtleIni        As Integer
Private Sub Form_Load()
   'Áðïìçíùíåõóå áñ÷éêÞ èÝóç Á÷éëëÝá êáé ÷ëÞíáò óôçí ïèüíç
    AchillesIni = Achilles.Left
    TurtleIni = Turtle.Left
End Sub

Private Sub Start_Click()
    Dim AchillesPos      As Double    'ÔñÝ÷ïõóá èÝóç Á÷éëëÝá
    Dim AchillesOldPos   As Double    'Ðñïçãïýìåíç èÝóç Á÷éëëÝá
    Dim TurtlePos        As Double    'ÔñÝ÷ïõóá èÝóç ÷ëÞíáò
    Dim tStartDelay      As Double    'Áñ÷Þ ìÝôñçóçò ÷ñüíïõ
   'Á÷éëëÝáò êáé ÷ëÞíá óôçí áñ÷éêÞ ôïõò èÝóç
    Achilles.Left = AchillesIni:     AchillesPos = AchillesIni
    Turtle.Left = TurtleIni:         TurtlePos = TurtleIni
    Do
        '×ñïíéêÞ êáèõóôÝñçóç
         tStartDelay = Timer
         Do While Timer - tStartDelay < 0.5
              DoEvents
         Loop
        'ÍÝá èÝóç Á÷éëëÝá êáé ÷ëÞíáò
         AchillesOldPos = AchillesPos
         AchillesPos = TurtlePos
         TurtlePos = TurtlePos+(TurtlePos- AchillesOldPos)/Faster
        'Ïåôáöïñü ôùí åéêüíùí óôçí áíôßóôïé÷ç èÝóç
         Achilles.Left = Cint(AchillesPos)
         Turtle.Left = Cint(TurtlePos)
    Loop While AchillesPos < TurtlePos
End Sub
```

Αμέσως μόλις φορτώνεται η φόρμα λαμβάνονται οι αρχικές θέσεις των εικονιδίων που απεικονίζουν τον Αχιλλέα και τη χελώνα. Έτσι, το πάτημα του πλήκτρου `Start` τοποθετεί τα εικονίδια στην αρχική τους θέση. Στη συνέχεια, στο βρόχο προκαλείται χρονική καθυστέρηση για να προφτάσει να δει το μάτι την κίνηση και γίνεται προσομοίωση της κίνησης τοποθετώντας τον Αχιλλέα στη θέση της χελώνας και τη χελώνα σε νέα θέση που υπολογίζεται από το πόσες φορές είναι ταχύτερος της ο Αχιλλέας. Ας σημειωθεί ότι οι μεταβλητές είναι τύπου `Double` για μεγαλύτερη ακρίβεια.

## Καρκινικές λέξεις και προτάσεις

Τα παιχνίδια που μπορούν να γίνουν με το λόγο είναι αμέτρητα και δείχνουν την ευστροφία του ανθρώπινου πνεύματος. Ένα από αυτά είναι και η ανίχνευση και η σύνθεση καρκινικών λέξεων και προτάσεων. Με τον όρο καρκινική λέξη ή πρόταση εννοούμε αυτή που μπορεί να διαβαστεί και ανάποδα, δηλαδή από δεξιά προς τα αριστερά, χωρίς να αλλάξει η έννοια ή το νόημά της. Φυσικά, πολλές φορές, χρειάζονται να γίνουν και μερικές αβαρίες στον τονισμό ή το χωρισμό των λέξεων στις προτάσεις. Αυτό όμως δεν αφαιρεί τίποτε από τη μαγεία που προκαλεί το τελικό αποτέλεσμα. Κλασικά παραδείγματα τέτοιων προτάσεων αποτελούν η πρόταση της γνωστής επιγραφής:

"ΝΙΨΟΝ ΑΝΟΜΗΜΑΤΑ ΜΗ ΜΟΝΑΝ ΟΨΙΝ"

που υπήρχε σε κρήνη της Αγίας Σοφίας, η οποία δε σώζεται, το γνωστό ευθυμολόγημα:

"Madam in Edem I'm Adam"

και η ρήση που αποδίδεται στο Ναπολέοντα:

"Able was I ere I saw Elba"

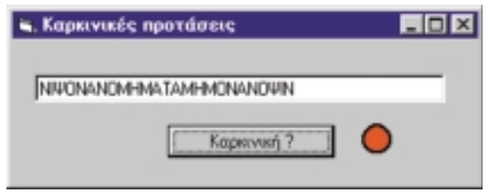
Επίσης οι λέξεις:

άλλα, έχε, όσο, 'Αννα, Σάββας, level, deed, civic

είναι καρκινικές λέξεις.

Από έναν υπολογιστή δεν έχουμε την απαίτηση να δημιουργεί καρκινικές λέξεις ή προτάσεις. Μπορούμε όμως να του αναθέσουμε να ελέγξει αν μια πρόταση ή λέξη είναι καρκινική.

Στο πρόγραμμα που θα δώσουμε η πρόταση που πρόκειται να ελεγχθεί για καρκινισμό, πληκτρολογείται σε ένα πλαίσιο κειμένου με όνομα `Sentence`. Στο πάτημα ενός πλήκτρου διαταγής (με όνομα `Cancer`), αν η πρόταση είναι καρκινική, αλλάζει σε κόκκινο το χρώμα ενός γεωμετρικού σχήματος (με όνομα `Led`), που έχει τη μορφή κύκλου. Αν η πρόταση δεν είναι καρκινική, το χρώμα του γεωμετρικού σχήματος γίνεται πράσινο. Επίσης, όταν εισάγεται νέα πρόταση το χρώμα του γεωμετρικού σχήματος γίνεται άσπρο.




***Εικόνα 10-3.*** *Το παράθυρο μετά τον έλεγχο καρκινικής φράσης*

Το πρόγραμμα που αναπτύξαμε είναι το εξής:

```
Private Sub Sentence_Change()
    Led.FillColor = vbWhite
End Sub

Private Sub Cancer_Click()
    Dim WithoutSpaces   As String
    Dim Letter          As String * 1
    Dim i               As Integer
    Dim Reverse         As String

   'Áöáßñåóç êåíþí äéáóôçìÜôùí, ìåôáôñïðÞ óå êåöáëáßá
    WithoutSpaces = ""
    For i = 1 To Len(Sentence)
        Letter = UCase(Mid(Sentence, i, 1))
        If Letter<>" " Then WithoutSpaces=WithoutSpaces+Letter
    Next i
   'ÁíôéóôñïöÞ ðñüôáóçò
    Reverse = ""
    For i = 1 To Len(WithoutSpaces)
        Letter = Mid(WithoutSpaces, i, 1)
        Reverse = Letter + Reverse
    Next i
```



```
   'Åëåã÷ïò êáñêéíéêüôçôáò
    If Reverse = WithoutSpaces Then
        Sentence = WithoutSpaces
        Led.FillColor = vbRed
    Else
        Led.FillColor = vbGreen
    End If
End Sub
```

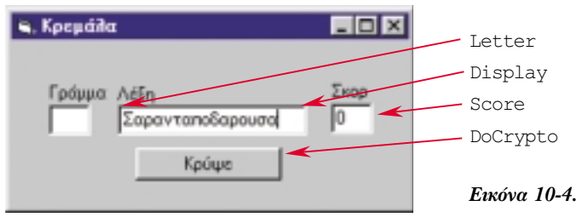
Το πρόγραμμα στον πρώτο βρόχο αφαιρεί τα κενά που υπάρχουν μεταξύ των λέξεων και μετατρέπει όλα τα γράμματα σε πεζά. Στο δεύτερο βρόχο αντιστρέφει τη λέξη και στο τέλος πραγματοποιεί τον τελικό έλεγχο για τον καρκινισμό της λέξης.

## Το παιχνίδι κρεμάλα

Το παιχνίδι "κρεμάλα" είναι από τα πιο δημοφιλή παιχνίδια των μαθητικών θρανίων. Για να μη χαλάσει η παράδοση, τώρα που το θρανίο θα αντικατασταθεί με τον πάγκο που βρίσκεται τοποθετημένος ο υπολογιστής, θα αναπτύξουμε ένα πρόγραμμά με το οποίο θα μπορούμε να παίζουμε αυτό το παιχνίδι με σύντροφο τον υπολογιστή. Ας δούμε όμως πρώτα τους κανόνες του παιχνιδιού. Το παιχνίδι παίζεται με δύο παίκτες που ακολουθούν τα εξής βήματα:

1. Ο ένας παίκτης σκέφτεται μια λέξη και τη γράφει κρυφά σε ένα χαρτί.
2. Φτιάχνει ένα σκαρίφημα της λέξης, από το πρώτο και από το τελευταίο γράμμα και μια σειρά παύλες, ώστε κάθε παύλα να αντιστοιχεί σε ένα ενδιάμεσο γράμμα.
3. Δείχνει το σκαρίφημα στον άλλο παίκτη.
4. Ο δεύτερος παίκτης λέει κάποιο γράμμα.
5. Αν το γράμμα υπάρχει μέσα στη λέξη, ο πρώτος παίκτης το σημειώνει στη θέση της αντίστοιχης παύλας. Αν όμως δεν υπάρχει, χρεώνει με ένα πόντο αποτυχίας το δεύτερο παίκτη.
6. Οι προσπάθειες του δεύτερου παίκτη συνεχίζονται, μέχρι να βρει τη λέξη, οπότε κερδίζει, ή μέχρι να υπερβεί κάποιο αριθμό αποτυχιών, οπότε χάνει.

Στο πρόγραμμα που ακολουθεί, ο υπολογιστής δέχεται την κρυφή λέξη σε ένα πλαίσιο κειμένου από τον πρώτο παίκτη και αναλαμβάνει να τον αντικαταστήσει στο παιχνίδι. Πατώντας ο παίκτης ένα πλήκτρο διαταγής η λέξη αντικαθίσταται στο πλαίσιο κειμένου με τον σκελετό της. Στη συνέχεια ο δεύτερος παίκτης προσπαθεί να μαντέψει ένα ένα γράμμα και παρακολουθεί το σκορ του. Αν συμπληρωθούν όλα τα γράμματα το παιχνίδι αρχίζει από την αρχή.




***Εικόνα 10-4.***

Το πρόγραμμα που αναπτύξαμε είναι το εξής:

```
Option Explicit
Dim Word        As String   'ÌåôáâëçôÞ ãéá ôçí êñõöÞ ëÝîç
Dim Crypto      As String   'Ìåôáâëçôþ ãéá ôïí óêåëåôü
Private Sub Form_Load()
    'Áñ÷éêïðïßçóç ìåôáâëçôþí, áðáíáñáãïðïßçóç áíôéêåéìÝíùí
    Score = 0
    Letter = ""
    Letter.Enabled = False
End Sub
```

```
Private Sub DoCrypto_Click()
    Dim CryptoLen   As Integer

   'Áí ç ëÝÎç Ý÷åé ëéãüÔåñá áðü 2 ãñÜììáôá, äéÜêïøå
    If Len(Display) < 2 Then GoTo ExitDoCrypto
   'ÌåôÜôñåøå ôá ãñÜììáôá óå êåöáëáßá êáé óõãêñÜôçóå ôç ëÝÎç
    Word = UCase(Display)
   'Äçìéïýñãçóå ôï óêåëåôü ôçò ëÝÎçò
    Crypto = Word
    CryptoLen = Len(Crypto)
    Mid(Crypto, 2, CryptoLen - 2) = String(CryptoLen, "-")
    Display = Crypto            'Ðáñïõóßáóå ôï óêåëåôü
   'Åíåñãïðïßçóå êáé áðåíåñãïðïßçóá áíôéêåßìåíá
    DoCrypto.Enabled = False   'Áðåíåñãïðïßçóå ôï ðëÞêôñï
    Letter.Enabled = True      'Åíåñãïðïßçóå êáé
    Letter.SetFocus            'åóôßáóå óôï ðëáßóéï êåéìÝíïõ ãéá ôï ãñÜììá

ExitDoCrypto:
End Sub

Private Sub Letter_LostFocus()
   Dim Exists       As Boole   'Äåßêôçò. True áí ôï ãñÜììá õðÜñ÷åé óôç ëÝÎç
   Dim Pos          As Integer 'ÈÝóç ãñÜììáôïò ìÝóá óôç ëÝÎç
  'Áñ÷éêïðïßçóç ìåôáâëçôþí
   Letter = UCase(Letter)
   Exists = False
  'Ç áíáæÞôçóç áñ÷ßæåé áðü ôï äåýôåñï ãñÜììá
   Pos = 2
   Pos = InStr(Pos, Word, Letter)
  'Áí âñÝèçêå ôï ãñÜììá, óÜñùóå ôç ëÝÎç ãéá åðáíáëÞøåéò ôïõ ßäéïõ ãñÜììáôïò
    Do While (Pos <> 0) And (Pos <> Len(Word))
        Exists = True                       'ÂñÝèçêå ãñÜììá
        Mid(Crypto, Pos, 1) = Letter        'Áíôéêáôýóôçóå ðáýëá ìå ãñÜììá
        Pos = InStr(Pos + 1, Word, Letter)  'ÍÝá áíáæÞôçóç
    Loop
  'Åëåãîå áí âñÝèçêå ôïõëü÷éóôïí ìéá öïñÜ ãñÜììá
    If Exists Then
        Display = Crypto    'Ðáñïõóßáóå ôïí óõìðëçñùìÝíï óêåëåôü
    Else
        Score = Score + 1   'Áýîçóå ôï óêïñ
    End If
    Letter = ""             'ÊáèÜñéóå ôï ãñÜììá
  'Åëåãîå áí âñÝèçêå ç ëÝÎç
    If Crypto <> Word Then
        Letter.SetFocus
    Else
        DoCrypto.Enabled = True
        Letter.Enabled = False
        Display.SetFocus
        Score = 0
    End If
End Sub
```

Η υπορουτίνα εξυπηρέτησης συμβάντων `DoCrypto_Click` εξυπηρετεί τον πρώτο παίκτη, ενώ η `Letter_LostFocus` εξυπηρετεί το δευτέρο.

## Μετατροπή ρωμαϊκών αριθμών σε αραβικούς

Οι Ρωμαίοι χρησιμοποιούσαν ένα κάπως περίεργο σύστημα αρίθμησης. Αν και σήμερα χρησιμοποιείται το αραβικό σύστημα που είναι πιο εύχρηστο, το σύστημα των Ρωμαίων δεν έχει εκτοπιστεί εντελώς από όλες τις ανθρώπινες δραστηριότητες. Στα "καντράν" μερικών ρολογιών ή σε επιγραφικές αναφορές ημερομηνιών, ακόμη και στη μέτρηση σπουδαίων γεγονότων, όπως είναι οι Ολυμπιάδες κ.ά., οι αριθμοί αναγράφονται στο Ρωμαϊκό σύστημα.



Οι Ρωμαίοι χρησιμοποιούσαν 7 γράμματα για την παράσταση των αριθμών, τα:

```
É,      V,      ×,      L,      C,      D,      M
```

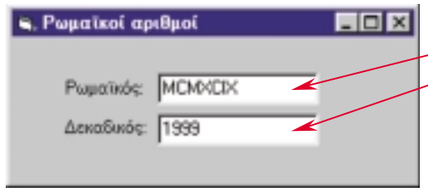
τα οποία αντιστοιχούν στους αριθμούς:

```
1,      5,      10,     50,     100,    500,    1000
```

Με αυτά τα γράμματα και μόνο παρίσταναν όλους τους αριθμούς. Για να δούμε όμως σε ποιον αραβικό αριθμό αντιστοιχεί ένας ρωμαϊκός, πρέπει να γνωρίζουμε τους κανόνες σύνθεσης που χρησιμοποιούσαν. Αυτοί είναι:

1. Όταν έχουμε συνεχόμενα δύο όμοια γράμματα προσθέτουμε τις αξίες τους, π.χ.
   `CCC = 100 + 100 + 100 = 300`
2. Όταν ένα γράμμα έχει μεγαλύτερη αξία από το επόμενό του, η αξία του είναι προσθετική, π.χ.
   `CXVI = 100 + 10 + 5 + 1 = 116`
3. Όταν ένα γράμμα έχει μικρότερη αξία από το επόμενό του, η αξία του είναι αφαιρετική, π.χ.
   `XL = 50 - 10 = 40`

Το πρόγραμμα που θα δημιουργήσουμε δέχεται μια σειρά από γράμματα (20 το πολύ) και όταν προκύψει το συμβάν **`LostFocus`** στο πλαίσιο κειμένου εισαγωγής της, ελέγχει αν αυτή αποτελεί ρωμαϊκό αριθμό και αν αποτελεί, υπολογίζει την τιμή του, διαφορετικά παρουσιάζει κατάλληλο μήνυμα.




***Εικόνα 10-5.*** *Η διεπαφή δεν είναι αναγκαίο να περιέχει πάντα πλήκτρα*

```
Private Sub Roman_LostFocus()
    Dim Letter              As String * 1
    Dim NextLetter          As String * 1
    Dim i                   As Integer
    Dim LetterWeight        As Integer
    Dim NextLetterWeight    As Integer
    If (Len(Roman) < 1) Or (Len(Roman) > 20) Then
        Arabic = "Âêôüò ïñßùí"
        GoTo ExitRoman      'ÔåñìÜôéóá
    End If
    Arabic = 0:    Roman = Roman & " "        'Áñ÷éêÝò ôéìÝò
    For i = 1 To Len(Roman) - 1                'Ãéá êÜèå ãñÜììá ôïõ ñùìáúêïý
        Letter = Mid(Roman, i, 1):   NextLetter = Mid(Roman, i+1, 1)
        Select Case Letter             'Âñåò ôï âÜñïò ôïõ ãñÜììáôïò
            Case "M"
                LetterWeight = 1000
            Case "D"
                LetterWeight = 500
            Case "C"
                LetterWeight = 100
            Case "L"
                LetterWeight = 50
            Case "X"
                LetterWeight = 10
            Case "V"
                LetterWeight = 5
            Case "I"
                LetterWeight = 1
            Case Else
                Arabic = "Ìç ñùìáúêüò"
                GoTo ExitRoman
        End Select
```

```
        Select Case NextLetter          'Âñåò ôï âÜñïò ôïõ ãñÜììáôïò
            Case "M"
                NextLetterWeight = 1000
            Case "D"
                NextLetterWeight = 500
            Case "C"
                NextLetterWeight = 100
            Case "L"
                NextLetterWeight = 50
            Case "X"
                NextLetterWeight = 10
            Case "V"
                NextLetterWeight = 5
            Case "I"
                NextLetterWeight = 1
            Case " "
                NextLetterWeight = 0
        End Select
        If LetterWeight >= NextLetterWeight Then
            Arabic = Arabic + LetterWeight
        Else
            Arabic = Arabic - LetterWeight
        End If
    Next i
    Roman = Left(Roman, Len(Roman) - 1)
    Roman.Setfocus
ExitRoman:
End Sub
```

Παρατηρήστε ότι πριν την είσοδο στο βρόχο αυξάνεται κατά μια θέση ο ρωμαϊκός αριθμός, ώστε να έχει νόημα και η ύπαρξη επομένου γράμματος για το τελευταίο του. Το βάρος του κενού διαστήματος ορίζεται 0.

## Πρόσθεση δυαδικών αριθμών

Δεν είναι λίγες οι φορές που ως μηχανικοί ασχολούμενοι με τους υπολογιστές, πρέπει να κάνουμε πράξεις με δυαδικούς αριθμούς. Σε αυτή την παράγραφο θα αναλύσουμε ένα πρόγραμμα, το οποίο προσθέτει δύο θετικούς ακέραιους αριθμούς γραμμένους στο δυαδικό σύστημα. Το πρόβλημα δεν είναι τόσο απλό όσο φαίνεται, αφού δεν υπάρχει τρόπος απευθείας εισαγωγής δυαδικού αριθμού σε ένα πλαίσιο κειμένου ούτε συναρτήσεις εκτέλεσης πράξεων μεταξύ δυαδικών αριθμών. Η είσοδος των δυαδικών αριθμών μπορεί να γίνει μόνο αν αυτοί εισαχθούν με μορφή συμβολοσειρών και οι πράξεις μεταξύ τους γίνονται εφαρμόζοντας bit προς bit τους αλγόριθμους των βασικών πράξεων. Τα βήματα του αλγορίθμου της πρόσθεσης των αριθμών είναι:

1. Ευθυγράμμιση των ψηφίων τους, δηλαδή τοποθέτηση του ψηφίου των μονάδων του ενός, κάτω από το ψηφίο των μονάδων του άλλου, τοποθέτηση του ψηφίου των δυάδων του ενός, κάτω από το ψηφίο των δυάδων του άλλου κ.ο.κ. Η ευθυγράμμιση επιβάλλεται, διότι ενώ η διαχείριση των συμβολοσειρών γίνεται από τα αριστερά προς τα δεξιά, η πρόσθεση γίνεται από τα δεξιά προς τα αριστερά. Για παράδειγμα, αν δώσουμε τις συμβολοσειρές "10010101" και "110" χωρίς να κάνουμε ευθυγράμμιση των ψηφίων, είναι σαν να προσπαθούμε να κάνουμε την πρόσθεση:

   ```
     10010111
   + 110
   ```

   Η ευθυγράμμιση των ψηφίων γίνεται τοποθετώντας μηδενικά στα αριστερά του μικρότερου αριθμού, ώστε και οι δύο αριθμοί να αποκτήσουν το ίδιο μήκος:

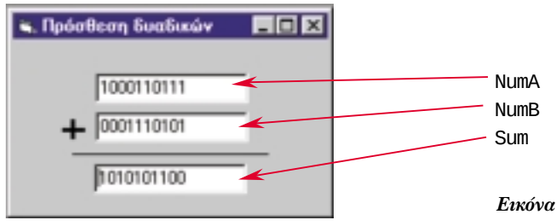
   ```
     10010111
   + 00000110
   ```

2. Πρόσθεση των αντίστοιχων ψηφίων λαμβάνοντας υπ' όψιν και τα διαδοχικά κρατούμενα. Ξεκινώντας από τα δεξιά, και με αρχικό κρατούμενο 0, προσθέτουμε το ψηφίο των μονάδων του ενός με το ψηφίο των μονάδων του άλλου, κινούμενοι προς τα αριστερά



προσθέτουμε το κρατούμενο, που προέκυψε από την προηγούμενη πρόσθεση, με το ψηφίο των δυάδων του ενός και με το ψηφίο των δυάδων του άλλου κ.ο.κ. Τα διαδοχικά ψηφία του αθροίσματος και τα διαδοχικά κρατούμενα υπολογίζονται ως εξής: Αν το επιμέρους άθροισμα είναι μικρότερο από 2 (0 ή 1, δηλαδή μονοψήφιος στο δυαδικό), τότε αυτό αποτελεί και το αντίστοιχο ψηφίο του αθροίσματος. Επίσης, το νέο κρατούμενο είναι 0. Αν όμως το άθροισμα μεταξύ δύο ψηφίων είναι μεγαλύτερο ή ίσο του 2 (2 ή 3, δηλαδή στο δυαδικό 10 ή 11 αντίστοιχα), τότε το αντίστοιχο ψηφίο του αθροίσματος υπολογίζεται αν αφαιρεθεί 2 (αποτέλεσμα 0 ή 1). Επίσης το νέο κρατούμενο είναι 1.

3. Συνένωση των επί μέρους αθροισμάτων για το σχηματισμό της συμβολοσειράς που παριστάνει το άθροισμα.
4. Τοποθέτηση του τελικού κρατουμένου, αν υπάρχει, μπρος από τη συμβολοσειρά άθροισμα και ευθυγράμμιση όλων των συμβολοσειρών.




***Εικόνα 10-6.*** *Παραστατική διεπαφή*

```
Private Sub NumB_LostFocus()
    Dim NumALen     As Integer      'ÌÞêïò Á ðñïóèåôÝïõ
    Dim NumBLen     As Integer      'ÌÞêïò Â ðñïóèåôÝïõ
    Dim BitA        As Integer      'ÔñÝ÷ïí bit Á ðñïóèåôÝïõ
    Dim BitB        As Integer      'ÔñÝ÷ïí bit B ðñïóèåôÝïõ
    Dim BitC        As Integer      'ÔñÝ÷ïí bit áèñïßóìáôïò
    Dim Carry       As Integer      'Êñáôïýìåíï
    Dim i           As Integer      'Äåßêôçò

   'ÅõèõãñÜììéóç øçößùí. Óõìðëçñþóç ìå ìçäåíéêÜ
    NumALen = Len(NumA):    NumBLen = Len(NumB)
    If NumALen < NumBLen Then
        NumA = String(NumBLen - NumALen, "0") + NumA
    Else
        NumB = String(NumALen - NumBLen, "0") + NumB
    End If

   'Ðñüóèåóç áñéèìþí
    Sum = ""
    Carry = 0  'Áñ÷éêü êñáôïýìåíï 0
    For i = Len(NumA) To 1 Step -1   'ÓÜñùóç áðü äåîéÜ óôá áñéóôåñÜ
        BitA = CLng(Mid(NumA, i, 1))       'Áñéèìçôéêþ ôéìÞ ôùí bits
        BitB = CLng(Mid(NumB, i, 1))
        BitC = BitA + BitB + Carry  'Áèñïéóìá
        If BitC < 2 Then            'Ôðïëïãéóìüò êñáôïõìÝíïõ
            Carry = 0
        Else
            Carry = 1
            BitC = BitC - 2
        End If
        Sum = CStr(BitC) & Sum      'ÔïðïèÝôçóç bit óôï ôåëéêü Üèñïéóìá
    Next i

   'ÔïðïèÝôçóç ôåëéêïý êñáôïõìÝíïõ êáé åõèõãñÜììéóç óõìâïëïóåéñþí
    If Carry = 1 Then
        Sum = "1" & Sum
        NumA = "0" & NumA
        NumB = "0" & NumB
    End If

End Sub
```

## Εργαστηριακές Ασκήσεις

1. Υλοποιήστε το πρόγραμμα υπολογισμού του αριθμού π.
2. Υλοποιήστε το πρόγραμμα που προσομοιώνει το παράδοξο του Αχιλλέα και της χελώνας.
3. Υλοποιήστε το πρόγραμμα που ελέγχει αν μια πρόταση είναι καρκινική ή όχι.
4. Υλοποιήστε το πρόγραμμα που προσομοιώνει το παιχνίδι κρεμάλα.
5. Υλοποιήστε το πρόγραμμα που μετατρέπει τους ρωμαϊκούς αριθμούς σε αραβικούς.
6. Υλοποιήστε το πρόγραμμα της πρόσθεσης δύο δυαδικών αριθμών.
7. Γράψτε πρόγραμμα για την προσομοίωση του παιχνιδιού "κορώνα- γράμματα". Συμπληρώστε το πρόγραμμα ώστε να παρουσιάζει την αμεροληψία του εμφανίζοντας σε δύο πλαίσια κειμένου το ποσοστό των ρίψεων κορώνα και το ποσοστό των ρίψεων γράμματα μετά από 1.000.000 ρίψεις.
8. Γράψτε πρόγραμμα που να προσομοιώνει την κλήρωση λαχείου.
   Υπόδειξη: Να παραχθούν 8 αριθμοί μεταξύ 0 και 9 και να τοποθετηθούν στη σειρά.

## Σημειώσεις:



# Μάθημα 11
# Εκσφαλμάτωση

Τελειώνοντας αυτό το μάθημα οι μαθητές θα μπορούν:

- Να διακρίνουν τα είδη των λαθών.
- Να εκμεταλλεύονται τα εργαλεία εκσφαλμάτωσης του περιβάλλοντος εργασίας.
- Να ιχνηλατούν το πρόγραμμά τους και να ανιχνεύουν τα λογικά λάθη που περιέχει.

Όσο πιο μεγάλα και πιο σύνθετα γίνονται τα προγράμματα μας, τόσο πιο εύκολο είναι να υποπέσουμε σε λάθη, τα οποία δύσκολα ανιχνεύονται. Ακόμα και ο πιο έμπειρος και προσεκτικός προγραμματιστής δεν μπορεί να αποφύγει τα προγραμματιστικά λάθη, τα αποκαλούμενα ***ζιζάνια*** ή ***ζουζούνια*** (bugs), και ευθύς εξαρχής να συντάξει προγράμματα που λειτουργούν άψογα σε όλο τον κύκλο ζωής τους. Για το λόγο αυτό μετά από τη συγγραφή του προγράμματος επιβάλλεται να ακολουθήσει η φάση των δοκιμών προκειμένου να εντοπιστούν και να διορθωθούν τα λάθη με τη διαδικασία της εκσφαλμάτωσης (debugging). Σε αυτό το μάθημα θα ασχοληθούμε με τα εργαλεία εκσφαλμάτωσης που διαθέτει η VB, τα οποία μας επιτρέπουν να διαγνώσουμε τα προβλήματα, να προσδιορίσουμε τα λάθη και να προχωρήσουμε στη διόρθωσή τους.

Όλα τα προγράμματα έχουν λάθη.
*Η αρχή του υπεραισιόδοξου προγραμματιστή*

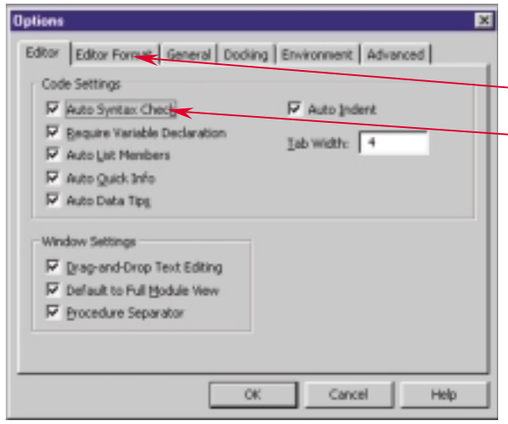
## Κατηγορίες λαθών

Ο βαθμός δυσκολίας της εκσφαλμάτωσης εξαρτάται από τα είδη των λαθών, τα οποία εν γένει μπορεί να είναι:

- Συντακτικά
- Εκτέλεσης
- Λογικά

Αν κάτι μπορεί να πάει στραβά, θα πάει.
*Ο νόμος του Μέρφυ*

Τα ***συντακτικά λάθη (syntax errors)*** οφείλονται στην παραβίαση των κανόνων σύνταξης και γραμματικής της VB. Για παράδειγμα, η παράλειψη μιας παρένθεσης ή ενός κόμματος, η λανθασμένη πληκτρολόγηση μιας κωδικής λέξης ή η ελλειπής σύνταξη μιας προγραμματιστικής δομής, αποτελούν συντακτικά λάθη. Τα λάθη αυτά εντοπίζονται εύκολα γιατί μας τα υποδεικνύει άμεσα ο μεταφραστής του περιβάλλοντος εργασίας. Μάλιστα, το περιβάλλον εργασίας περιέχει τη λειτουργία ***Auto Syntax Check***, η οποία όταν είναι ενεργοποιημένη, κατά την πληκτρολόγηση κάθε γραμμής ελέγχεται ο κώδικας της για κάποια είδη συντακτικών λαθών.




1. Από το μενού επιλέγουμε **Tools ¦ Options.**
2. Διαλέγουμε τον καρτελοδείκτη **Editor.**
3. Μαρκάρουμε το πλαίσιο σημείωσης **Auto Syntax Check.**

***Εικόνα 11-1.*** *Ρύθμιση για αυτόματο συντακτικό και ορθογραφικό έλεγχο*

Αν η γραμμή δεν περιέχει λάθη γίνεται καλλιτεχνική μορφοποίησή της, με τοποθέτηση ενός μόνο κενού διαστήματος μεταξύ των λέξεων, με μπλε χρωματισμό κάποιων κωδικών λέξεων, με τήρηση της ομοιομορφίας κεφαλαίων και πεζών γραμμάτων μέσα στις λέξεις. Αντίθετα, αν η γραμμή περιέχει λάθη χρωματίζεται κόκκινη και ταυτόχρονα εμφανίζεται διαλογικό παράθυρο με μήνυμα που επεξηγεί το λάθος.

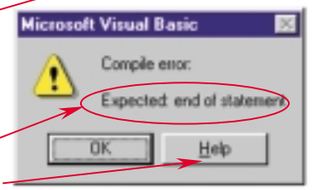
**Παράδειγμα 11-1.**

Έστω ότι εισάγουμε το πρόγραμμα της εφαρμογής "μετατροπή ακεραίου δεκαδικού σε δυαδικό" του μαθήματος 9. Κατά την πληκτρολόγηση της πρώτης γραμμής του αλγορίθμου, από βιασύνη, δεν πληκτρολογούμε την αριστερή παρένθεση. Τη στιγμή που αλλάζουμε γραμμή ο αυτόματος συντακτικός έλεγχος ανιχνεύει το σφάλμα.

```
Private Sub DecToBin_Click()

    Dim N    As Long
    Dim Y    As Long

    N = CLngDecimal)

End Sub
```

Η γραμμή με το λάθος σημειώνεται με κόκκινο χρώμα.

Πλαισιώνεται με μπλε χρώμα κάποιο σημείο αριστερά του οποίου βρίσκεται το λάθος.




Εμφανίζεται διαλογικό παράθυρο με:

- Διαγνωστικό μήνυμα λάθους.
- Πλήκτρο βοήθειας για αναζήτηση κατατοπιστικών πληροφοριών σχετικών με το διαγνωστικό μήνυμα του λάθους.

***Εικόνα 11-2.*** *Ο προγραμματιστής έκλεισε παρένθεση χωρίς να την έχει ανοίξει προηγουμένως.*

Τα μηνύματα του διαλογικού παραθύρου που παρουσιάζει ο μεταφραστής δεν είναι πάντα και τόσο κατατοπιστικά. Το λάθος που προκαλεί την αδυναμία μετάφρασης της εντολής δεν είναι προφανές. Γι' αυτό και ο μεταφραστής απαντά τόσο γενικά. Στη συγκεκριμένη περίπτωση, ο μεταφραστής δεν μπορεί να διακρίνει ότι εμείς θέλουμε να περιλάβουμε τη λέξη `Decimal` μεταξύ παρενθέσεων. Υποθέτει ότι η παρένθεση στο τέλος της γραμμής τοποθετήθηκε από λάθος και μας απαντά ότι "αναμένεται τερματισμός της εντολής" και όχι συνέχειά της με κάτι που θεωρεί περιττό.

Ας σημειωθεί, ότι δεν ανιχνεύονται όλα τα συντακτικά λάθη αυτόματα κατά την εισαγωγή των γραμμών. Για παράδειγμα δεν ανιχνεύεται η παράληψη δήλωσης μεταβλητών, η έλλειψη ενός τμήματος μιας προγραμματιστικής δομής (`If` χωρίς `End If`) κ.ά. Αυτά τα λάθη ανιχνεύονται μόνον κατά την πλήρη μετάφραση της διαδικασίας διαχείρισης του συμβάντος που κωδικοποιούμε. Η πλήρης μετάφραση γίνεται είτε τη στιγμή που προκύπτει το συμβάν είτε τη στιγμή έναρξης του προγράμματος αν αντί για **Run ¦ Start** επιλέξουμε **Run ¦ Start With Full Compile** από τη γραμμή μενού.

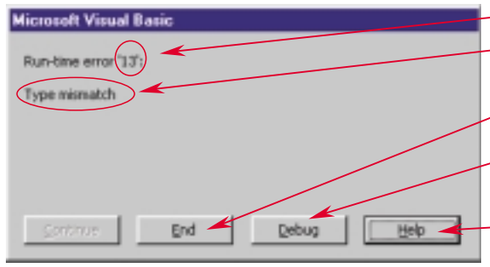
Τα ***λάθη εκτέλεσης (run-time errors)*** οφείλονται σε λειτουργίες που καλείται να εκτελέσει το υπολογιστικό σύστημα κάτω από συνθήκες που είναι ασαφείς ή/και απαγορευτικές. Για παράδειγμα η προσπάθεια διαίρεσης με διαιρέτη μηδέν, η προσπάθεια εισαγωγής συμβολοσειράς στην παράμετρο μιας αριθμητικής συνάρτησης, η προσπάθεια χρήσης μιας αποσυνδεδεμένης περιφερειακής συσκευής ή η προσπάθεια ανάγνωσης ανύπαρκτου αρχείου οδηγούν σε λάθη εκτέλεσης. Τα λάθη αυτά προκαλούν συνήθως διακοπή της εκτέλεσης του προγράμματος και αντιμετωπίζονται είτε με κατάλληλους ελέγχους πριν εκτελεστεί μια διαδικασία που μπορεί να προκαλέσει λάθος (π.χ. έλεγχος αν ο διαιρέτης είναι 0 πριν την εκτέλεση μιας διαίρεσης) ή με την εισαγωγή στο πρόγραμμα κατάλληλων υπορουτινών διαχείρισης λαθών (βλέπε παγίδευση λαθών).

**Παράδειγμα 11-2.**

Έστω ότι έχουμε εισάγει όλο το πρόγραμμα της εφαρμογής "μετατροπή ακεραίου δεκαδικού σε δυαδικό" και ζητάμε την εκτέλεσή του. Επειδή δεν έχουμε προνοήσει να δέχεται το πλαίσιο κειμένου Decimal μόνον αριθμητικές τιμές είναι δυνατόν να προκληθεί λάθος εκτέλεσης.



Πράγματι, αν κατά τη στιγμή της εκτέλεσης του προγράμματος, πληκτρολογήσουμε μια συμβολοσειρά στο πλαίσιο κειμένου Decimal, θα εμφανιστεί το διαλογικό παράθυρο που προειδοποιεί για το λάθος.




Κωδικός λάθους

Περιγραφή λάθους

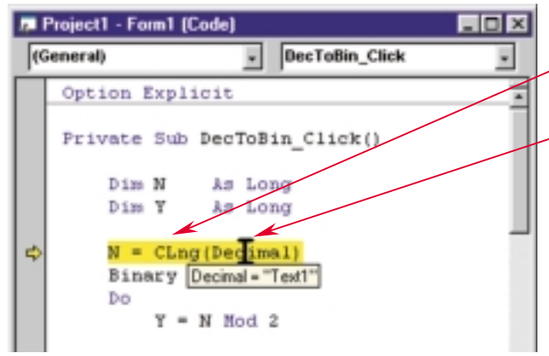
Κάντε κλικ για τερματισμό της εκτέλεσης του προγράμματος.

Κάντε κλικ για μετάβαση στο σημείο του κώδικα που προκαλεί τη διακοπή.

Κάντε κλικ για επεξήγηση της αιτίας που προκάλεσε το σφάλμα.

***Εικόνα 11-3****. Προειδοποιητικό διαλογικό παράθυρο για την εμφάνιση λάθους εκτέλεσης.*

Αν πατήσουμε το πλήκτρο Debug, εμφανίζεται το παράθυρο κώδικα με σημειωμένη την εντολή στην οποία ανιχνεύθηκε το λάθος.




Η γραμμή με το σφάλμα χρωματίζεται κίτρινη.

Αν οδηγήσουμε το δρομέα πάνω στην παράμετρο της συνάρτησης εμφανίζεται ετικέτα που μας πληροφορεί για την τιμή της.

***Εικόνα 11-4.*** *Το παράθυρο κώδικα με το σημείο στο οποίο ανιχνεύθηκε το λάθος.*

Το περιβάλλον εργασίας έχει εισέλθει σε μια κατάσταση που είναι γνωστή ως ***κατάσταση διακοπής (break mode)***, της οποίας τις δυνατότητες θα μελετήσουμε με λεπτομέρεια σε προσεχή παράγραφο.

Ας σημειωθεί ότι αν έχει ζητηθεί να γίνει η εκτέλεση του προγράμματος από εκτελέσιμο αρχείο και όχι κάτω από την εποπτεία του περιβάλλοντος εργασίας παρουσιάζεται διαφορετικό διαλογικό παράθυρο, δεν υπάρχει η δυνατότητα εισαγωγής σε κατάσταση διακοπής και προκαλείται ο τερματισμός του.

Τέλος τα ***λογικά λάθη (logic errors)***, αποτελούν την πιο δύσκολη κατηγορία. Οφείλονται σε λάθος αλγόριθμο ή κωδικοποίηση αλγορίθμου και διαπιστώνονται από την περίεργη συμπεριφορά του προγράμματος. Το πρόγραμμα παρόλο που είναι συντακτικά σωστό και τρέχει χωρίς να εκτελεί απαγορευμένες λειτουργίες δίνει λανθασμένα αποτελέσματα.

**Παράδειγμα 11-3.**

Έστω ότι στο πρόγραμμα "μετατροπή ακεραίου δεκαδικού σε δυαδικό" γράφουμε:

```
Do
    N = N \ 2                'Ôï ðçëßêï óôç èÝóç ôïõ Í
    Y = N Mod 2
    Binary = Y & Binary      'Ôá äéáäï÷éêÜ ðñüëïéðá óôá áñéóôåñÜ
Loop Until N = 0             'Áí ôï ðçëßêï åßíáé 0 ôåñìÜôéóá
```

Αντίστροφη τοποθέτηση των εντολών

αντί για:

```
Do
    Y = N Mod 2              'Ôï ðçëßêï óôç èÝóç ôïõ Í
    N = N \ 2                'Ôá äéáäï÷éêÜ ðñüëïéðá óôá áñéóôåñÜ
Binary = Y & Binary          'Áí ôï ðçëßêï åßíáé 0 ôåñìÜôéóá
Loop Until N = 0
```

Η αντίστροφη τοποθέτηση των εντολών προκαλεί παραγωγή εσφαλμένων αποτελεσμάτων. Σε αυτό το παράδειγμα ένας έλεγχος με αριθμούς που δίνουν γνωστά αποτελέσματα θα μας δείξει ότι κάτι δεν πάει καλά. Σε άλλες όμως περιπτώσεις το σφάλμα είναι διαλείπον και είναι δύσκολη ακόμη και η διαπίστωση της ύπαρξής του. Για άλλα δεδομένα τα αποτελέσματα είναι τα αναμενόμενα και για άλλα όχι.

Τα βήματα εκσφαλμάτωσης σε αυτού του είδους τα λάθη δεν είναι προκαθορισμένα. Ανάλογα με το είδος της ανωμαλίας που παρουσιάζει το πρόγραμμα χρειάζεται να ακολουθήσουμε και διαφορετικό δρόμο και το μόνο βασικό βήμα που μπορούμε να κάνουμε είναι να ελέγξουμε και να αναλύσουμε τα αποτελέσματα χρησιμοποιώντας συνήθως εμπειρικούς κανόνες. Ένας τρόπος είναι να ζητήσουμε την εκτέλεση του προγράμματος με δεδομένα για τα οποία γνωρίζουμε εκ των προτέρων τα σωστά αποτελέσματα. Ωστόσο, ακόμη και στην περίπτωση που το πρόγραμμα θα ανταποκριθεί ορθά για κάποια δεδομένα, ο έλεγχος πρέπει να συνεχιστεί μέχρι να εξασφαλιστεί ότι το πρόγραμμα λειτουργεί σωστά για οποιαδήποτε διαδρομή του αλγορίθμου, για οποιεσδήποτε συνθήκες και με οποιαδήποτε στοιχεία.

Τα λογικά λάθη γίνονται σε σημεία που προκαλούν τη μεγαλύτερη δυνατή ζημιά.

*Βασική αρχή εντοπισμού λαθών*

## Το πρόγραμμα σε κατάσταση διακοπής

Κατά την ανάπτυξη μιας εφαρμογής χρησιμοποιούμε το περιβάλλον εργασίας είτε σε κατάσταση σχεδιασμού (design), οπότε συντάσσουμε και διορθώνουμε τον κώδικά μας χωρίς να μπορούμε να ελέγξουμε τα αποτελέσματα, είτε σε κατάσταση εκτέλεσης (run), οπότε μπορούμε να εκτελέσουμε το πρόγραμμα και να παίρνουμε αποτελέσματα αλλά δεν έχουμε τη δυνατότητα να επεμβαίνουμε και να τροποποιούμε τον κώδικα. Ένας τρίτος τρόπος λειτουργίας του περιβάλλοντος εργασίας είναι η λειτουργία σε κατάσταση διακοπής (break mode). Όταν το περιβάλλον εργασίας βρίσκεται σε κατάσταση διακοπής μπορούμε να πάρουμε στιγμιότυπα της κατάστασης εκτέλεσης, να επέμβουμε στον κώδικα και να υποβοηθήσουμε ποικιλοτρόπως τη διαδικασία εκσφαλμάτωσης. Συγκεκριμένα σε κατάσταση διακοπής μπορούμε:

- Να τροποποιήσουμε τον κώδικα του προγράμματος.
- Να ανιχνεύσουμε τη διαδοχή των διαδικασιών που έχουν εκτελεστεί.
- Να παρακολουθήσουμε τις τιμές μεταβλητών, ιδιοτήτων και παραστάσεων.
- Να αλλάξουμε τις τιμές μεταβλητών και ιδιοτήτων.
- Να αλλάξουμε τη ροή του προγράμματος.
- Να ζητήσουμε την εκτέλεση εντολών εκτός κώδικα προγράμματος.

Το περιβάλλον εργασίας εισέρχεται σε κατάσταση διακοπής, όταν ενώ εκτελεί ένα πρόγραμμα:

- Συναντά ***σημείο διακοπής (breakpoint).***
- Εκτελεί εντολή `Stop`.
- Εκτελεστεί η μέθοδος `Assert` του αντικειμένου `Debug` (`Debug.Assert`) και η συνθήκη που αποτελεί παράμετρό της έχει τιμή `False`.
- Πατήσουμε το συνδυασμό πλήκτρων `Ctrl + Break` ή κάνουμε κλικ στο πλήκτρο `Break` της εργαλειοθήκης εκσφαλμάτωσης.
- Ανιχνευθεί λάθος εκτέλεσης.

## Εισαγωγή σημείων διακοπής

Τα σημεία διακοπής τα χρησιμοποιούμε για να σταματήσουμε την εκτέλεση του κώδικά μας στα σημεία που είναι ύποπτα για πρόκληση προβλημάτων, ώστε να μας δοθεί η δυνατότητα να αξιοποιήσουμε τα εργαλεία εκσφαλμάτωσης. Τα σημεία διακοπής είναι προσωρινά και δεν αποθηκεύονται μαζί με τον κώδικα. Για να εισάγουμε ένα σημείο διακοπής:

1. Τοποθετούμε το δρομέα στο σημείο του κώδικα στο οποίο επιθυμούμε να γίνει η διακοπή της εκτέλεσης.
2. Από τη γραμμή μενού επιλέγουμε **Debug ¦ Toggle Breakpoint** ή πατάμε το πλήκτρο F9.



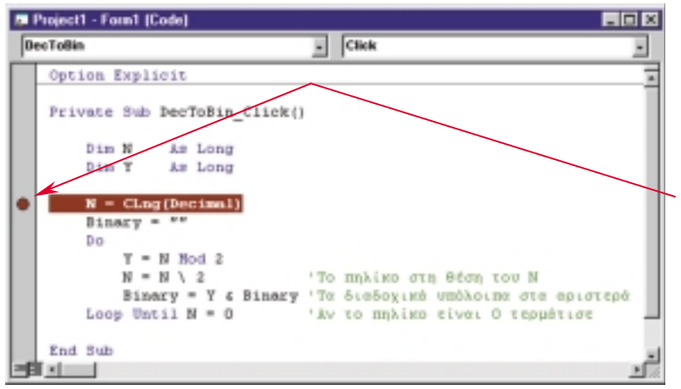
Εναλλακτικά, για να τοποθετήσουμε ένα σημείο διακοπής σε μια γραμμή μπορούμε να κάνουμε κλικ στο περιθώριο του παραθύρου κώδικα, στο ύψος της γραμμής.

Η γραμμή του κώδικα στην οποία εισάγεται το σημείο διακοπής σημειώνεται με μια κουκίδα και αποκτά διαφορετικό χρώμα. Μπορούμε να θέσουμε περισσότερα του ενός σημεία διακοπής. Η εκτέλεση του προγράμματος θα διακόπτεται κάθε φορά που συναντάται ένα τέτοιο σημείο. Για να απομακρύνουμε από τον κώδικα ένα σημείο διακοπής ακολουθούμε την ίδια διαδικασία με αυτή της εισαγωγής του. Εναλλακτικά, επιλέγουμε στο μενού **Debug** την υποεπιλογή **Clear All Breakpoints** και απομακρύνουμε όλα τα σημεία διακοπής.

Το χρώμα των γραμμών στις οποίες έχει τοποθετηθεί σημείο διακοπής καθορίζεται από τον καρτελοδείκτη **Edit Format** του διαλογικού παραθύρου **Options.**

**Παράδειγμα 11-4**.

Τοποθετούμε ένα σημείο διακοπής στη γραμμή `N=CLng(Decimal)`, ώστε πριν γίνει η εκτέλεσή της και πραγματοποιηθεί η κλήση της συνάρτησης `CLng`, που μπορεί να προκαλέσει λάθος εκτέλεση, να έχουμε τη δυνατότητα να ξεκινήσουμε την εκσφαλμάτωση.




***Εικόνα 11-5****. Εισαγωγή σημείου διακοπής*

## Η εντολή `Stop` και η μέθοδος `Debug.Assert`

Η εντολή `Stop` είναι ανάλογη του σημείου διακοπής με τη διαφορά ότι παραμένει ως τμήμα του κώδικα μέχρι να την απομακρύνουμε. Κατά τη μεταγλώττιση οι εντολές `Stop` αντιμετωπίζονται ακριβώς όπως και οι εντολές `End`. Γι' αυτό πρέπει να χρησιμοποιούνται με προσοχή. Ξεχασμένες εντολές `Stop` μπορεί να προκαλέσουν διακοπές και απρόβλεπτα λάθη εκτέλεσης σε μια εφαρμογή που έχει παραδοθεί προς χρήση.

**Παράδειγμα 11-5.**

Διακοπή της εκτέλεσης του προγράμματος με εντολή `Stop`.

```
Private Sub DecToBin_Click()
    Dim N     As Long
    Dim Y     As Long
    Stop
    N = CLng(Decimal)
    Binary = ""
      :
End Sub
```

Με τη μέθοδο `Debug.Assert` μπορούμε να προκαλέσουμε την εισαγωγή του περιβάλλοντος εργασίας σε κατάσταση διακοπής κάτω από ορισμένες συνθήκες. Η μέθοδος αυτή χρησιμοποιεί μια δίτιμο (boolean) έκφραση για να καθορίσει, αν το περιβάλλον εργασίας θα περάσει ή όχι σε κατάσταση διακοπής. Το προτέρημα της `Debug.Assert`, έναντι της εντολής `Stop`, είναι ότι απομακρύνεται από τον κώδικα και δε μεταφράζεται κατά τη δημιουργία εκτελέσιμου προγράμματος σε αρχείο. Έτσι, δεν υπάρχει περίπτωση ξεχασμένες εντολές να προκαλέσουν λάθη.

**Παράδειγμα 11-6.**

Η κωδικοποιήση:

```
Private Sub DecToBin_Click()
    Dim N    As Long
    Dim Y    As Long
    If Not IsNumeric(Decimal) Then Stop
    N = CLng(Decimal)
    Binary = ""
      :
End Sub
```

και η κωδικοποίηση:

```
Private Sub DecToBin_Click()
    Dim N    As Long
    Dim Y    As Long
    Debug.Assert IsNumeric(Decimal)
    N = CLng(Decimal)
    Binary = ""
             :
End Sub
```

είναι ισοδύναμες κατά την εκσφαλμάτωση.

## Διαδικασία εκσφαλμάτωσης

Όπως ήδη έχουμε αναφέρει, η εύρεση των λογικών λαθών είναι μια από τις πιο επίπονες και δύσκολες διαδικασίες. Το λογικό λάθος δεν είναι δυνατόν να το ανιχνεύσει το υπολογιστκό σύστημα. Την ύπαρξή του τη διαπιστώνουμε εκ των υστέρων από την απόκλιση των αποτελεσμάτων σε σχέση με τα αναμενόμενα. Ο μόνος τρόπος ανίχνευσης των λογικών λαθών είναι η τοποθέτηση σημείων διακοπής στην περιοχή του κώδικα που υποπτευόμαστε ότι βρίσκονται, η ελεγχόμενη εκτέλεση του κώδικα σε αυτή την περιοχή και η παρακολούθηση αλλαγής των τιμών των μεταβλητών και των ιδιοτήτων των αντικειμένων.

### Βηματική εκτέλεση του προγράμματος

Όταν το πρόγραμμα βρίσκεται σε κατάσταση διακοπής, είναι δυνατή η εκτέλεση των εντολών του βήμα βήμα. Η ***βηματική (stepwise)*** εκτέλεση πραγματοποιείται με τις επιλογές του μενού **Debug**. Συγκεκριμένα, με την επιλογή:

**Step Into** (Βήμα στο): Εκτελείται η εντολή στην οποία είχε γίνει η διακοπή. Κατόπιν το περιβάλλον εργασίας επανέρχεται σε κατάσταση διακοπής και μάλιστα στην επόμενη εντολή που έχει σειρά. Το πλήκτρο συντόμευσης της επιλογής είναι το πλήκτρο F8. Πατώντας το πλήκτρο F8 ο προγραμματιστής μπορεί να παρακολουθήσει τη διαδρομή που ακολουθεί το πρόγραμμά του, παρατηρώντας το δείκτη εκτέλεσης προγράμματος (κίτρινη διαφανής γραμμή) να πηδά από εντολή σε εντολή.

**Step Over** (Βήμα πάνω από): Εκτελείται η εντολή στην οποία είχε γίνει η διακοπή. Η μόνη διαφορά της **Step Over** σε σχέση με την **Step Into** είναι ότι στην περίπτωση που θα συναντήσει υπορουτίνες και συναρτήσεις μέσα στον κώδικα, τις εκτελεί με μιας και όχι εντολή προς εντολή. Έτσι είναι δυνατό να επιταχυνθεί κάπως η διαδικασίας της εκσφαλμάτωσης.

**Step Out** (Βήμα εκτός): Εκτελείται το υπόλοιπο μιας υπορουτίνας ή μιας συνάρτησης και ο δείκτης εκτέλεσης προγράμματος μεταφέρεται στην επόμενη γραμμή της καλούσας διαδικασίας.

**Run To Cursor** (Εκτέλεση μέχρι το δρομέα): Εκτελείται το υπόλοιπο του προγράμματος μέχρι του σημείου που έχει τοποθετηθεί ο δρομέας κειμένου μέσα στον κώδικα. Λειτουργεί δηλαδή σαν να έχει τοποθετηθεί ένα σημείο διακοπής στη γραμμή που βρίσκεται ο δρομέας.



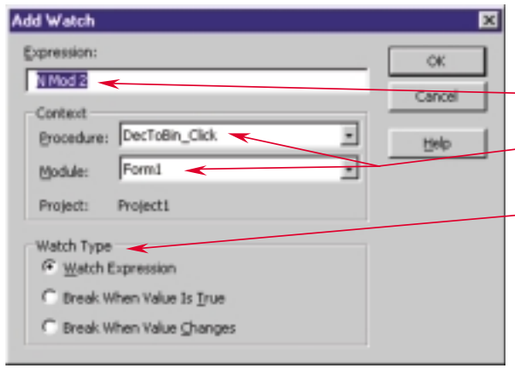
**Set Next Statement** (Θέσε ως επόμενη εντολή): Εντολοδοτείται η παράκαμψη της σειράς των εντολών (χωρίς να πραγματοποιηθεί καμιά εκτέλεση) και η τοποθέτηση της διακοπής στην εντολή που έχουμε υποδείξει με το δρομέα. Χρησιμοποιείται συνήθως για να επαναφέρουμε το σημείο εκτέλεσης των εντολών κάποιες εντολές πίσω και να παρακολουθήσουμε ξανά τον τρόπο εκτέλεσης του προγράμματος.

### ΄Ελεγχος τιμών μεταβλητών και εκφράσεων

Το παράθυρο παρακολούθησης

Εκτελώντας το πρόγραμμα εντολή εντολή, μπορούμε να δούμε τις τιμές των μεταβλητών και των ιδιοτήτων αντικειμένων τοποθετώντας επάνω στο όνομά τους το δρομέα του ποντικιού. Έχουμε δει ότι δίπλα τους εμφανίζεται αυτόματα μια ετικέτα με την τιμή τους. Όμως, για να έχουμε κάπου συγκεντρωμένες όλες τις μεταβλητές και τις παραστάσεις, των οποίων η μεταβολή των τιμών μας ενδιαφέρει, χρησιμοποιούμε το παράθυρο παρακολούθησης (Watch Window). Σε αυτό το παράθυρο προβάλλονται, όχι μόνο οι τιμές των μεταβλητών και των παραστάσεων που έχουμε εισάγει, αλλά και πληροφορίες για την προγραμματιστική μονάδα και τη διαδικασία μέσα στην οποία έχουν την αναφερόμενη τιμή. Επίσης, από το λογισμικό που διαχειρίζεται αυτό το παράθυρο, μπορούμε να ζητήσουμε το πέρασμα υπό συνθήκη σε κατάσταση διακοπής, δηλαδή τη μετάπτωση σε κατάσταση διακοπής, όταν μια από τις υπό παρακολούθηση παραστάσεις αποκτήσει την τιμή `True` ή αλλάξει τιμή. Αυτή η δυνατότητα είναι πολύ εξυπηρετική, αφού ο προγραμματιστής δεν είναι υποχρεωμένος να παρακολουθεί συνεχώς τις τιμές ιδιαίτερα όταν η μεταβολή τους γίνεται μέσα σε ένα βρόχο. Για την εισαγωγή μιας μεταβλητής ή παράστασης στο παράθυρο παρακολούθησης πραγματοποιούμε τα εξής βήματα:

Η παράσταση δεν είναι αναγκαίο να υπάρχει και μέσα στο πρόγραμμα.




1. Από το μενού **Debug** επιλέγουμε **Add Watch** οπότε εμφανίζεται το διαλογικό παράθυρο.
2. Στο πλαίσιο **Expression** πληκτρολογούμε την παράσταση ή τη μεταβλητή.
3. Στην ομάδα **Context** ορίζουμε την περιοχή ελέγχου της παράστασης, δηλαδή, σε ποια σημεία του κώδικα μας ενδιαφέρει η τιμή της.
4. Στην ομάδα **Watch Type** επιλέγουμε τον τρόπο απόκρισης του περιβάλλοντος εργασίας στην τιμή της υπό παρακολούθηση παράστασης.

***Εικόνα 11-6.*** *Εισαγωγή παράστασης στο παράθυρο παρακολούθησης*

Αν, πριν ανοίξουμε το διαλογικό παράθυρο **Add Watch**, ο δρομέας βρίσκεται πάνω στο όνομα μιας μεταβλητής ή μιας ιδιότητας, το πλαίσιο κειμένου **Expression** συμπληρώνεται αυτόματα. Επίσης, αν επιλέξουμε πρώτα μια παράσταση και στη συνέχεια ανοίξουμε το διαλογικό παράθυρο **Add Watch**, το πλαίσιο κειμένου **Expression** συμπληρώνεται αυτόματα με τον κώδικα της παράστασης.

Στην ομάδα **Context** συμπληρώνουμε τα πλαίσια **Procedure** και **Module** επιλέγοντας τη διαδικασία ή όλες τις διαδικασίες και την προγραμματιστική μονάδα ή όλες τις προγραμματιστικές μονάδες στις οποίες θέλουμε να παρακολουθούμε τις εκφράσεις.

Στην ομάδα **Watch Type** επιλέγουμε τον τρόπο απόκρισης του περιβάλλοντος εργασίας στην υπό παρακολούθηση έκφραση. Συγκεκριμένα επιλέγουμε:

**Watch Expression** για απλή εμφάνιση της τιμής της έκφρασης

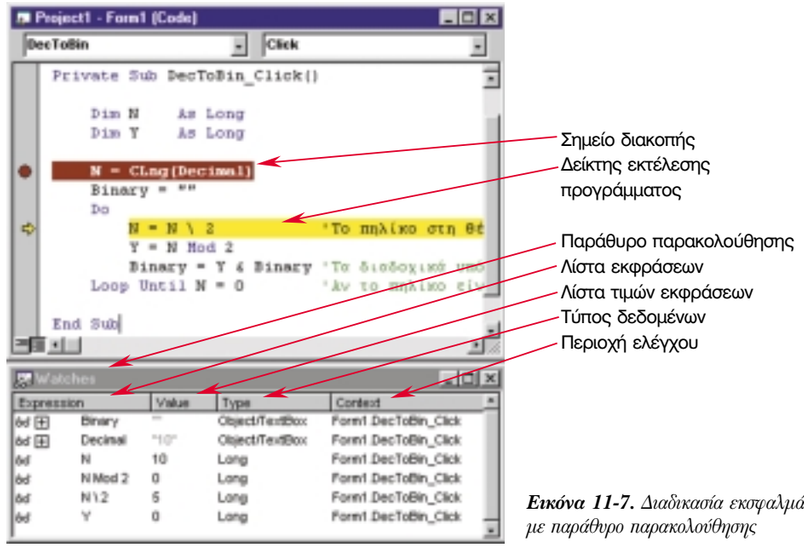
**Break When Value is True** για διακοπή της εκτέλεσης κάθε φορά που η τιμή της έκφρασης γίνεται αληθής.

**Break When Value Changes** για διακοπή της εκτέλεσης, όταν η τιμή της έκφρασης αλλάξει.

Κάθε έκφραση παρακολούθησης που εμφανίζεται στο παράθυρο μπορεί να διορθωθεί ή να διαγραφεί. Για να διορθώσουμε μια έκφραση επιλέγουμε την έκφραση και από το μενού **Debug** επιλέγουμε **Edit Watch**. Για να διαγράψουμε μια έκφραση την επιλέγουμε και πατάμε το πλήκτρο `Delete`.

**Άσκηση 11-1.**

Στο πρόγραμμα "μετατροπή ακεραίου δεκαδικού σε δυαδικό" έχουμε κάνει λάθος στην κωδικοποίηση και δε λαμβάνουμε σωστά αποτελέσματα. Να τοποθετηθεί σημείο διακοπής και να εισαχθούν στο παράθυρο παρακολούθησης μεταβλητές και εκφράσεις.




***Εικόνα 11-7.*** *Διαδικασία εκσφαλμάτωσης με παράθυρο παρακολούθησης*

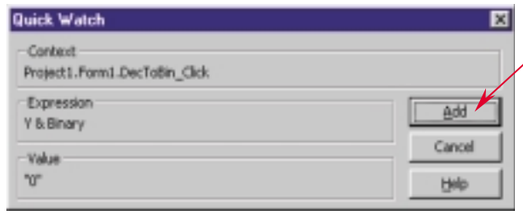
Πραγματοποιούμε βηματική εκτέλεση και παρακολουθούμε τις τιμές των μεταβλητών και των εκφράσεων για να εντοπίσουμε το σημείο που προκαλείται εκτροπή των τιμών από τα αναμενόμενα.

Το παράθυρο γρήγορου ελέγχου

Στην περίπτωση που θέλουμε να δούμε και την τιμή μιας μεταβλητής, ιδιότητας ή παράστασης χωρίς να γίνει υποχρεωτικά η εισαγωγή της στο παράθυρο παρακολούθησης, την επιλέγουμε μέσα στον κώδικα και ανοίγουμε ένα ***παράθυρο γρήγορου ελέγχου (Quick Watch)*** από το μενού **Debug.**

**Άσκηση 11-2.**

Στον κώδικα της πιο πάνω άσκησης επιλέγουμε την παράσταση Y & Binary και ανοίγουμε το παράθυρο γρήγορου ελέγχου.

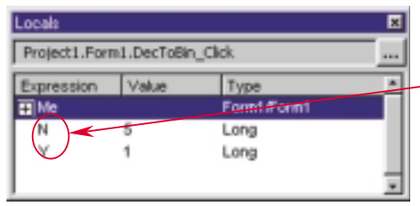



Για να συνεχιστεί ο έλεγχος της παράστασης μπορούμε να την προσθέσουμε στο παράθυρο παρακολούθησης.

***Εικόνα 11-8.*** *Διαδικασία εκσφαλμάτωσης με παράθυρο γρήγορου ελέγχου*

Το παράθυρο τοπικών μεταβλητών

Η εισαγωγή μεταβλητών στο παράθυρο παρακολούθησης απαιτεί χρόνο από τον προγραμματιστή. Το πρόβλημα γίνεται ιδιαίτερα έντονο σε προγράμματα με πολλές διαδικασίες συμβάντων. Η λύση δίνεται με τη χρήση του παραθύρου τοπικών μεταβλητών (Locals Window). Το παράθυρο αυτό εμφανίζεται επιλέγοντας **View ¦ Locals Window.**




Μεταβαίνοντας από διαδικασία σε διαδικασία το παράθυρο τοπικών μεταβλητών ενημερώνεται αυτόματα με τις τρέχουσες τοπικές μεταβλητές.

***Εικόνα 11-9.*** *Το παράθυρο τοπικών μεταβλητών*



## Το παράθυρο άμεσης εκτέλεσης εντολών

Κατά την εκσφαλμάτωση ο προγραμματιστής χρειάζεται ένα είδος προχείρου για να κρατά το ιστορικό τιμών των μεταβλητών και των παραστάσεων, να πραγματοποιεί δοκιμές και να εκτελεί λειτουργίες που δεν του επιτρέπουν τα άλλα βοηθητικά παράθυρα που ήδη αναφέραμε. Το παράθυρο άμεσης εκτέλεσης εντολών (Immediate Window) χρησιμοποιείται με αυτό το σκοπό. Το παράθυρο αυτό εμφανίζεται επιλέγοντας από τη γραμμή μενού **View ¦ Immediate Window**. Στο παράθυρο άμεσης εκτέλεσης μπορούμε να:

- γράψουμε εντολές, οι οποίες εκτελούνται άμεσα τη στιγμή που πατάμε το πλήκτρο `Enter` για να αλλάξουμε γραμμή. Συνήθως χρησιμοποιούμε αυτή τη δυνατότητα, για να αποδώσουμε τιμές σε μεταβλητές ή ιδιότητες και να συνεχίσουμε την εκτέλεση του προγράμματος με τις νέες τιμές.
- ελέγξουμε τις τιμές παραστάσεων και μεταβλητών χωρίς να χρησιμοποιήσουμε κατ' ανάγκη το παράθυρο παρακολούθησης. Αυτός ο τρόπος ελέγχου έχει το επιπλέον πλεονέκτημα ότι κρατείται το ιστορικό των μεταβολών στην οθόνη (στο παράθυρο παρακολούθησης μπορούμε να δούμε μόνο τις τρέχουσες τιμές).

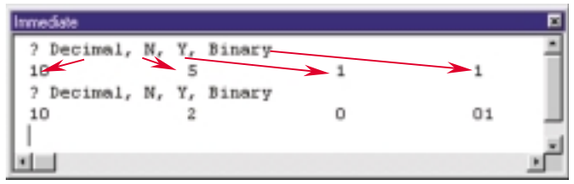
Για να ελέγξουμε τις τιμές παραστάσεων χρησιμοποιούμε τη μέθοδο `Print` με παραμέτρους τις παραστάσεις για να τυπωθούν οι τιμές τους. Έχουμε δύο δυνατότητες:

- Να πληκτρολογήσουμε απευθείας τη μέθοδο `Print` στο παράθυρο. Ας σημειωθεί ότι εδώ μπορούμε να χρησιμοποιήσουμε και το σύμβολο (?) ως σύντμηση της κωδικής λέξης `Print`.
- Να συμπεριλάβουμε στον κώδικά μας εντολές με τη μέθοδο `Print` του αντικειμένου Debug, ώστε να γίνει η εκτύπωση ακόμα και αν το περιβάλλον δε βρίσκεται σε κατάσταση διακοπής.

Οι εντολές `Debug.Print` απομακρύνονται αυτόματα από τον κώδικα τη στιγμή της μετάφρασης του προγράμματος σε εκτελέσιμο αρχείο.

**Άσκηση 11-3.**

Ως συνέχεια της άσκησης 11-1, σε δύο διαδοχικές ανακυκλώσεις να τυπωθούν, στο παράθυρο άμεσης εκτέλεσης, οι τιμές των μεταβλητών που μας ενδιαφέρουν.

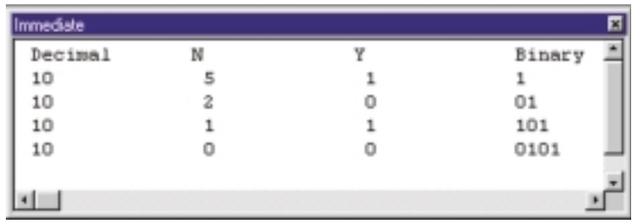



***Εικόνα 11-10.*** *Άμεση εκτέλεση εντολών*

**Άσκηση 11-4.**

Παρεμβάλλουμε εντολές `Debug.Print` μέσα στον κώδικα.

```
N = CLng(Decimal)
Binary = ""
Debug.Print "Decimal", "N", "Y", "Binary"
Do
    N = N \ 2              'Ôï ðçëßêï óôïç èÝóç ôïõ Í
    Y = N Mod 2
    Binary = Y & Binary      'Ôá äéáäï÷éêÜ ðñüëïéðá óôá áñéóôåñÜ
  Debug.Print Decimal, N, Y, Binary
Loop Until N = 0             'Áí ôï ðçëßêï åßíáé 0 ôåñìÜôéóá
```




***Εικόνα 11-11.*** *Το παράθυρο άμεσης εκτέλεσης εντολών ενημερώνεται και σε κατάσταση εκτέλεσης*

## Η γραμμή εργαλείων εκσφαλμάτωσης

Η γραμμή εργαλείων εκσφαλμάτωσης προσφέρει γρήγορη προσπέλαση στις πιο συχνά χρησιμοποιούμενες λειτουργίες. Η γραμμή εργαλείων εκσφαλμάτωσης εμφανίζεται από την επιλογή **View ¦ Toolbars ¦ Debug.** Τη γραμμή αυτή την έχουμε σε άμεση χρήση όταν πραγματοποιούμε εκσφαλμάτωση.

**Start** Εκτελεί την εφαρμογή.

**Break** Διακόπτει προσωρινά την εκτέλεση. Για να συνεχίσουμε κάνουμε κλικ στο πλήκτρο Continue (Το πλήκτρο Start με άλλο όνομα).

**End** Τερματίζει την εκτέλεση του προγράμματος. Επιστροφή σε κατάσταση σχεδιασμού.

**Toggle Breakpoint** Εισάγει ή απομακρύνει ένα σημείο διακοπής.

**Step Into** Εκτελεί την αμέσως επόμενη γραμμή κώδικα.

**Step Over** Εκτελεί όλο τον κώδικα μιας διαδικασίας.

**Step Out** Εκτελεί την υπόλοιπη τρέχουσα διαδικασία.

**Locals Window** Εμφανίζει το παράθυρο τοπικών μεταβλητών.

**Immediate Window** Εμφανίζει το παράθυρο άμεσης εκτέλεσης.

**Watch Window** Εμφανίζει το παράθυρο παρακολούθησης.

**Quick Watch** Εμφανίζει το παράθυρο γρήγορου ελέγχου.

**Call Stack** Εμφανίζει λίστα με τις τρέχουσες ενεργές κλήσεις διαδικασιών (διαδικασίες που έχουν καλέσει άλλες και δεν έχουν ολοκληρωθεί

## Ανακεφαλαίωση

Τα είδη των λαθών είναι συντακτικά, εκτέλεσης, λογικά. Τα συντακτικά λάθη οφείλονται στην παραβίαση των κανόνων σύνταξης και γραμματικής της VB. Τα λάθη εκτέλεσης οφείλονται σε λειτουργίες που καλείται να εκτελέσει το υπολογιστικό σύστημα κάτω από συνθήκες που είναι ασαφείς ή/και απαγορευτικές. Τα λογικά λάθη οφείλονται σε λάθος αλγόριθμο ή κωδικοποίηση αλγορίθμου.

΄Ενας τρίτος τρόπος λειτουργίας του περιβάλλοντος εργασίας είναι η λειτουργία σε κατάσταση διακοπής. ΄Οταν το περιβάλλον εργασίας βρίσκεται σε κατάσταση διακοπής μπορούμε να πάρουμε στιγμιότυπα της κατάστασης εκτέλεσης, να επέμβουμε στον κώδικα και να υποβοηθήσουμε ποικιλοτρόπως τη διαδικασία εκσφαλμάτωσης. Σε κατάσταση διακοπής μπορούμε να πραγματοποιήσουμε βηματική εκτέλεση του προγράμματος και να παρακολουθήσουμε τις τιμές μεταβλητών και εκφράσεων στα παράθυρα παρακολούθησης, γρήγορου ελέγχου και τοπικών μεταβλητών. Επίσης, μπορούμε να εκμεταλλευτούμε τις δυνατότητες του παραθύρου άμεσης εκτέλεσης για άμεση εκτέλεση των εντολών και εκτύπωση τιμών.

## Εργαστηριακές Ασκήσεις

1. Πραγματοποιήστε βήμα βήμα το παράδειγμα 11-2. Ελέγξτε τις τιμές των μεταβλητών με τη βοήθεια του δρομέα του ποντικιού.
2. Τοποθετήστε σημείο διακοπής όπως στο παράδειγμα 11-4. Παρουσιάστε τη γραμμή εργαλείων εκσφαλμάτωσης και πραγματοποιήστε βηματική εκτέλεση του προγράμματος.
3. Επαναλάβετε την προηγούμενη άσκηση με ανοιχτό το παράθυρο παρακολούθησης (άσκηση 11-1). Για μια παράσταση καλέστε και το παράθυρο γρήγορου ελέγχου .
4. Πραγματοποιήστε την άσκηση 11-4.
5. Επαναλάβετε την άσκηση 4.
   α) Ενώ βρίσκεστε στο εσωτερικό του βρόχου, ζητήστε να μεταφερθεί ο δείκτης προγράμματος στην εντολή αρχικοποίησης του N πριν το βρόχο και συνεχίστε τη βηματική εκτέλεση. Να παρακολουθείτε τα αποτελέσματα στο παράθυρο άμεσης εκτέλεσης εντολών.



β) Πριν τερματίσει το πρόγραμμα, ζητήστε να εκτελεστεί η εντολή:

```
[Decimal].Text = 5
```

στο παράθυρο άμεσης εκτέλεσης. Επαναφέρετε το δείκτη προγράμματος στην εντολή αρχικοποίησης του N πριν το βρόχο και συνεχίστε τη βηματική εκτέλεση. Παρακολουθήστε τη δραστηριότητα στο παράθυρο άμεσης εκτέλεσης.

Υπόδειξη: Τις αγκύλες τις χρησιμοποιούμε σε ένα όνομα, όταν υπάρχει δεσμευμένη λέξη που συμπίπτει με το όνομα. Αν προσπαθήσετε να δώσετε την εντολή χωρίς τις αγκύλες θα δείτε ότι προκαλείται λάθος εκτέλεσης.

## Σημειώσεις:

## Σημειώσεις:



# Μάθημα 12
# Δημιουργία Μενού

Τελειώνοντας αυτό το μάθημα οι μαθητές θα μπορούν:

- Να δημιουργούν γραμμές μενού, υπομενού και πτυσσόμενα μενού.
- Να αναγνωρίζουν τις ιδιότητες των επιλογών των μενού.
- Να αντιστοιχίζουν στις επιλογές των μενού κώδικα διαχείρισης συμβάντων.

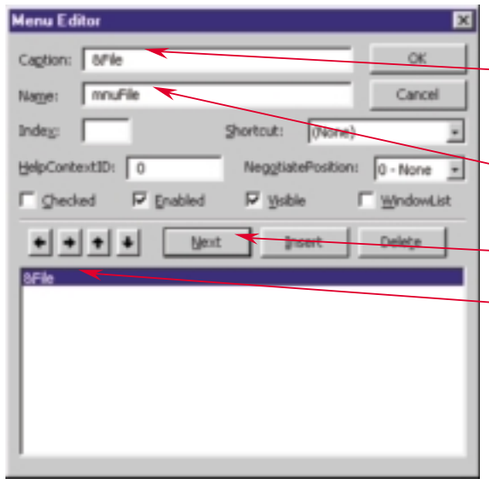
Στα προηγούμενα μαθήματα αναπτύξαμε μικρές εφαρμογές στις οποίες χρησιμοποιήσαμε πλήκτρα διαταγής για να εντολοδοτούμε την έναρξη κάποιων διαδικασιών. Όμως, όταν το πλήθος των διαδικασιών και των διαφορετικών επιλογών είναι πολύ μεγάλο, μια φόρμα με μεγάλο πλήθος πλήκτρων διαταγής και επιλογής, πολλά από τα οποία δεν είναι πάντα σε χρήση, προκαλεί σίγουρα σύγχυση και αποστροφή στο χρήστη. Μια άριστη εναλλακτική λύση δημιουργίας διεπαφής με το χρήστη αποτελούν τα μενού. Με τα μενού οργανώνουμε τις διαταγές σε ομάδες, με τρόπο κοινό και γνώριμο στο χρήστη, ανάλογο εκείνου που χρησιμοποιούν οι περισσότερες εφαρμογές των Windows.

## Επεξεργαστής μενού

Το περιβάλλον εργασίας διαθέτει τον ***επεξεργαστή μενού (menu editor)*** με τον οποίο γίνεται εύκολα η δημιουργία γραμμών μενού και η διαχείριση των μενού, των υπομενού και των επιλογών που τοποθετούνται μέσα σε αυτά. Συγκεκριμένα, με τον επεξεργαστή μενού μπορούμε:

- Να δημιουργήσουμε ***γραμμές μενού (menu bar)***, μενού με διαταγές, ***υπομενού (submenu)*** καθώς και ***αναδυόμενα μενού (pop-up menu)***.
- Να αποδώσουμε στις επιλογές του μενού, που καλούνται και ***στοιχεία του μενού (menu item)***, ιδιότητες όπως όνομα (**`Name`**), περιγραφή (**`Caption`**), δυνατότητα εμφάνισης ή απόκρυψης (**`Visible`**) κ.ά
- να συνδέσουμε ένα μενού με το σύστημα άμεσης βοήθειας (Help)

Σε μια φόρμα επισυνάπτουμε μια γραμμή μενού ακολουθώντας την εξής διαδικασία:




1. Από τη γραμμή μενού του περιβάλλοντος εργασίας επιλέγουμε **Tools ¦ Menu Editor** ή πατάμε το αντίστοιχο πλήκτρο της γραμμής εργαλείων.
2. Στην ιδιότητα **`Caption`** πληκτρολογούμε το περιγραφικό όνομα της πρώτης επιλογής του μενού, όπως θέλουμε να εμφανίζεται κατά την εκτέλεση της εφαρμογής.
3. Στην ιδιότητα **`Name`** πληκτρολογούμε το όνομα με το οποίο θα αναφέρουμε το αντικείμενο μέσω των εντολών του προγράμματος.
4. Κάνουμε κλικ στο πλήκτρο **Next** οπότε το αντικείμενο εμφανίζεται στη ***λίστα επιλογών***.
5. Το παράθυρο του επεξεργαστή μενού επανεμφανίζεται κενό έτοιμο να δεχτεί τον ορισμό των υπόλοιπων επιλογών του μενού. Επαναλαμβάνουμε τα βήματα 2 έως 4 μέχρι να ολοκληρώσουμε το μενού.

***Εικόνα 12-1.*** *Διαδικασία δημιουργίας μενού στο παράθυρο του επεξεργαστή μενού*

Από ότι φαίνεται οι επιλογές μενού είναι αντικείμενα που έχουν ιδιότητες. Οι διαφορά τους από τα άλλα αντικείμενα είναι ότι η αρχική απόδοση τιμών στις ιδιότητές τους δε γίνεται από το παράθυρο ιδιοτήτων αλλά από τον επεξεργαστή μενού για λόγους που θα αναπτύξουμε πιο κάτω.

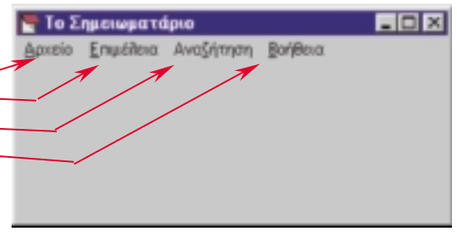
Για να συνδέσουμε μια επιλογή της γραμμής μενού με ένα πλήκτρο άμεσης πρόσβασης, θέτουμε το χαρακτήρα (&) μπρος από το αντίστοιχο γράμμα της ιδιότητας **Caption**. Για παράδειγμα, αν αποδώσουμε την τιμή `&File` στη ιδιότητα **Caption** μιας επιλογής, ορίζουμε ότι το πλήκτρο άμεσης πρόσβασης θα είναι το F. Αυτό σημαίνει ότι όταν εκτελούμε την εφαρμογή μπορούμε να ανοίξουμε απευθείας την επιλογή **File** πατώντας το συνδυασμό πλήκτρων `Alt + F`.

Η λειτουργία των πλήκτρων άμεσης πρόσβασης έχει εξηγηθεί στο μάθημα 4.

**Παράδειγμα 12-1.**

Η γραμμή μενού της φόρμας μιας εφαρμογής, η οποία είναι παρόμοια με το ***Σημειωματάριο*** (***Notepad***) των Windows, θέλουμε να περιέχει τις επιλογές Αρχείο, Επιμέλεια, Αναζήτηση, Βοήθεια. Αφού δημιουργήσουμε τη φόρμα, καλούμε τον επεξεργαστή μενού και εισάγουμε τις επιλογές μενού που προδιαγράψαμε.

| Caption | Name |
|---|---|
| &Áñ÷åßï | mnuFile |
| &ÅðéìÝëåéá | mnuEdit |
| Áíá&æÞôçóç | mnuSearch |
| &ÂïÞèåéá | mnuHelp |




***Εικόνα 12-2.*** *Η φόρμα και οι ιδιότητες των επιλογών της γραμμής μενού*

Ας σημειωθεί ότι τα πλήκτρα άμεσης πρόσβασης φαίνεται ότι δε λειτουργούν, και αυτό γιατί δεν έχουμε δημιουργήσει μενού κάτω από κάθε επιλογή.

Η διαδικασία που περιγράψαμε πιο πάνω είναι η βασική διαδικασία η οποία παράγει ένα μενού με επιλογές, που κατά την εκτέλεση θα εμφανιστούν σε μια γραμμή και για τις οποίες έχουμε ορίσει τις βασικές ιδιότητες **Caption** και **Name**. Αν θέλουμε να δημιουργήσουμε επίπεδα επιλογών (μενού και υπομενού), να απενεργοποιήσουμε ή/και να αποκρύψουμε κάποιες επιλογές μπορούμε να το κάνουμε κατά το χρόνο δημιουργίας του μενού ή και εκ των υστέρων αξιοποιώντας τα υπόλοιπα πλαίσια και πλήκτρα που διαθέτει ο επεξεργαστής μενού. Συγκεκριμένα, για να:

- **δημιουργήσουμε επίπεδα επιλογών**

  Στη λίστα επιλογών του επεξεργαστή μενού υποδεικνύουμε την επιλογή που μας ενδιαφέρει και πατάμε ένα από τα πλήκτρα με τα βελάκια, που χαρακτηρίζονται ως ***σκιαγράφησης επιπέδων (outlining buttons).***

  μεταφέρει την επιλογή προς τα δεξιά και εμφανίζει τελείες μπροστά από το όνομά της, που σημαίνει ότι η επιλογή κατέβηκε ένα επίπεδο.

  μεταφέρει την επιλογή προς τα αριστερά που σημαίνει ότι η επιλογή ανέβηκε ένα επίπεδο.

  μεταφέρει την επιλογή μια θέση επάνω (αλλάζει με την προηγούμενή της).

  μεταφέρει την επιλογή μια θέση κάτω (αλλάζει με την επόμενή της).

  `Insert` εισάγει μια επιλογή πάνω από αυτήν που υποδεικνύουμε στη λίστα επιλογών.

  `Delete` διαγράφει μια επιλογή που υποδεικνύουμε στη λίστα επιλογών.

- **αντιστοιχίσουμε πλήκτρα συντομίας (shortcut keys) στις επιλογές**

  Στη λίστα επιλογών του επεξεργαστή μενού υποδεικνύουμε την επιλογή που μας ενδιαφέρει και από το σύνθετο πλαίσιο λίστας Shortcut επιλέγουμε τον επιθυμητό συνδυασμό.

- **αποδώσουμε ιδιότητες στις επιλογές**

  **`Enabled`** ενεργοποιεί/απενεργοποιεί την επιλογή μέσα στο μενού.

  **`Visible`** εμφανίζει/κρύβει την επιλογή μέσα στο μενού.

  **`Checked`** εμφανίζει το σημείο μαρκαρίσματος δίπλα στην επιλογή.

  Στις παραπάνω ιδιότητες μπορούμε να αποδώσουμε τιμές όχι μόνο από τον επεξεργαστή μενού αλλά και μέσα από τον κώδικα που γράφουμε για τη διαδικασία διαχείρισης συμβάντων **`Click`** των επιλογών του μενού.

Μόνο μια γραμμή μενού μπορεί να υπάρχει ανά φόρμα. Στην ειδική περίπτωση που θέλουμε να χρησιμοποιεί ο χρήστης περισσότερες από μια, κάνουμε ορατές ή αόρατες τις κατάλληλες βασικές επιλογές.



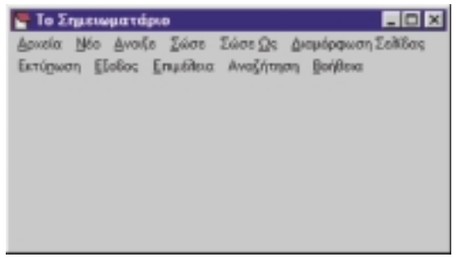
**Παράδειγμα 12-2.**

Συνεχίζουμε το προηγούμενο παράδειγμα προκειμένου να δείξουμε επιπλέον δυνατότητες του επεξεργαστή μενού. Δημιουργούμε τις υποεπιλογές της επιλογής **Αρχείο**, ακολουθώντας την εξής διαδικασία:

1. Στο πλαίσιο λίστας αντικειμένων υποδεικνύουμε την επιλογή &ÅðéìÝëåéá και πατάμε το πλήκτρο **`Insert`**. Αμέσως ανοίγει χώρος για να υποδεχτεί την πρώτη υποεπιλογή.
2. Σύμφωνα με τα προηγούμενα συμπληρώνουμε το πλαίσιο **`Caption`** και **`Name`** για κάθε υποεπιλογή και συγκεκριμένα:

| Caption | Name |
|---|---|
| &Áñ÷åßï | mnuFile |
| &ÍÝï | mnuFileNew |
| &Áíïéîå | mnuFIleOpen |
| &Óþóå | mnuFileSave |
| Óþóå &Ùò | mnuFileSaveAs |
| &Áéáìüñöùóç Óåëßäáò | mnuFileSetUp |
| Âêôý&ðùóç | mnuFilePrint |
| &Åîïäïò | mnuFileExit |
| &ÅðéìÝëåéá | mnuEdit |
| Áíá&æÞôçóç | mnuSearch |
| &ÂïÞèåéá | mnuHelp |

Αν ζητήσουμε εκτέλεση της εφαρμογής παρατηρούμε ότι δε δημιουργήθηκε το μενού της επιλογής **Αρχείο** αλλά προστέθηκαν όλες οι επιλογές στη γραμμή μενού.

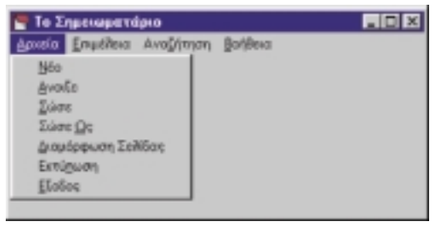



***Εικόνα 12-3****. Αν οι επιλογές της γραμμής μενού δε χωρούν στοπλάτος του παραθύρου επεκτείνονται και σε δεύτερη γραμμή*

3. Για να εμφανίσουμε τις νέες επιλογές ως μενού της επιλογής **Αρχείο**, τις υποδεικνύουμε μια προς μια στη λίστα αντικειμένων του επεξεργαστή μενού και πατάμε το πλήκτρο βέλος δεξιά για να κατέβουν ένα επίπεδο. Στη λίστα επιλογών του επεξεργαστή μενού έχουμε:

```
&Áñ÷åßï
...&ÍÝï
...&Áíïéîå
...&Óþóå
...Óþóå &Ùò
...&Äéáìüñöùóç Óåëßäáò
...Âêôý&ðùóç
...&Åîïäïò
&ÅðéìÝëåéá
```

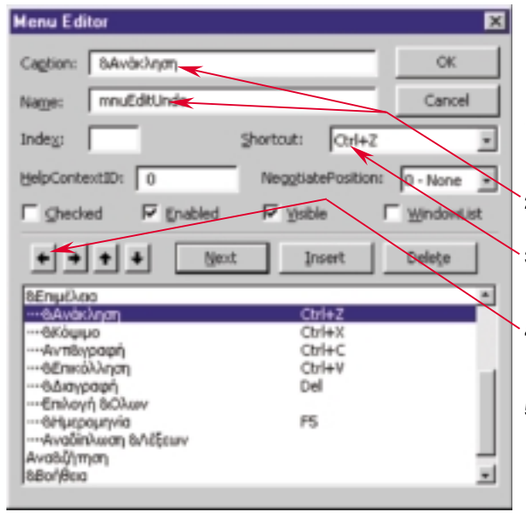
Τότε η γραμμή μενού αποκτά πάλι τέσσερις επιλογές και επιλέγοντας **Αρχείο** με το ποντίκι ή με το συνδυασμό των πλήκτρων `Alt + Á` το μενού με τις υποεπιλογές εμφανίζεται στο χώρο του παραθύρου.




***Εικόνα 12-4.*** *Το μενού* ***Αρχείο***

**Παράδειγμα 12-3.**

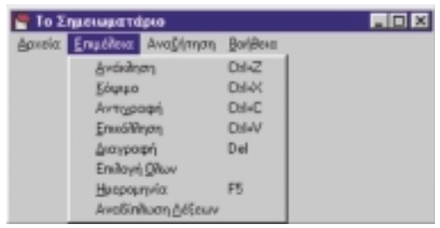
Δημιουργούμε το μενού της επιλογής **Επιμέλεια** και αντιστοιχίζουμε πλήκτρα συντομίας στις υποεπιλογές του.




1. Με το πλήκτρο **Insert** δημιουργούμε χώρο πάνω από την επιλογή αναζήτηση.
2. Δηλώνουμε τις ιδιότητες **`Caption`** και **`Name`**.
3. Επιλέγουμε τον επιθυμητό συνδυασμό πλήκτρων συντόμευσης.
4. Ενθέτουμετην επιλογή σε δεύτερο επίπεδο στην ιεραρχία των επιλογών.
5. Επαναλαμβάνουμε τα βήματα 1 έως 4 για όλες τις επιλογές.

***Εικόνα 12-5***. *Δηλώσεις πλήκτρων συντόμευσης*

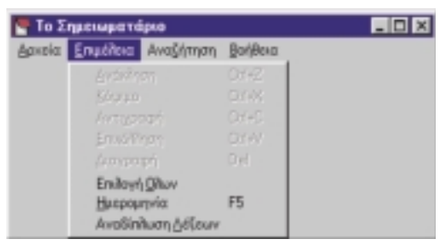
Κατά την εκτέλεση της εφαρμογής, αν καλέσουμε το μενού **Επιμέλεια** ο συνδυασμός των πλήκτρων συντόμευσης εμφανίζεται δεξιά από το όνομα κάθε επιλογής. Με τη χρήση των πλήκτρων συντομίας μπορούμε να έχουμε άμεση πρόσβαση σε μια υποεπιλογή χωρίς να ανοίξουμε το μενού που την περιέχει.




***Εικόνα 12-6***. *Το μενού Επιμέλεια*

**Παράδειγμα 12-4.**

Αν θέλουμε κάποιες επιλογές να είναι αρχικά απενεργοποιημένες (όπως πράγματι συμβαίνει στην εφαρμογή Notepad) δεν έχουμε παρά να τις επιλέξουμε στη λίστα αντικειμένων του επεξεργαστή μενού και να ξεμαρκάρουμε την ιδιότητα τους **`Enabled`**. Συνήθως την ιδιότητα **`Enabled`**, όπως και τη **`Visible`**, τις χειριζόμαστε μέσω του κώδικα, όπου κάτω από συνθήκες τους αποδίδουμε αληθή ή ψευδή τιμή.



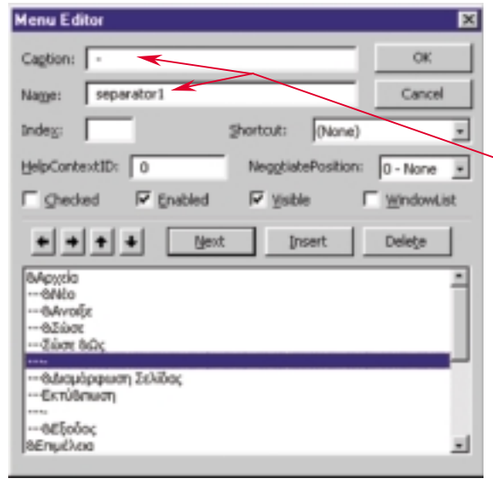

***Εικόνα 12-7***. *Το μενού Επιμέλεια με απενεργοποιημένες επιλογές*

Για να χωρίσουμε σε ομάδες τις υποεπιλογές χρησιμοποιούμε ***οριζόντιες διαχωριστικές γραμμές (separator bars).*** Οι γραμμές αυτές αποτελούν επίσης αντικείμενα, τα οποία όμως δεν επιλέγονται κατά την εκτέλεση της εφαρμογής.



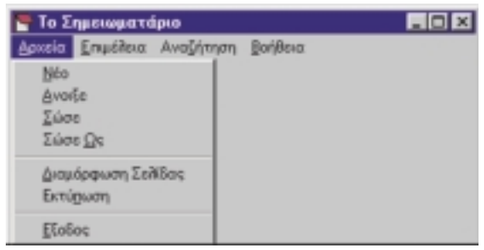
**Παράδειγμα 12-5.**

Για να παρεμβάλλουμε διαχωριστική γραμμή ανάμεσα στην επιλογή **Σώσε Ως** και **Διαμόρφωση Σελίδας** ακολουθούμε την εξής διαδικασία:




1. Επιλέγουμε στη λίστα επιλογών τη &Äéáìüñöùóç Óåëßäáò και πατάμε το πλήκτρο **Insert.**
2. Για την καινούργια επιλογή πληκτρολογούμε στην ιδιότητα **`Caption`** μια παύλα (-) και στην ιδιότητα **`Name`** εισάγουμε υποχρεωτικά ένα όνομα (π.χ `separator1`), το οποίο όμως δεν έχει καμιά σημασία για την εφαρμογή (δεν το χρησιμοποιούμε σε κανένα σημείο του κώδικα).
3. Με τον ίδιο τρόπο παρεμβάλλουμε μια διαχωριστική γραμμή ανάμεσα στην επιλογή **Εκτύπωση** και **Έξοδος**.

***Εικόνα 12-8.*** *Δημιουργία ομάδων επιλογών*




***Εικόνα 12-9***. *Η τελική μορφή του μενού Αρχείο*

## Επισύναψη κώδικα σε επιλογή μενού

Για να εκτελέσουμε τις επιθυμητές εργασίες μέσω των επιλογών ενός μενού πρέπει να δημιουργήσουμε τον αντίστοιχο κώδικα. Τα αντικείμενα των μενού υποστηρίζουν ένα μόνο συμβάν, το συμβάν **`Click`**, και επομένως στην αντίστοιχη υπορουτίνα συμβάντος γράφουμε τον κατάλληλο κώδικα. Για να εμφανίσουμε στο παράθυρο κώδικα την υπορουτίνα διαχείρισης συμβάντος **`Click`** μιας επιλογής, κλείνουμε τον επεξεργαστή μενού και με το ποντίκι επιλέγουμε την επιλογή από τη λίστα αντικειμένων που μας ενδιαφέρει.

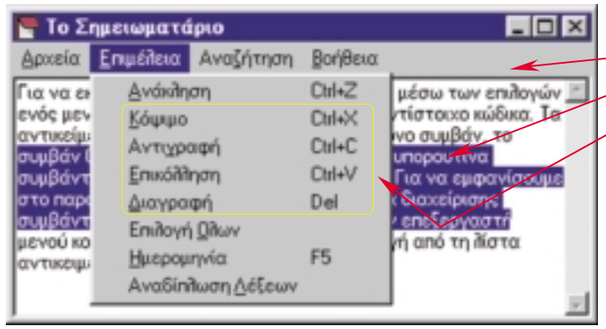
**Άσκηση 12-1.**

Θα γράψουμε τον κώδικα που θα αποκόβει, θα αντιγράφει, θα επικολλά και θα διαγράφει ένα τμήμα του κειμένου, που έχουμε γράψει μέσα σε ένα πλαίσιο κειμένου.

Στη φόρμα του Σημειωματαρίου τοποθετούμε ένα πλαίσιο κειμένου με όνομα `TextArea`. Στο πλαίσιο κειμένου θα γράφουμε το προς επεξεργασία κείμενο, που κατά κανόνα θα περιέχει πολλές γραμμές. Γι' αυτό δίνουμε την τιμή `True` στην ιδιότητα **`MultiLine`** (πλαίσιο κειμένου πολλών γραμμών) και την τιμή **2-Vertical** στην ιδιότητα **`ScrollBars`** (εμφάνιση κατακόρυφης ράβδου κύλισης).

Η επιλογή κειμένου γίνεται όπως ακριβώς σε όλους τους επεξεργαστές κειμένου. Κάνουμε κλικ στην αρχή της περιοχής που θέλουμε να επιλέξουμε και σύουμε το ποντίκι μέχρι το τέλος της. Όταν έχει γίνει επιλογή μιας περιοχής κειμένου, η τιμή της ιδιότητας **`SelText`** του πλαισίου κειμένου ισούται με τη συμβολοσειρά που έχει επιλεγεί.

Η αποκοπή, η αντιγραφή και η επικόλληση κειμένου θα γίνεται με τη βοήθεια του ***προχείρου (Cliboard)***. Όπως είδαμε στο μάθημα 4 υπάρχει ένα αντικείμενο με όνομα `Clipboard`. Το αντικείμενο αυτό υποστηρίζεται από τη μέθοδο **`Clear`**, με την οποία διαγράφουμε το περιεχόμενό του, τη μέθοδο **`SetText`**, με την οποία συμπληρώνουμε το περιεχόμενό του και από τη μέθοδο **`GetText`**, με την οποία αντλούμε την τιμή του περιεχομένου του.




***Εικόνα 12-10****. Επιλέγουμε Επιμέλεια για επεξεργασία της επιλεγμένης περιοχής.*

Οι διαδικασίες εξυπηρέτησης συμβάντων **`Click`** για τις επιλογές **Επιμέλεια**, **Κόψιμο**, **Αντιγραφή**, **Επικόλληση**, **Διαγραφή** είναι:

```
Private Sub mnuEdit_Click()
   'Áíåñãïðïßçóç/Áðåíåñãïðïßçóç åðéëïãþí
   'ÁíÜêëçóç
    ...
   'Êüøéìï, ÁíôéãñáöÞ, Äéáãñáöή (Ýëåãîå áí õðÜñ÷åé åðéëåãìÝíï êåßìåíï)
    If Len(TextArea.SelText) > 0 Then
        mnuEditCut.Enabled = True
        mnuEditCopy.Enabled = True
        mnuEditDelete.Enabled = True
    Else
        mnuEditCut.Enabled = False
        mnuEditCopy.Enabled = False
        mnuEditDelete.Enabled = False
    End If
   'Âðéêüëëçóç (¸ëãîå áí õðÜñ÷åé êåßìåíï óôï Ðñü÷åéñï)
    mnuEditPaste.Enabled = (Clipboard.GetText() <> "")
End Sub

Private Sub mnuEditCut_Click()
   'ÄéáãñáöÞ ôïõ ðåñéå÷ïìÝíïõ ôïõ Ðñï÷åßñïõ
    Clipboard.Clear
   'ÁíôéãñáöÞ åðéëåãìÝíïõ êåéìÝíïõ óôï Ðñü÷åéñï
    Clipboard.SetText TextArea.SelText
   'ÄéáãñáöÞ åðéëåãìÝíïõ êåéìÝíïõ
    TextArea.SelText = ""
End Sub

Private Sub mnuEditCopy_Click()
   'ÄéáãñáöÞ ôïõ ðåñéå÷ïìÝíïõ ôïõ Ðñï÷åßñïõ
    Clipboard.Clear
   'ÁíôéãñáöÞ åðéëåãìÝíïõ êåéìÝíïõ óôï Ðñü÷åéñï
    Clipboard.SetText TextArea.SelText
End Sub

Private Sub mnuEditPaste_Click()
   'ÔïðïèÝôçóç êåéìÝíïõ áðü ôï Ðñü÷åéñï óôï óçìåßï ôïõ êåéìÝíïõ ðïõ åðéëÝ÷èçêå
    TextArea.SelText = Clipboard.GetText()
End Sub
```



```
Private Sub mnuEditDelete_Click()
   'ÄéáãñáöÞ åðéëåãìÝíïõ êåéìÝíïõ
    TextArea.SelText = ""
End Sub
```

## Αναδυόμενα μενού

Τα αναδυόμενα μενού (Pop-Up Menu) είναι μενού επιλογών που εμφανίζονται σε μια φόρμα, όταν ο χρήστης πατάει το δεξί πλήκτρο του ποντικιού. Τα αναδυόμενα μενού εμφανίζονται στο σημείο που βρίσκεται ο δείκτης του ποντικιού τη συγκεκριμένη στιγμή. Τα χρησιμοποιούμε ως μενού συντομίας για να παρουσιάσουμε τις πιο χρήσιμες διαταγές για το αντικείμενο που δείχνουμε. Τα αναδυόμενα μενού δημιουργούνται με τη βοήθεια του επεξεργαστή μενού, όπως περιγράψαμε παραπάνω. Μπορούμε να δημιουργήσουμε πολλά αναδυόμενα μενού, αλλά κάθε φορά μπορούμε να εμφανίζουμε μόνο ένα. Κάθε μενού επιλογών της γραμμής μενού μιας φόρμας μπορεί να εμφανιστεί κατά την εκτέλεση με τη μορφή αναδυόμενου μενού. Ωστόσο, μπορούμε να δημιουργήσουμε αναδυόμενα μενού που δεν είναι ορατά μέσω της γραμμής μενού αρκεί να δώσουμε στην ιδιότητα **`Visible`** την τιμή `False`.

Για να εμφανίσουμε ένα αναδυόμενο μενού χρησιμοποιούμε τη μέθοδο **`PopupMenu`**, της οποίας μια μορφή είναι η:

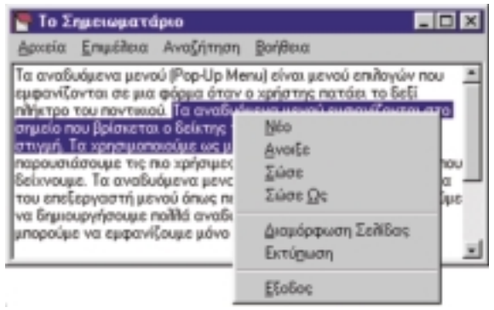
**`PopupMenu`** *όνομα_μενού*

Τυπικά, καλούμε τη μέθοδο **`PopupMenu`** στη διαδικασία διαχείρισης συμβάντος **`MouseUp`** για το αντικείμενο που θα εξυπηρετηθεί από το μενού. Όπως είναι φανερό μπορούμε να εμφανίσουμε ένα αναδυόμενο μενού για κάθε αντικείμενο που υποστηρίζει το συμβάν **`MouseUp`**.

Το μενού που θέλουμε να ορίσουμε ως αναδυόμενο πρέπει να έχει τουλάχιστον ένα υπομενού!

**Παράδειγμα 12-6.**

Ορίζουμε ως αναδυόμενο μενού, το μενού `mnuEdit`, που δημιουργήσαμε σε προηγούμενο παράδειγμα. Η εμφάνισή του θα γίνεται κάθε φορά που πατάμε το δεξί πλήκτρο του ποντικιού επάνω στο πλαίσιο κειμένου `TextArea` και στο σημείο που υποδεικνύει ο δείκτης του ποντικιού.




***Εικόνα 12-11.*** *Το αναδυόμενο μενού είναι ένα μενού της γραμμής μενού.*

Γι' αυτό συντάσσουμε τη διαδικασία διαχείρισης συμβάντος **`MouseUp`** για το πλαίσιο κειμένου και από την τιμή της παραμέτρου `Button` ελέγχουμε αν πατήθηκε το δεξί πλήκτρο του ποντικιού (`vbRightButton`).

```
Private Sub Form_MouseUp(Button As Integer, Shift As Integer, _
                         X As Single, Y As Single)
If Button = vbRightButton Then  'ÐáôÞèçêå äåîß ðëÞêôñï;
    PopupMenu mnuEdit
    End If
End Sub
```

Για να ορίσουμε ως αναδυόμενο μενού το μενού `mnuFile`, δεν έχουμε παρά να αντικαταστήσουμε στον παραπάνω κώδικα την `PopupMenu mnuEdit` με την `PopupMenu mnuFile`. Τέλος, σημειώνουμε ότι η `PopupMenu mnuHelp` θα δημιουργήσει λάθος εκτέλεσης γιατί για την επιλογή **Βοήθεια** δεν έχουμε ορίσει μενού.

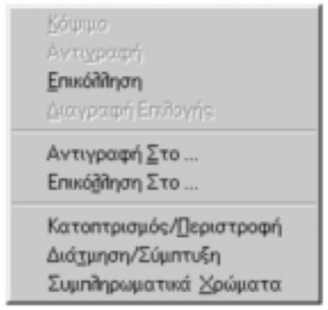
Στην περίπτωση που θέλουμε να δημιουργήσουμε ένα αναδυόμενο μενού, το οποίο να μη φαίνεται στη γραμμή μενού δεν έχουμε παρά να το χαρακτηρίσουμε ως μη ορατό.

## Ανακεφαλαίωση

Το περιβάλλον εργασίας διαθέτει τον επεξεργαστή μενού με τον οποίο γίνεται εύκολα η δημιουργία γραμμών μενού και η διαχείριση των μενού, των υπομενού και των επιλογών που τοποθετούνται μέσα σε αυτά. Με τον επεξεργαστή μενού μπορούμε να δημιουργήσουμε γραμμές μενού, μενού με διαταγές, υπομενού και αναδυόμενα μενού. Επίσης, μπορούμε να αποδώσουμε στις επιλογές του μενού τις ιδιότητες τους όπως το όνομα, την περιγραφή, τη δυνατότητα εμφάνισης ή απόκρυψης κ.ά.
Για να εκτελέσουμε τις επιθυμητές εργασίες μέσω των επιλογών ενός μενού πρέπει να δημιουργήσουμε τον αντίστοιχο κώδικα. Τα αντικείμενα των μενού υποστηρίζουν ένα μόνο συμβάν, το συμβάν **`Click`**, και επομένως στην αντίστοιχη υπορουτίνα συμβάντος γράφουμε τον κατάλληλο κώδικα.

## Εργαστηριακές Ασκήσεις

1. Πραγματοποιήστε ό,τι περιγράφεται στο παράδειγμα 12-1 για τη δημιουργία της γραμμής μενού.
2. Ακολουθήστε βήμα βήμα όσα περιγράφονται στο παράδειγμα 12-2 για τη δημιουργία του μενού **Αρχείο**.
3. Ακολουθήστε βήμα βήμα όσα περιγράφονται στο παράδειγμα 12-3 για τη δημιουργία του μενού **Επιμέλεια**.
4. Δημιουργήστε μενού για την επιλογή **Αναζήτηση** και **Βοήθεια**. Βελτιώστε όλες τις επιλογές της γραμμής μενού, ώστε να διαχωρίζονται οι ομάδες επιλογών τους με διαχωριστικές γραμμές και να είναι απενεργοποιημένες οι επιλογές που δεν είναι δυνατόν να λειτουργήσουν εξαρχής.
   Υπόδειξη: Συμβουλευτείτε και το NotePad των Windows.
5. Πραγματοποιήστε την άσκηση 12-1 για να επισυνάψετε κώδικα στις επιλογές **Επιμέλεια, Κόψιμο, Αντιγραφή, Επικόλληση, Διαγραφή**.
6. Πραγματοποιήστε ότι περιγράφεται στο παράδειγμα 12-6 για τη δημιουργία αναδυόμενου μενού.
7. Δημιουργήστε ένα αναδυόμενο μενού με τις βασικές επιλογές της εφαρμογής, το οποίο να μην εμφανίζει αντίστοιχη επιλογή στη γραμμή τίτλου.
8. Δημιουργήστε φόρμα, γραμμή μενού, μενού και υπομενού για εφαρμογή παρόμοια με το πρόγραμμα ***Ζωγραφική (Paint)*** των Windows.
9. Για το πρόγραμμα Ζωγραφική δημιουργήστε αναδυόμενο μενού της μορφής:




10. Στο μενού εφαρμογής του Σημειωματαρίου που δημιουργήσατε, να εισάγετε μια επιλογή με την οποία να είναι δυνατή η αλλαγή της ιδιότητας Caption όλων των επιλογών του μενού, ώστε η περιγραφή να γίνεται είτε στα ελληνικά είτε στα αγγλικά.



# Μάθημα 13
# Διαλογικά παράθυρα
# Εφαρμογές με πολλά παράθυρα

Τελειώνοντας αυτό το μάθημα οι μαθητές θα μπορούν:

- Να δημιουργούν απλά διαλογικά παράθυρα.
- Να χρησιμοποιούν τα προτυποποιημένα διαλογικά παράθυρα τύπου Open, Print κ.ά.
- Να δημιουργούν εφαρμογές με πολλές φόρμες.

Πολύ συχνά κατά την εκτέλεση μιας εφαρμογής ο χρήστης χρειάζεται να ενημερωθεί με στοιχεία από την εφαρμογή ή/και να δώσει στοιχεία στην εφαρμογή. Σ' αυτόν το διάλογο, βασικό ρόλο παίζουν τα ειδικής μορφής παράθυρα, που είναι γνωστά ως ***διαλογικά παράθυρα (dialog boxes)***. Η VB δίνει τη δυνατότητα να δημιουργήσουμε διαλογικά παράθυρα από πολύ απλά ως ιδιαίτερα σύνθετα. Συγκεκριμένα μπορούμε:

- Να καλέσουμε στον κώδικά του προγράμματος τις διαλογικές συναρτήσεις `MsgBox` και `InputBox`.
- ·Να χρησιμοποιήσουμε το μη ορατό αντικείμενο CommonDialog για να παρουσιάσουμε προτυποποιημένα κοινά διαλογικά παράθυρα σαν αυτά που παρουσιάζουν οι επιλογές **Open, Print** κ.ά. των εφαρμογών Office της Microsoft.
- Να δημιουργήσουμε εξαρχής διαλογικά παράθυρα βασισμένα σε φόρμες που να ικανοποιούν τις ιδιαίτερες ανάγκες μας.

Η επικοινωνία χρήστη - εφαρμογής δε βασίζεται μόνο σε μια φόρμα και διαλογικά παράθυρα. Σε σύνθετες εφαρμογές μπορεί να βελτιωθεί σημαντικά χρησιμοποιώντας πολλές φόρμες και πολλά παράθυρα. Η εμφάνιση και η απόκρυψη των παραθύρων γίνεται, όπως θα δούμε, με κατάλληλες εντολές και μεθόδους.

## Συναρτήσεις εισόδου και εξόδου
## Η συνάρτηση `MsgBox`

Πρόκειται για μια κάπως περίεργη συνάρτηση, η οποία δεν επιστρέφει μόνο κάποια τιμή αλλά και εμφανίζει ένα διαλογικό παράθυρο (Message box) στην οθόνη. Τα διαλογικό παράθυρο περιέχει ένα μήνυμα προς το χρήστη και ένα ή περισσότερα πλήκτρα διαταγής, με τα οποία ο χρήστης δηλώνει την απάντησή του. Η τιμή που επιστρέφει η συνάρτηση είναι ένας ακέραιος αριθμός και καθορίζεται από το πλήκτρο που πατάει ο χρήστης για να κλείσει το διαλογικό παράθυρο. Η γενική μορφή της συνάρτησης είναι:

`MsgBox` (*μήνυμα, μορφή, τίτλος*)

Η τιμή της παραμέτρου *μήνυμα* αποτελεί το μήνυμα προς τον χρήστη, το οποίο θα εμφανιστεί στο εσωτερικό του διαλογικού παραθύρου. Η παράμετρος *μορφή* καθορίζει τον αριθμό και τον τύπο των πλήκτρων διαταγής που θα τοποθετηθούν στο διαλογικό παράθυρο, το εικονίδιο που πιθανόν θα περιέχεται σε αυτό και τη συμπεριφορά του παραθύρου. Τέλος, η παράμετρος *τίτλος* καθορίζει τον τίτλο του παραθύρου.

Από τις τρεις παραμέτρους μόνον η παράμετρος *μήνυμα* είναι υποχρεωτική. Οι άλλες δύο επιτρέπεται να παραληφθούν. Στην περίπτωση που παραληφθεί η δεύτερη παράμετρος, η θέση της πρέπει να δεσμευτεί με ένα κόμμα. Αν παραλείψουμε την παράμετρο *μορφή*, η συνάρτηση συμπεριφέρεται σαν να δόθηκε *μορφή* με τιμή 0 και εμφανίζει στο εσωτερικό του παραθύρου ένα μόνο πλήκτρο διαταγής. Αν παραλείψουμε τον τίτλο, ως τίτλος του παραθύρου τοποθετείται το όνομα του έργου της εφαρμογής.

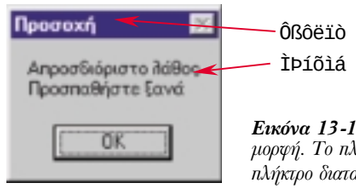
Το μέγεθος της τιμής της παραμέτρου *μήνυμα* μπορεί να είναι μέχρι 1024 χαρακτήρες περίπου. Το μήνυμα μπορεί να αποτελείται από περισσότερες της μιας γραμμή. Για να καθορίσουμε τα σημεία στα οποία θα γίνεται η αλλαγή γραμμής, μεταξύ των συμβολοσειρών που αποτελούν το περιεχόμενο κάθε γραμμής πρέπει να παρεμβάλλουμε τους χαρακτήρες που αντιστοιχούν στην εσωτερική σταθερά `vbCrLf`.

**Παράδειγμα 13-1.**

Η εντολή:

```
lReturn = MsgBox("Áðñïóäéüñéóôï ëÜèïò" & vbCrLf & _
                 "ÐñïóðáèÞóôå îáíÜ", , "Ðñïóï÷Þ")
```

εμφανίζει στην οθόνη διαλογικό παράθυρο με μήνυμα δύο γραμμών και με τίτλο: "Ðñïóï÷Þ".




***Εικόνα 13-1.*** *MsgBox με τίτλο και μη προσδιορισμένη μορφή. Το πλαίσιο διαλόγου παρουσιάζεται λιτό με ένα μόνο πλήκτρο διαταγής, που το πάτημά του κλείνει το παράθυρο.*

Η τιμή που δέχεται η παράμετρος μορφή είναι αριθμητική και προκύπτει από το άθροισμα των επιθυμητών τιμών σύμφωνα με τις σταθερές και τις λειτουργίες που περιγράφονται στον πίνακα 13-1.

| Σταθερά | Τιμή | Λειτουργία |
|---|---|---|
| vbOKOnly | 0 | Εμφάνισε μόνο το πλήκτρο Ok |
| vbOKCancel | 1 | Εμφάνισε τα πλήκτρα Ok και Cancel |
| vbAbortRetryIgnore | 2 | Εμφάνισε τα πλήκτρα Abort, Retry και Ignore |
| vbYesNoCancel | 3 | Εμφάνισε τα πλήκτρα Yes, No και Cancel |
| vbYesNo | 4 | Εμφάνισε τα πλήκτρα Yes και No |
| VbRetryCancel | 5 | Εμφάνισε τα πλήκτρα Retry êáé Cancel |
| VbCritical | 16 | Εμφάνισε το εικονίδιο |
| vbQuestion | 32 | Εμφάνισε το εικονίδιο |
| vbExclamation | 48 | Εμφάνισε το εικονίδιο |
| VbInformation | 64 | Εμφάνισε το εικονίδιο |
| vbDefaultButton1 | 0 | Το πρώτο πλήκτρο είναι προεπιλεγμένο |
| vbDefaultButton2 | 256 | Το δεύτερο πλήκτρο είναι προεπιλεγμένο |
| vbDefaultButton3 | 512 | Το τρίτο πλήκτρο είναι προεπιλεγμένο |
| vbDefaultButton4 | 768 | Το τέταρτο πλήκτρο είναι προεπιλεγμένο |
| vbApplicationModal | 0 | Υποχρεωτική απόκριση σε επίπεδο εφαρμογής |
| vbSystemModal | 4096 | Υποχρεωτική απόκριση σε επίπεδο συστήματος |

Οι τιμές των σταθερών δεν είναι συνεχόμενες και φαίνεται σαν να έχουν επιλεγεί αυθαίρετα. Παρατηρήστε όμως ότι κάποιες από αυτές αποτελούν δυνάμεις του 2. Αυτό σημαίνει ότι έχει γίνει κάποια αντιστοίχιση λειτουργιών με τα bits ενός ακέραιου αριθμού.

***Πίνακας 13-1****. Σταθερές των οποίων οι τιμές καθορίζουν τη μορφή του διαλογικού παραθύρου που εμφανίζει η συνάρτηση* MsgBox.

Παρατηρώντας τον πίνακα διακρίνουμε τέσσερις ομάδες τιμών:

- Η πρώτη ομάδα, με τιμές (0 - 5), καθορίζει τον αριθμό και τον τύπο των πλήκτρων που θα περιέχει το διαλογικό παράθυρο.
- Η δεύτερη ομάδα, με τιμές (16, 32, 48, 64), καθορίζει το εικονίδιο που θα περιέχει το διαλογικό παράθυρο.
- Η τρίτη ομάδα, με τιμές (0, 256, 512, 768), καθορίζει ποιο πλήκτρο θα είναι προεπιλεγμένο.
- Η τέταρτη ομάδα, με τιμές (0, 4096), καθορίζει αν η απόκριση του χρήστη θα είναι υποχρεωτική σε επίπεδο εφαρμογής ή σε επίπεδο συστήματος.

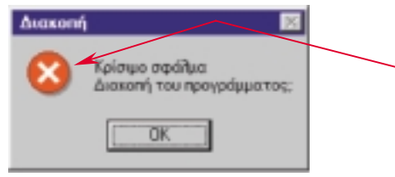
Σε ότι αφορά την τέταρτη ομάδα έχουμε να προσθέσουμε ότι ένα διαλογικό παράθυρο είναι πάντα ***υποχρεωτικής απόκρισης (modal).*** Αυτό σημαίνει ότι από τη στιγμή που εμφανίζεται, διακόπτεται η εκτέλεση της εφαρμογής και απαιτείται οπωσδήποτε απόκριση από τον χρήστη. Ο χρήστης δεν έχει τη δυνατότητα να επιλέξει άλλο παράθυρο της τρέχουσας εφαρμογής και σε αρκετές περιπτώσεις (που θεωρούνται κρίσιμες) δεν μπορεί να επιλέξει ούτε παράθυρο άλλης εφαρμογής. Είναι υποχρεωμένος να πατήσει ένα πλήκτρο του διαλογικού παραθύρου

Τα παράθυρα των Windows που δεν είναι διαλογικά δεν επιβάλλουν υποχρεωτική απόκριση του χρήστη και γι' αυτό χαρακτηρίζονται ως ***modeless***.



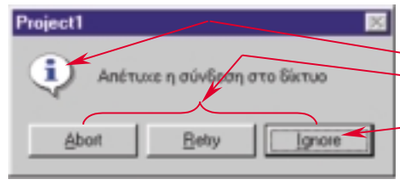
για να το κλείσει και να ολοκληρωθεί η επεξεργασία που το είχε εμφανίσει. Μετά μπορεί να συνεχίσει τη διαχείριση των άλλων παραθύρων. Όταν ένα διαλογικό παράθυρο εμποδίζει το χρήστη να επιλέξει παράθυρα άλλης εφαρμογής χαρακτηρίζεται ως **υποχρεωτικής απόκρισης σε επίπεδο συστήματος**, διαφορετικά χαρακτηρίζεται ως **υποχρεωτικής απόκρισης σε επίπεδο εφαρμογής.** Η τιμή της παραμέτρου μορφή μπορεί να προκύψει ως άθροισμα σταθερών που ανήκουν σε διαφορετικές ομάδες του πίνακα. Σε αυτήν την περίπτωση το παράθυρο έχει τα επιμέρους χαρακτηριστικά που του αποδίδει η τιμή της κάθε ομάδας.

**Παράδειγμα 13-2.**




***Εικόνα 13-2.***

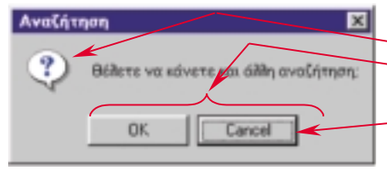
```
MsgBox("Êñßóéìï óöÜëìá" & vbCrLf &-
       "Äéáêïðþ ôïõ
       Vðñïãñáììáôïò;", _
       vbCritical, _
       "Äéáêïðþ")
```




***Εικόνα 13-3.***

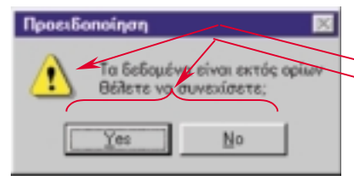
```
MsgBox("ÁðÝôõ÷å ç óýíäåóç óôï äßêôõï", _
       vbInformation + _
       vbAbortRetryIgnore + _
       vbDefaultButton3)
```

Στη γραμμή τίτλου εμφανίζεται ο τίτλος του έργου




***Εικόνα 13-4.***

```
MsgBox("ÈÝëåôå íá êÜíåôå êáé Üëëç
áíáæÞôçóç;", _
       vbQuestion + _
       vbOKCancel + _
       vbDefaultButton2, _
"ÁíáæÞôçóç")
```




***Εικόνα 13-5.***

```
MsgBox("Ôá äåäïìÝíá âßíáé åêôüò ïñßùí" & _
       vbCrLf & "ÈÝëåôå íá óõíå÷ßóåôå;", _
       vbExclamation + _
       vbYesNo, _
"Ðñïåéäïðïßçóç")
```

Η τιμή που επιστρέφει η συνάρτηση MsgBox μας βοηθά να προσδιορίσουμε το πλήκτρο που πάτησε ο χρήστης, ως απόκρισή του στο διαλογικό παράθυρο. Οι επιστρεφόμενες τιμές και η περιγραφή τους δίνονται στον πίνακα 13-2.

| Σταθερά | Τιμή | Περιγραφή |
|---|---|---|
| VbOK | 1 | Πατήθηκε το πλήκτρο Ok |
| VbCancel | 2 | Πατήθηκε το πλήκτρο Cancel |
| VbAbort | 3 | Πατήθηκε το πλήκτρο Abort |
| VbRetry | 4 | Πατήθηκε το πλήκτρο Retry |
| vbIgnore | 5 | Πατήθηκε το πλήκτρο Ignore |
| vbYes | 6 | Πατήθηκε το πλήκτρο Yes |
| vbNo | 7 | Πατήθηκε το πλήκτρο No |

***Πίνακας 13-2***. Αντιστοιχία πλήκτρων με σταθερές.

**Παράδειγμα 13-3.**

Το τμήμα προγράμματος που εμφανίζει το διαλογικό παράθυρο της εικόνας 13-3, ανιχνεύει το πλήκτρο που πατά ο χρήστης, και εκτελεί κατά περίπτωση την κατάλληλη λειτουργία, έχει τη μορφή:

```
UserAnswer = MsgBox("ÁðÝôõ÷å ç óýíäåóç óôï äßêôõï", _
                    vbInformation + vbAbortRetryIgnore + vbDefaultButton3)
Select Case UserAnswer
    Case vbAbort
        'Ãåêáôüëåéøå êÜèå ðñïóðÜèåéá
                    :
    Case vbRetry
        'ÐñïóðÜèçóå îáíÜ
                    :
    Case vbIgnore
        'Ãåíüçóå ôçí áóôïéµßá êáé óõíÝ÷éóå
                    :
End select
```

Την `MsgBox` μπορούμε να τη χρησιμοποιήσουμε και σαν εντολή στην περίπτωση που θέλουμε να εμφανίσουμε κάποια πληροφορία χωρίς να μας ενδιαφέρει η απόκριση του χρήστη. Τότε συντάσσεται ως:

`MsgBox` *μήνυμα, μορφή, τίτλος*

## Η συνάρτηση InputBox

Η συνάρτηση `InputBox` εμφανίζει ένα διαλογικό παράθυρο που περιέχει ένα πλαίσιο κειμένου και δύο πλήκτρα διαταγής, το `Ok` και το `Cancel`. Ο χρήστης πρέπει να συμπληρώσει το πλαίσιο κειμένου και να κάνει κλικ σε ένα από τα δύο πλήκτρα. Η τιμή που επιστρέφει η συνάρτηση με επιλογή του `Ok` είναι ακριβώς το περιεχόμενο του πλαισίου κειμένου, ενώ η επιλογή του πλήκτρου `Cancel` επιστρέφει τη μηδενική συμβολοσειρά (""). Η γενική μορφή της συνάρτησης είναι:

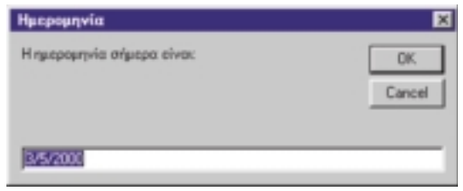
`InputBox`(*μήνυμα, τίτλος, προεπιλογή, x, y*)

με μοναδική υποχρεωτική παράμετρο το *μήνυμα*. Σ' αυτήν την παράμετρο γράφουμε ένα βοηθητικό κείμενο που καθοδηγεί το χρήστη και το οποίο μπορεί να είναι μέχρι 1024 περίπου χαρακτήρες (εξαρτάται από το πλάτος των χρησιμοποιούμενων χαρακτήρων). Η παράμετρος *τίτλος* χρησιμεύει για να καθορίσουμε τον τίτλο του διαλογικού παραθύρου. Αν παραλείψουμε αυτήν την παράμετρο, ως τίτλος του παραθύρου εμφανίζεται το όνομα του έργου της εφαρμογής. Η παράμετρος *προεπιλογή* μας επιτρέπει να εμφανίζουμε αυτόματα, με την παρουσίαση του διαλογικού παραθύρου, κάποια προκαθορισμένη απάντηση σε περίπτωση που ο χρήστης δεν πληκτρολογήσει κάποια τιμή. Αν δε δώσουμε τιμή σε αυτήν την παράμετρο το πλαίσιο κειμένου εμφανίζεται άδειο. Τέλος, με τις παραμέτρους *x* και *y* μπορούμε να προσδιορίσουμε τη θέση εμφάνισης του διαλογικού παραθύρου σε σχέση με την πάνω αριστερή γωνία της οθόνης.

Η συνάρτηση `InputBox` προσφέρει ένα γρήγορο τρόπο για να ζητήσουμε από το χρήστη απλές πληροφορίες, όπως ένα συνθηματικό ή ένα όνομα αρχείου. Το εμφανιζόμενο διαλογικό παράθυρο δεν είναι ιδιαίτερα ποιοτικό. Το εύρος των εφαρμογών της συνάρτησης είναι περιορισμένο.

**Άσκηση 13-1.**

Να δημιουργηθεί ένα παράθυρο, στο οποίο το πάτημα του πλήκτρου διαταγής `SetDate` να εμφανίζει διαλογικό παράθυρο, στο οποίο ο χρήστης να είναι υποχρεωμένος να εισάγει μια ημερομηνία.




***Εικόνα 13-6***. *Διαλογικό παράθυρο από συνάρτηση* `InputBox`



Ο κώδικας που επισυνάπτουμε στο συμβάν **`Click`** του πλήκτρου είναι:

```
Private Sub SetDate_Click()
  Dim Today   As String
    Do
        Today = InputBox("Ç çìåñïìçíßá óÞìåñá åßíáé:", _
                         "Çìåñïìçíßá", _
                         CStr(Date))
    Loop Until IsDate(Today)
    Date = Today
End Sub
```

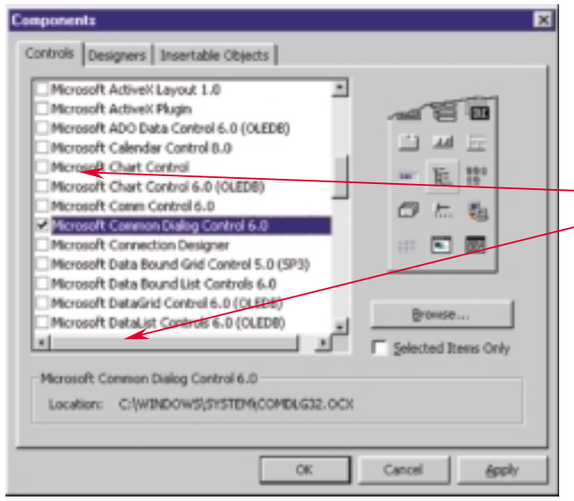
## Προτυποποιημένα κοινά διαλογικά παράθυρα

Κατά κανόνα οι χρήστες των Windows είναι εξοικειωμένοι με κάποια τυποποιημένα διαλογικά παράθυρα του Microsoft Office. Για παράδειγμα, γνωρίζουν πώς να επιλέξουν ένα αρχείο στο διαλογικό παράθυρο που εμφανίζει η επιλογή **File ¦ Open** του επεξεργαστή κειμένου Word ή του υπολογιστικού φύλλου Excel. Επίσης, γνωρίζουν τις ρυθμίσεις εκτύπωσης που πρέπει να κάνουν στο διαλογικό παράθυρο που εμφανίζει η επιλογή **File ¦ Print.** Από την πλευρά του προγραμματιστή πάλι, είναι πολύ δύσκολο να δημιουργηθούν τόσο σύνθετα και αξιόπιστα στη λειτουργία διαλογικά παράθυρα.

Όμως, υπάρχει ένα πρόσθετο αντικείμενο ελέγχου, που είναι γνωστό με το όνομα **CommonDialog** (***κοινό διαλογικό παράθυρο)***, το οποίο μπορούμε να εκμεταλλευτούμε και να προβάλλουμε διαλογικά παράθυρα για:

- Επιλογή αρχείου με σκοπό να το ανοίξουμε (Open).
- Επιλογή αρχείου με σκοπό να σώσουμε δεδομένα (Save).
- Καθορισμό ρυθμίσεων εκτύπωσης (Print).
- Χρήση αρχείου βοήθειας (Help).
- Επιλογή γραμματοσειράς (Font).
- Επιλογή χρώματος (Color).

Αν στην εργαλειοθήκη δεν υπάρχει το αντίστοιχο εργαλείο για την προσθήκη κοινών διαλογικών παραθύρων, κάνουμε τα εξής:




1. Επιλέγουμε **Project ¦ Components** από τη γραμμή μενού.
2. Κάνουμε κλικ στον καρτελοδείκτη **Controls.**
3. Κάνουμε κλικ στη γραμμή **Microsoft Common Dialog Control.**
4. Το εικονίδιο του αντικειμένου εμφανίζεται στην εργαλειοθήκη.

***Εικόνα 13-7.*** *Προσθήκη του CommonDialog στην εργαλειοθήκη*

Ένα αντικείμενο "κοινό διαλογικό παράθυρο" τοποθετείται πάνω σε μια φόρμα, κατά τη σχεδίαση της διεπαφής, και παραμένει αόρατο κατά τη διάρκεια εκτέλεσης του προγράμματος. Το κοινό διαλογικό παράθυρο δεν υπακούει σε συμβάντα και διαθέτει μόνον ιδιότητες και μεθόδους. Οι μέθοδοι που υποστηρίζει, καθορίζουν τον τύπο του παραθύρου που θα

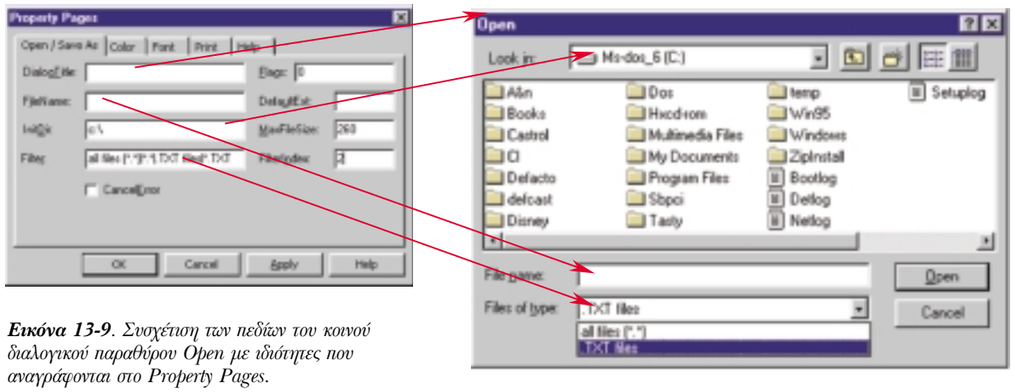
εμφανιστεί κατά τη διάρκεια εκτέλεσης του προγράμματος. Επίσης, οι ιδιότητες του, καθορίζουν τα χαρακτηριστικά αυτών των παραθύρων. Οι μέθοδοι που προκαλούν εμφάνιση των διαλογικών παραθύρων είναι οι:

| | |
|---|---|
| `ShowOpen` | Άνοιξε αρχείο |
| `ShowSave` | Σώσε σε αρχείο |
| `ShowColor` | Επίλεξε χρώμα |
| `ShowFont` | Επίλεξε γραμματοσειρά |
| `ShowPrinter` | Χαρακτηριστικά εκτύπωσης |
| `ShowHelp` | Επίκληση βοήθειας |

Κάνοντας κλικ στην επιλογή **Custom** του παραθύρου ιδιοτήτων ενός αντικειμένου "κοινό διαλογικό παράθυρο", εμφανίζεται το παράθυρο **Property Pages**. Αυτό το παράθυρο έχει μια σειρά σελιδοδείκτες, που αντιστοιχούν στις μεθόδους που περιγράψαμε παραπάνω. Επιλέγοντας ένα σελιδοδείκτη βλέπουμε τις ιδιότητες που σχετίζονται με το διαλογικό παράθυρο που θα εμφανιστεί κατά τη διάρκεια εκτέλεσης του προγράμματος τη στιγμή κλήσης της συγκεκριμένης μεθόδου.

**Παράδειγμα 13-4.**

Ο πρώτος καρτελοδείκτης αντιστοιχεί στις μεθόδους ShowOpen και ShowSave.




***Εικόνα 13-9****. Συσχέτιση των πεδίων του κοινού διαλογικού παραθύρου Open με ιδιότητες που αναγράφονται στο Property Pages.*

Η ιδιότητα **`DialogTitle`** περιέχει την επικεφαλίδα του παραθύρου. Αν είναι κενή εμφανίζεται η τυποποιημένη επικεφαλίδα του αντίστοιχου κοινού διαλογικού παραθύρου (στην εικόνα 13-9 η επικεφαλίδα `Open`).

Η ιδιότητα **`FileName`** περιέχει το όνομα αρχείου που επιθυμούμε να είναι προεπιλεγμένο. Η ιδιότητα **`InitDir`** περιέχει το φάκελο των αρχείων, που θα εμφανιστεί αρχικά στην περιοχή **Look in**.

Η ιδιότητα **`Filter`** περιέχει μια λίστα με τις περιγραφές των τύπων των αρχείων και των φίλτρων που θέλουμε να υπάρχουν στην πτυσσόμενη λίστα της περιοχής **Files of type**. Στην εικόνα 13-8 η πτυσσόμενη λίστα περιέχει τις περιγραφές "`all files (*.*)`" και ".ô×ô `files`". Σε αυτήν την ιδιότητα, το σύμβολο (¦) παίζει το ρόλο του διαχωριστικού μεταξύ των περιγραφών και των φίλτρων, τα οποία εναλλάσσονται μέσα στη συμβολοσειρά της τιμής της. Σχετική είναι και η ιδιότητα **`FilterIndex`**, η οποία καθορίζει πιο από τα φίλτρα τς λίστας θα είναι προεπιλεγμένο. Αν έχει την τιμή 0 δεν επιλέγεται κανένα φίλτρο και στο πινάκιο των φακέλων και των αρχείων εμφανίζονται όλα τα αρχεία. Στην εικόνα 13-8 η ιδιότητα **`FilterIndex`** έχει την τιμή 2, οπότε προεπιλέγεται το δεύτερο φίλτρο.

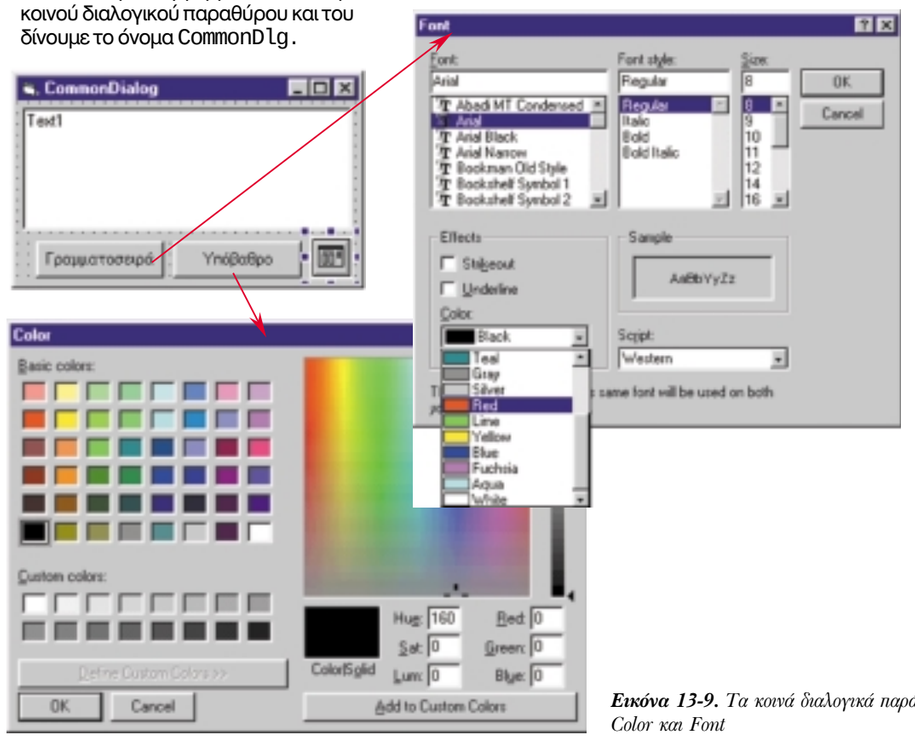
**Άσκηση 13-2.**

Να δημιουργηθεί φόρμα στην οποία να υπάρχει πλαίσιο κειμένου με όνομα `TextArea` και δύο πλήκτρα, το `SetBackColor` και το `SetFont`. Όταν ο χρήστης πατά το πλήκτρο `SetBackColor` να εμφανίζεται διαλογικό παράθυρο, από το οποίο να κάνει την επιλογή χρώματος του φόντου του πλαισίου κειμένου και όταν πατά το πλήκτρο `SetFont` να



διαλογικό παράθυρο από το οποίο να κάνει την επιλογή γραμματοσειράς.

1. Τοποθετούμε πάνω στη φόρμα τα ορατά αντικείμενα.
2. Τοποθετούμε στη φόρμα ένα αντικείμενο κοινού διαλογικού παραθύρου και του δίνουμε το όνομα `CommonDlg`.




***Εικόνα 13-9.*** *Τα κοινά διαλογικά παράθυρα Color και Font*

3. Γράφουμε τον κώδικα:

```
Private Sub SetBackColor_Click()
  'Ïñéóå ôéò éäéüôçôåò êáé åìöÜíéóå ôï êïéíü äéáëïãéêü ðáñÜèõñï
   CommonDlg.Flags = cdlCCFullOpen
   CommonDlg.ShowColor
  'ÌåôÝöåñå ôéò åðéëïãÝò ôïõ ÷ñþìáôïò
   TextArea.BackColor = CommonDlg.Color
End Sub

Private Sub SetFont_Click()
  'Ïñéóå ôéò éäéüôçôåò êáé åìöÜíéóå ôï êïéíü äéáëïãéêü ðáñÜèõñï
   CommonDlg.Flags = cdlCFBoth Or cdlCFEffects
   CommonDlg.ShowFont
  'ÌåôÝöåñå ôéò åðéëïãÝò ôïõ ÷ñþìáôïò
   TextArea.FontName = CommonDlg.FontName
   TextArea.FontSize = CommonDlg.FontSize
   TextArea.FontBold = CommonDlg.FontBold
   TextArea.FontItalic = CommonDlg.FontItalic
   TextArea.FontStrikethru = CommonDlg.FontStrikethru
   TextArea.FontUnderline = CommonDlg.FontUnderline
   TextArea.ForeColor = CommonDlg.Color
End Sub
```

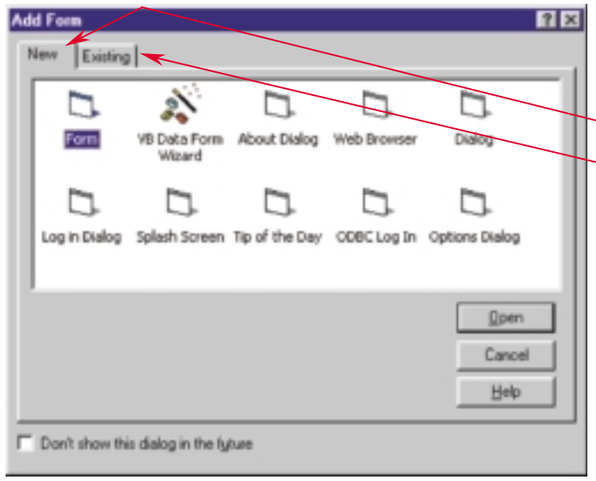
Η ιδιότητα **`Flags`** καθορίζει το πώς θα εμφανίζεται το κοινό διαλογικό παράθυρο. Στην υπορουτίνα `SetBackColor_Click` της δίνεται, ως τιμή η εσωτερική σταθερά `cdlCCFullOpen` για να εμφανιστεί το παράθυρο σε πλήρη ανάπτυξη με την

αναπτυγμένη παλέτα χρωμάτων. Στην υπορουτίνα `SetFont_Click` της δίνεται κατάλληλη τιμή, ώστε να εμφανιστούν στο παράθυρο οι γραμματοσειρές οθόνης, οι γραμματοσειρές εκτυπωτή (`cdlCFBoth`) αλλά και οι επιλογές ειδικών εφέ υπογράμμισης, διαγραφής, χρώματος (`cdlCFEffects`). Υπάρχει πλήθος εσωτερικών σταθερών για την ιδιότητα **`Flags`**. Για να δούμε αυτές τις σταθερές πρέπει να ανατρέξουμε στην επιλογή **Help ¦ Books Online** και να κάνουμε αναζήτηση με κωδική λέξη `CommonDialog`.

## Εφαρμογές με πολλές φόρμες

Στις περισσότερες εφαρμογές ένα παράθυρο δεν είναι αρκετό για να καλύψει όλες τις ανάγκες της διεπαφής με το χρήστη. Όπως είδαμε, τα διαλογικά παράθυρα μπορούν να συμβάλλουν στη βελτίωση της διεπαφής, αλλα δεν είναι και τόσο ευέλικτα για να καλύψουν όλες τις ανάγκες μας. Με την VB είναι δυνατόν να αναπτύξουμε εφαρμογές που συντίθενται από πολλές φόρμες, ώστε και η διεπαφή να αποτελείται από πολλά παράθυρα.

Για να προσθέσουμε μια νέα φόρμα στην εφαρμογή, επιλέγουμε **Project ¦ Add Form** από τη γραμμή μενού. Στη συνέχεια, από την περιοχή υποδειγμάτων του παραθύρου `Add Form` διαλέγουμε το πρότυπο μιας φόρμας (συνήθως το προτεινόμενο υπόδειγμα `Form`) και στη συνέχεια πατάμε το πλήκτρο **`Open`**.

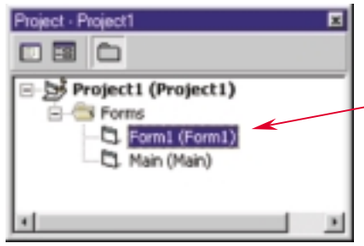



***Εικόνα 13-10.*** *Επιλογή υποδείγματος φόρμας*

Δημιουργία νέας φόρμας

Επιλογή μιας από τις ήδη υπάρχουσες φόρμες

Στην οθόνη εμφανίζεται η νέα φόρμα και ταυτόχρονα στο παράθυρο έργου, κάτω από τον φάκελο `Forms`, εμφανίζεται και το όνομα της νέας φόρμας. Τώρα πλέον είναι δυνατόν να αλλάξουμε το όνομα της φόρμας, τον τίτλο του παραθύρου της, τις διαστάσεις της και το όνομα του αρχείου στο οποίο θα αποθηκευθεί, σύμφωνα με όσα περιγράψαμε στο μάθημα 4 (άσκηση 4-1).




***Εικόνα 13-11****. Το όνομα της νέας φόρμας εμφανίζεται και στο παράθυρο έργου*

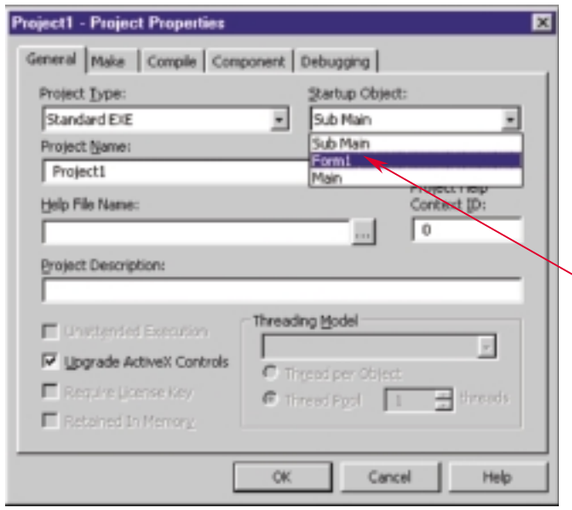
Με διπλό κλικ πάνω στο όνομα της φόρμας, η φόρμα έρχεται στο προσκήνιο, ακόμη και αν καλύπτεται ήδη από άλλες.

Κάθε φόρμα είναι απολύτως αυτόνομη. Το αρχείο στο οποίο αποθηκεύεται περιέχει την περιγραφή του οπτικού σχεδίου της φόρμας και τον κώδικα της προγραμματιστικής της μονάδας. Αυτό έχει πολλαπλά οφέλη. Αν η φόρμα σχεδιαστεί κατάλληλα, μπορεί να χρησιμοποιηθεί ακριβώς η ίδια από πολλές εφαρμογές. Έτσι, πετυχαίνουμε και μείωση του



χρόνου ανάπτυξης νέων εφαρμογών, αφού είναι δυνατόν να εκμεταλλευτούμε παλαιότερες εργασίες μας, αλλά και ομοιογένεια στη μορφή των εφαρμογών που αναπτύσσουμε. Η προσθήκη μιας φόρμας, που ήδη υπάρχει, γίνεται επιλέγοντας τον καρτελοδείκτη **Existing** του διαλογικού παραθύρου **Add Form.**

Η φόρμα με την οποία θα εκκινήσει μια εφαρμογή ορίζεται από την επιλογή **Project ¦ *όνομα_έργου* Properties.** Αφού επιλέξουμε τον καρτελοδείκτη **General,** επιλέγουμε μέσα από την πτυσσόμενη λίστα **StartUp Object** το όνομα της ***φόρμας εκκίνησης.***




Ως φόρμα εκκίνησης του έργου `Project1` επιλέγεται η φόρμα `Form1`.

***Εικόνα 13-12****. Το διαλογικό παράθυρο με τις ιδιότητες του έργου*

Μια εφαρμογή εκκινεί εμφανίζοντας ένα μόνο παράθυρο. Τα υπόλοιπα παράθυρα δεν φορτώνονται αμέσως όλα μαζί στη μνήμη του υπολογιστή και δεν είναι ορατά. Ο προγραμματιστής με κατάλληλη κωδικοποίηση διαμορφώνει την εφαρμογή, ώστε ο χρήστης να ανοίγει και να κλείνει τα παράθυρά της κατά βούληση. Για να φορτωθεί ένα παράθυρο στη μνήμη του υπολογιστή, πρέπει να εκτελεστεί η εντολή `Load`, η οποία έχει τη μορφή:

`Load` *φόρμα*

Το παράθυρο της *φόρμας* ναι μεν φορτώνεται στη μνήμη, οπότε είναι δυνατόν να γίνουν αναφορές σε ιδιότητες και αντικείμενά του, δεν παρουσιάζεται όμως ακόμη στην οθόνη. Για να γίνει ορατό το παράθυρο στην οθόνη, πρέπει να εκτελεστεί η μέθοδος `Show`, η οποία έχει τη μορφή:

*φόρμα.* `Show` *στυλ*

Αν το *στυλ* έχει την τιμή 1 (εσωτερική σταθερά `vbModal`) το παράθυρο της *φόρμας* "δεσμεύει" αποκλειστικά την προσοχή του χρήστη (υποχρεωτικής απόκρισης) και δεν τον αφήνει να αλληλεπιδράσει με άλλα παράθυρα. Αν το *στυλ* έχει την τιμή 0, ή δεν υπάρχει η παράμετρος *στυλ*, ο χρήστης μπορεί να αλληλεπιδράσει και με άλλα παράθυρα. Η μέθοδος `Show` μπορεί να εκτελεστεί χωρίς να έχει προηγηθεί εκτέλεση της εντολής `Load`. Σε αυτήν την περίπτωση είναι σαν να προηγείται η εκτέλεση της εντολής `Load`.

Για να κάνουμε αόρατο ένα παράθυρο, χωρίς να το ξεφορτώσουμε από τη μνήμη, πρέπει να εκτελέσουμε τη μέθοδο `Hide`, η οποία έχει τη μορφή:

*φόρμα.* `Hide`

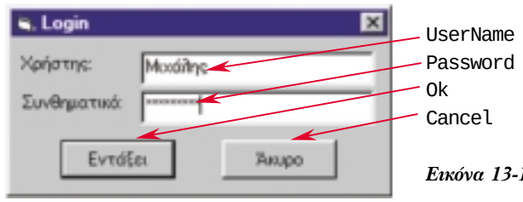
Για να ξεφορτώσουμε ένα παράθυρο από τη μνήμη, εκτελούμε τη μέθοδο `Unload`, η οποία έχει τη μορφή:

`UnLoad` *φόρμα*

Η εντολή `UnLoad` μπορεί να εκτελεστεί χωρίς να έχει προηγηθεί εκτέλεση της μεθόδου `Hide`.

**Άσκηση 13-3.**

Σε ένα πρόγραμμα υπάρχει μια φόρμα με το όνομα Main. Να δημιουργηθεί μια ακόμα φόρμα με όνομα Login, στην οποία να υπάρχουν δύο πλαίσια κειμένου με όνομα UserName και Password, καθώς και δύο πλήκτρα, το Ok και το Cancel. Ως φόρμα εκκίνησης της εφαρμογής να καθοριστεί η φόρμα Login. Το πάτημα του πλήκτρου Cancel να προκαλεί το κλείσιμο της φόρμας και τον τερματισμό της εφαρμογής. Το πάτημα του πλήκτρου Ok να προκαλεί τον έλεγχο του συνθηματικού. Αν το συνθηματικό βρεθεί σωστό να κλείνει η φόρμα Login και να φορτώνεται η φόρμα Main.




*Εικόνα 13-13.*

Για να εμφανίζονται αστερίσκοι στη θέση των χαρακτήρων που πληκτρολογούμε, δίνουμε στην ιδιότητα **PasswordChar** του πλαισίου κειμένου Password την τιμή *. Στη συνέχεια, στο παράθυρο κωδικοποίησης γράφουμε τον κώδικα:

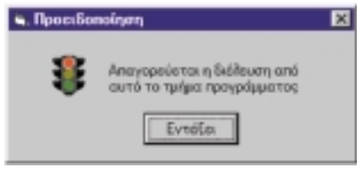
```
Private Sub Cancel_Click()
   Unload Me
End Sub

Private Sub OK_Click()
   If Password = "password" Then
     Me.Hide
     Main.Show
   Else
     MsgBox "Ôï óõíèçìáôéêü äåí åßíáé óùóôü!", , "Login"
     Password.SetFocus
   End If
End Sub
```

Δημιουργία πρωτότυπων πλαισίων διαλόγου

Όταν οι ανάγκες μας επιβάλλουν τη δημιουργία ενός πλαισίου διαλόγου διαφορετικού από αυτά που μπορεί να μας προσφέρει έτοιμα η VB, μπορούμε να το δημιουργήσουμε από την αρχή χρησιμοποιώντας μια φόρμα. Συγκεκριμένα, σε μια φόρμα κάνουμε τα εξής:

1. Δίνουμε στην ιδιότητα **Border** της φόρμας την τιμή **Fixed Dialog** (τα πλαίσια διαλόγου δεν πρέπει να αλλάζουν μέγεθος). Ας σημειωθεί, ότι δίνοντας αυτήν την τιμή στην ιδιότητα **Border** δε λαμβάνονται υπ' όψιν οι τιμές των ιδιοτήτων **MaxButton** και **MinButton** και το παράθυρο θα εμφανιστεί χωρίς πλήκτρα.
2. Ορίζουμε τον τίτλο του πλαισίου μέσω της ιδιότητας **Caption** της φόρμας.
3. Τοποθετούμε πάνω στη φόρμα τα αντικείμενα ελέγχου.
4. Εμφανίζουμε τη φόρμα ως υποχρεωτικό διαλογικόμ παράθυρο δίνοντας στην παράμετρο *στυλ* της μεθόδου **Show** την τιμή vbModal, ή ως μη υποχρεωτικό διαλογικό παράθυρο δίνοντας της την τιμή vbModeless.

**Παράδειγμα 13-5.**




***Εικόνα 13-14**. Πλαίσιο διαλόγου έξω από τα καθερωμένα*

Το εικονίδιο στο εσωτερικό του παραθύρου δεν είναι ένα από τα προτυποποιημένα. Ο προγραμματιστής μπορεί να διαλέξει ένα οποιοδήποτε. Επίσης, μπορεί να τοποθετήσει πλαίσια κειμένου και άλλα αντικείμενα ελέγχου χωρίς περιορισμούς.



Κατά τη φόρτωση ενός παραθύρου στη μνήμη, είτε με την εκτέλεση της εντολής Load είτε με την εκτέλεση της μεθόδου Show, προκύπτει το συμβάν **Load**. Συνήθως, στην υπορουτίνα διαχείρισης συμβάντος Form_Load γίνονται οι αρχικοποιήσεις των τιμών των μεταβλητών. Επίσης, κατά την εκφόρτωση ενός παραθύρου από τη μνήμη προκύπτει το συμβάν **Unload**. Συνήθως, στην υπορουτίνα διαχείρισης συμβάντος Form_Unload γίνονται αποδεσμεύσεις πόρων (συσκευών, αρχείων κ.ά) του υπολογιστικού συστήματος.

**Παράδειγμα 13-6.**

Στην υπορουτίνα διαχείρισης συμβάντος **Unload** που ακολουθεί, εμφανίζεται ένα διαλογικό παράθυρο το οποίο ρωτά τον χρήστη, αν επιθυμεί τη διακοπή του προγράμματος. Αν ο χρήστης απαντήσει αρνητικά δίνεται η τιμή True στην παράμετρο Cancel της υπορουτίνας. Σ' αυτήν την περίπτωση η VB αναστέλλει τη διαδικασία Unload.

```
Private Sub Form_Unload(Cancel As Integer)
   Dim Answer       As Long
   Answer = MsgBox("Ôåñìáôéóìüò Ðñïãñáììáôïò", vbYesNo)
   If Answer = vbNo Then
      Cancel = True
   Else
      End
   End If
End Sub
```

Από μια φόρμα μπορούμε να αναφερθούμε σε ένα αντικείμενο ελέγχου μιας άλλης φόρμας. Η αναφορά γίνεται με αναγραφή του ονόματος της φόρμας πριν από το όνομα του αντικειμένου ελέγχου.

**Παράδειγμα 13-7**.

Στη φόρμα Main υπάρχει πλαίσιο κειμένου με το όνομα Money. Από μια άλλη φόρμα, ή από μια βασική προγραμματιστική μονάδα, είναι δυνατή η αλλαγή των ιδιοτήτων του πλαισίου κειμένου.

```
Main.Money.Text = "ÊÜôù üñéï"
Main.Money.ForeColor = vbYellow
Main.Money.BackColor = vbRed
```

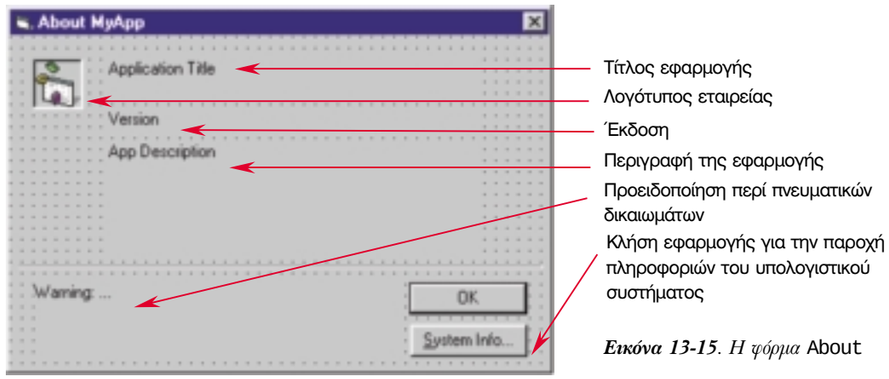
## Ανακεφαλαίωση

Η συνάρτηση MsgBox εμφανίζει στην οθόνη ένα διαλογικό παράθυρο, το οποίο περιέχει ένα μήνυμα προς το χρήστη και ένα ή περισσότερα πλήκτρα διαταγής. Η τιμή που επιστρέφει η συνάρτηση είναι ένας ακέραιος αριθμός και καθορίζεται από το πλήκτρο που πατάει ο χρήστης για να κλείσει το διαλογικό παράθυρο. Η συνάρτηση InputBox, εμφανίζει ένα διαλογικό παράθυρο που περιέχει ένα πλαίσιο κειμένου και δύο πλήκτρα διαταγής, το Ok και το Cancel. Η συνάρτηση επιστρέφει ως τιμή το περιεχόμενο του πλαισίου κειμένου, αν πατηθεί το πλήκτρο Ok, ενώ η επιλογή του πλήκτρου Cancel, επιστρέφει τη μηδενική συμβολοσειρά (""). Με το αντικείμενο ελέγχου "κοινό διαλογικό παράθυρο" μπορούμε να εκμεταλλευτούμε τις δυνατότητες των διαλογικών παραθύρων για να επιλέξουμε αρχείο με σκοπό να το ανοίξουμε (Open), να επιλέξουμε αρχείο με σκοπό να αποθηκεύσουμε δεδομένα (Save), να καθορίσουμε τις ρυθμίσεις εκτύπωσης (Print), να καλέσουμε αρχείο βοήθειας (Help), να επιλέξουμε γραμματοσειρές (Font), να επιλέξουμε χρώμα (Color).

Με την VB είναι δυνατόν να αναπτύξουμε εφαρμογές που συντίθενται από πολλές φόρμες. Κάθε φόρμα είναι απολύτως αυτόνομη. Το αρχείο στο οποίο αποθηκεύεται μια φόρμα περιέχει και την περιγραφή του οπτικού της σχεδίου και τον κώδικα της προγραμματιστικής της μονάδας. Για να φορτωθεί ένα παράθυρο στη μνήμη του υπολογιστή, πρέπει να εκτελεστεί η εντολή Load. Για να γίνει ορατό το παράθυρο και στην οθόνη, πρέπει να εκτελεστεί η μέθοδος Show. Για να κάνουμε αόρατο ένα παράθυρο, χωρίς να το ξεφορτώσουμε από τη μνήμη, πρέπει να εκτελέσουμε τη μέθοδο Hide. Για να ξεφορτώσουμε ένα παράθυρο από τη μνήμη εκτελούμε τη μέθοδο Unload.

## Εργαστηριακές Ασκήσεις

1. Να εκτελεστεί η άσκηση 13-1. Πώς μπορείτε να ελέγξετε, αν το πρόγραμμα αλλάζει την ώρα του υπολογιστικού συστήματος;
2. Να εκτελεστεί η άσκηση 13-2. Να γίνουν δοκιμές για διάφορες γραμματοσειρές και χρώματα.
3. Να εκτελεστεί η άσκηση 13-3. Να τοποθετήσετε τη φόρμα `Login` σε μια από τις εφαρμογές που έχετε υλοποιήσει σε προηγούμενα μαθήματα.
4. Αποτελεί κοινή τεχνική, στην αρχή του προγράμματος να εμφανίζεται ένα παράθυρο (***παράθυρο splash***) με κάποια πληροφοριακά στοιχεία για το πρόγραμμα και με τον λογότυπο της εταιρείας που έχει κάνει την κατασκευή του. Όσο διάστημα παρατηρεί ο χρήστης το παράθυρο στην οθόνη, ο κώδικας της φόρμας φορτώνει και τις υπόλοιπες φόρμες του προγράμματος κρατώντας αυτές αόρατες. Οι φόρμες γίνονται ορατές όταν ζητηθεί μέσα από τον κώδικα και η ταχύτητα εμφάνισής τους είναι πολύ μεγαλύτερη από αυτή που θα ήταν αν φορτώνονταν εκείνη τη στιγμή από το δίσκο. Να προσθέσετε σε μια εφαρμογή σας ένα παράθυρο splash.
   Υπόδειξη: Υπάρχει υπόδειγμα φόρμας στο πινάκιο υποδειγμάτων του διαλογικού παραθύρου **Add Form.**
5. Θα έχετε παρατηρήσει ότι σε όλες σχεδόν τις επαγγελματικές εφαρμογές υπάρχει και μια επιλογή στο μενού **Help,** η επιλογή **About,** από την οποία δίνονται πληροφορίες για την εφαρμογή, το υλικό και το λογισμικό που είναι εγκατεστημένο στον υπολογιστή. Ένα από τα υποδείγματα δημιουργίας φόρμας της VB, που εμφανίζονται στο παράθυρο `Add Form` της δημιουργίας φόρμας είναι το `About Dialog` που δημιουργεί φόρμες του τύπου




***Εικόνα 13-15****. Η φόρμα* `About`

Σε ένα από τα προγράμματα που έχετε δημιουργήσει, να εισάγετε τη δυνατότητα εμφάνισης του παραθύρου About είτε από επιλογή μενού είτε από το πάτημα πλήκτρου. Συμπληρώστε κατάλληλα τις ετικέτες και το πλαίσιο εικόνας της φόρμας.



## Μάθημα 14
# Πρόσθετα αντικείμενα ελέγχου Εκτυπώσεις

Τελειώνοντας αυτό το μάθημα οι μαθητές θα μπορούν:

- Να δημιουργούν λίστες στην οθόνη.
- Να προσφέρουν τη δυνατότητα επιλογής συγκεκριμένων λειτουργιών σε έναν χρήστη.
- Να εκτυπώνουν τα αποτελέσματα ενός προγράμματος στον εκτυπωτή.

Όσο αποτελεσματικός και αποδοτικός αν είναι ο κώδικας του προγράμματος που γράφουμε για μια εφαρμογή, δεν αρκεί για να της δώσει επαγγελματική μορφή και να την κάνει αποδεκτή από τους υποψήφιους χρήστες της. Έχουμε ήδη δει, ότι ο σχεδιασμός της διεπαφής με το χρήστη αποτελεί το πιο σημαντικό στοιχείο για την ανάδειξη μιας εφαρμογής. Η διεπαφή πρέπει να προσφέρει ευκολία χρήσης, να κεντρίζει το ενδιαφέρον του χρήστη και γενικά να τον προδιαθέτει θετικά απέναντι στην εφαρμογή.

Μέχρι τώρα για τη δημιουργία της διεπαφής χρησιμοποιήσαμε τα πιο βασικά αντικείμενα ελέγχου, την ετικέτα, το πλαίσιο κειμένου, το πλήκτρο διαταγής. Σ' αυτό το μάθημα θα μελετήσουμε και τον τρόπο εκμετάλλευσης και κάποιων άλλων βασικών αντικειμένων ελέγχου της εργαλειοθήκης και θα δούμε πώς μπορούμε να εμπλουτίσουμε την εργαλειοθήκη με εργαλεία που δημιουργούν πρόσθετα αντικείμενα ελέγχου του τύπου ActiveX. Επίσης, θα μελετήσουμε πώς γίνονται εκτυπώσεις μέσω του ειδικού αντικειμένου εκτυπωτή.

## Λίστα και Συνδυασμένη λίστα

Η λίστα (ListBox) και η συνδυασμένη λίστα (ComboBox) είναι ιδιαίτερα χρήσιμα αντικείμενα ελέγχου και χρησιμοποιούνται στις περιπτώσεις που θέλουμε να εμφανίσουμε, σε μορφή καταλόγου, μια σειρά από επιλογές από τις οποίες ο χρήστης να μπορεί να επιλέξει μια κατά περίπτωση.

Η λίστα χρησιμοποιείται όταν θέλουμε να δεσμεύσουμε τον χρήστη να επιλέξει μέσα από ένα περιορισμένο πλήθος επιλογών. Αντίθετα, η συνδυασμένη λίστα χρησιμοποιείται στις περιπτώσεις που θέλουμε να δημιουργήσουμε μια ανοιχτή λίστα, δηλαδή μια λίστα την οποία να μπορεί να ενημερώσει ο χρήστης. Η συνδυασμένη λίστα αποτελεί συνδυασμό μιας λίστας και ενός πλαισίου κειμένου. Αν η λίστα δεν περιέχει όλες τις επιλογές που θα ήθελε ένας χρήστης, αυτός μπορεί να της προσθέσει μια νέα επιλογή πληκτρολογώντας την στο πλαίσιο κειμένου.


Λίστα Συνδυασμένη λίστα

***Εικόνα 14-1.***

Η εμφάνιση του πλαισίου συνδυασμένης λίστας καθορίζεται από την ιδιότητα **`Style`**. Οι τιμές που μπορεί να πάρει η ιδιότητα **`Style`** είναι:

**0 - Dropdown Combo**. Το αντικείμενο μοιάζει με πλαίσιο κειμένου που έχει ένα βέλος στα δεξιά. Με κλικ πάνω στο βέλος, η λίστα αναπτύσσεται προς τα κάτω (πτυσσόμενη λίστα). Το περιεχόμενο της λίστας μπορεί να συμπληρωθεί με νέα στοιχεία, τα οποία πληκτρολογεί ο χρήστης στο πλαίσιου κειμένου (ανοιχτή λίστα).

**1 - Simple Combo**. Το αντικείμενο μοιάζει με λίστα πάνω από την οποία υπάρχει πλαίσιο κειμένου. Το περιεχόμενο της λίστας μπορεί να συμπληρωθεί με νέα στοιχεία, τα οποία πληκτρολογεί ο χρήστης στο πλαίσιου κειμένου (ανοιχτή λίστα).

**2 - Dropdown List**. Το αντικείμενο μοιάζει με πλαίσιο κειμένου που έχει ένα βέλος στα δεξιά. Με κλικ πάνω στο βέλος, η λίστα αναπτύσσεται προς τα κάτω (πτυσσόμενη λίστα). Το περιεχόμενο της λίστας δεν μπορεί να συμπληρωθεί με νέα στοιχεία (κλειστή λίστα).

Αν η ιδιότητα **`Style`** έχει την τιμή 0 ή 1, όταν γίνεται τροποποίηση του στοιχείου στο πλαίσιο κειμένου, προκαλείται συμβάν **`Change`**.

Προσθήκη στοιχείων στη λίστα

Η συμπλήρωση με στοιχεία, τόσο της λίστας όσο και της συνδυασμένης λίστας, μπορεί να γίνει και κατά τη διάρκεια του σχεδιασμού της διεπαφής και κατά τη διάρκεια εκτέλεσης του προγράμματος. Κατά το σχεδιασμό, εισάγουμε τα στοιχεία δίνοντας τιμές στην

ιδιότητα **List**. Το ένα στοιχείο πληκτρολογείται μετά το άλλο και ο διαχωρισμός τους γίνεται πατώντας σε συνδυασμό τα πλήκτρα Ctrl + Enter.

Η προσθήκη στοιχείων μέσα από τον κώδικα του προγράμματος γίνεται με εκτέλεση της μεθόδου AddItem (πρόσθεσε στοιχείο) για κάθε ένα στοιχείο που θέλουμε να εισάγουμε. Η σύνταξη της μεθόδου είναι:

*αντικείμενο*.AddItem *στοιχείο, δείκτης*

όπου *αντικείμενο* είναι το όνομα της λίστας ή της συνδυασμένης λίστας, *στοιχείο* είναι μια συμβολοσειρά με το κείμενο που θέλουμε να προσθέσουμε ως γραμμή στη λίστα και δείκτης είναι ένας ακέραιος που υποδεικνύει τη θέση της λίστας στην οποία θα εισαχθεί το νέο στοιχείο. Αν δεν ορίσουμε *δείκτη*, το *στοιχείο* εισάγεται στο τέλος της λίστας, αν η ιδιότητα **Sorted** (λίστα ταξινομημένη) έχει την τιμή False. Αν η ιδιότητα **Sorted** της λίστας έχει την τιμή True, το στοιχείο τοποθετείται αυτόματα στην κατάλληλη θέση, ώστε τα στοιχεία της λίστας να είναι και να παρουσιάζονται ταξινομημένα.

Προσπέλαση λίστας

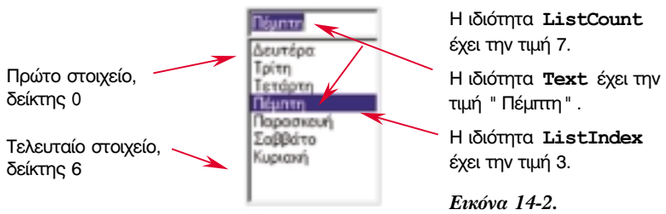
Η ιδιότητα **NewIndex** (νέος δείκτης) επιστρέφει το δείκτη του τελευταίου στοιχείου που έχει εισαχθεί στη λίστα και είναι χρήσιμη όταν δουλεύουμε με ταξινομημένες λίστες. Όταν εισάγουμε ένα στοιχείο σε ταξινομημένη λίστα, εξορισμού η VB εισάγει το στοιχείο με αλφαβητική σειρά. Η ιδιότητα **NewIndex** μας δίνει τη θέση του νεοεισαχθέντος στοιχείου και μπορούμε να χρησιμοποιήσουμε την τιμή της για να το επιλέξουμε.

Για να υποδείξει ο χρήστης ένα στοιχείο της λίστας κάνει κλικ επάνω του. Τότε το στοιχείο τονίζεται με μπλε χρώμα και η ιδιότητα **Text** της λίστας λαμβάνει ως τιμή το κείμενο του επιλεγμένου στοιχείου. Μέσα από το πρόγραμμα μπορούμε να προσπελάσουμε τα στοιχεία μιας λίστας ή μιας συνδυασμένης λίστας χρησιμοποιώντας την ιδιότητα **ListIndex**. Η ιδιότητα **ListIndex** θέτει ή επιστρέφει το δείκτη του επιλεγμένου στοιχείου. Αν δεν έχουμε επιλέξει στοιχείο, η ιδιότητα **ListIndex** έχει τιμή ίση με -1. Το πρώτο στοιχείο της λίστας έχει δείκτη 0. Το πλήθος των στοιχείων της λίστας το δίνει η τιμή της ιδιότητας **ListCount**. Επομένως, ο δείκτης του τελευταίου στοιχείου της λίστας ισούται με **ListCount** - 1. Τέλος, η ιδιότητα **ItemData**(*δείκτης*) έχει για τιμή τη συμβολοσειρά του στοιχείου της λίστας που υποδεικνύει ο δείκτης.

**Παράδειγμα 14-1.**

Σε μια συνδυασμένη λίστα γράφουμε τις ημέρες της εβδομάδας αρχίζοντας από τη Δευτέρα. Κάνουμε κλικ στην Πέμπτη. Οι ιδιότητες που περιγράψαμε έχουν τις τιμές:

Πρόκειται για πίνακα ιδιοτήτων. Δείτε μάθημα 20.




*Εικόνα 14-2.*

Αν η λίστα έχει το όνομα Days οι τιμές των ιδιοτήτων **ItemData** είναι:

```
Days.ItemData(0) < "Äåõôÿñá"
Days.ItemData(1) < "Ôñßôç"
          :
Days.ItemData(6) < "Êõñéáêþ"
```

Διαγραφή στοιχείου λίστας

Για να απομακρύνουμε ένα επιλεγμένο στοιχείο από τη λίστα χρησιμοποιούμε τη μέθοδο RemoveItem, η οποία συντάσσεται ως εξής:

*αντικείμενο*.RemoveItem *δείκτης*

Η μέθοδος Clear διαγράφει όλα τα στοιχεία της λίστας.

Αξιοποιώντας την ιδιότητα **ListIndex** μπορούμε να υποδείξουμε στη μέθοδο RemoveItem, το στοιχείο της λίστας που έχει επιλεγεί από τον χρήστη.

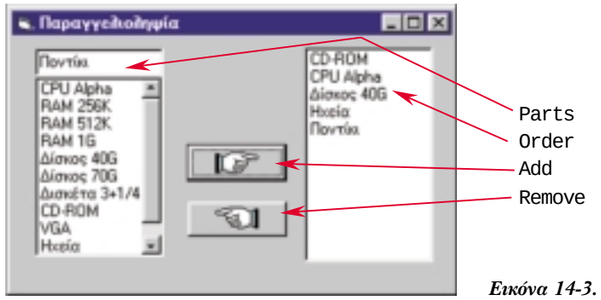


**Άσκηση 14-1.**

Σε μια φόρμα να τοποθετηθεί μια συνδυασμένη λίστα με όνομα Parts και μια λίστα με όνομα Order. Επίσης, να τοποθετηθούν και δύο πλήκτρα, με όνομα Add και Remove. Στη συνδυασμένη λίστα να υπάρχουν καταχωρισμένα τα εξαρτήματα από τα οποία μπορεί να συντεθεί ένας ηλεκτρονικός υπολογιστής.

Να γραφεί πρόγραμμα με το οποίο ο χρήστης να μπορεί να διαλέξει υλικά από τη συνδυασμένη λίστα Parts, ή να μπορεί να πληκτρολογήσει την περιγραφή ενός υλικού που δεν υπάρχει στη λίστα και όταν πατήσει το πλήκτρο Add, το υλικό να αντιγράφεται στη λίστα Order. Σε περίπτωση που ο χρήστης κάνει λάθος, να έχει τη δυνατότητα, αφού υποδείξει το υλικό στη λίστα Order, να το αφαιρεί, πατώντας το πλήκτρο Remove.

1. Τοποθετούμε το αντικείμενο Parts πάνω στη φόρμα. Δίνουμε στην ιδιότητα Style, της συνδυασμένης λίστας την τιμή **1-Simple Combo**. Η συνδυασμένη λίστα θα φαίνεται με ανεπτυγμένο το περιεχόμενο της και θα προσφέρει τη δυνατότητα να πληκτρολογούμε στην περιοχή του πλαισίου κειμένου ακόμα και εξαρτήματα τα οποία δεν υπάρχουν στη λίστα.
2. Τοποθετούμε το αντικείμενο Order πάνω στη φόρμα. Δίνουμε στην ιδιότητα **Sorted** την τιμή True, ώστε κάθε στοιχείο που τοποθετείται στη λίστα να ταξινομείται αυτόματα στη σωστή θέση.
3. Τοποθετούμε τα πλήκτρα διαταγής, τους αφαιρούμε την ιδιότητα **Caption**, δίνουμε στην ιδιότητα **Style** την τιμή **1-Graphical** και τοποθετούμε το κατάλληλο εικονίδιο πάνω στα πλήκτρα ενημερώνοντας την ιδιότητα Picture.




*Εικόνα 14-3.*

4. Εισάγουμε το πρόγραμμα στο παράθυρο κώδικα.

```
Private Sub Form_Load()
  'ÁíçìÝñùóç ôçò ëßóôáò ìå ôéò ðáñéãñáöÝò ôùí åîáñôçìÜôùí
   Parts.AddItem "CPU Alpha"
   Parts.AddItem "RAM 256K"
   Parts.AddItem "RAM 512K"
   Parts.AddItem "RAM 1G"
   Parts.AddItem "Äßóêïò 40G"
   Parts.AddItem "Äßóêïò 70G"
   Parts.AddItem "ÄéóêÝôá 3+1/4"
   Parts.AddItem "CD-ROM"
   Parts.AddItem "VGA"
   Parts.AddItem "Ç÷åßá"
   Parts.AddItem "Ðëçêôñïëüãéï"
End Sub

Private Sub Add_Click()
   Order.AddItem Parts.Text
End Sub

Private Sub Delete_Click()
   If Order.ListIndex > -1 Then
      Order.RemoveItem Order.ListIndex
   End If
End Sub
```

Ο κώδικας για το συμβάν **Load** της φόρμας χρησιμοποιεί τη μέθοδο AddItem για να συμπληρώσει τη λίστα. Στο προηγούμενο μάθημα είχαμε αναφέρει ότι συνήθως τις αρχικοποιήσεις μέσα στο πρόγραμμα τις πραγματοποιούμε τη στιγμή της εκτέλεσης της υπορουτίνας διαχείρισης του συμβάντος Load.

## Πλήκτρο σημείωσης και πλήκτρο επιλογής

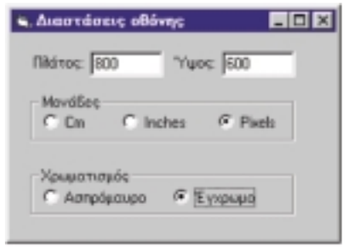
Το πλήκτρο σημείωσης (CheckBox) και το πλήκτρο επιλογής (OptionButton) λειτουργούν όπως ακριβώς και ένας διακόπτης. Ο χρήστης κάνοντας κλικ επάνω τους επιλέγει ανάμεσα σε δύο καταστάσεις αληθή-ψευδή ή on-off. Η διαφορά τους είναι, ότι κατά την εκτέλεση της εφαρμογής σε μια ομάδα πλήκτρων σημείωσης μπορούν να είναι ταυτόχρονα επιλεγμένα δύο ή περισσότερα πλήκτρα, ενώ σε μια ομάδα πλήκτρων επιλογής μόνο το ένα από αυτά μπορεί να είναι επιλεγμένο κάθε φορά.

Η ιδιότητα **Caption** αυτών των αντικειμένων μας δίνει τη δυνατότητα να ορίσουμε το κείμενο που θα εμφανιστεί δίπλα σε ένα τέτοιου είδους πλήκτρο. Η ιδιότητα **Alignment** προσδιορίζει αν το κείμενο θα βρίσκεται στα δεξιά του αντικειμένου ή στα αριστερά του. Η ιδιότητα **Value** για το πλήκτρο επιλογής μπορεί να έχει τιμή True ή False, ανάλογα με το αν το πλήκτρο έχει επιλεγεί ή όχι. Για το πλήκτρο σημείωσης η ιδιότητα Value μπορεί να πάρει τις τιμές των εσωτερικών σταθερών vbUnchecked (μη επιλεγμένο, με τιμή 0), vbChecked (επιλεγμένο, με τιμή 1) ή vbGrayed (μη διαθέσιμο, με τιμή 2).

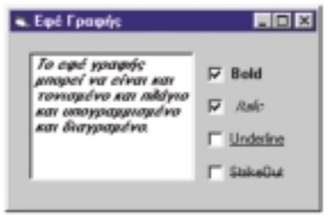
Συνήθως τα πλήκτρα σημείωσης και τα πλήκτρα επιλογής τοποθετούνται σε μια φόρμα ομαδοποιημένα. Για να τα ομαδοποιήσουμε, είτε για λόγους εμφάνισης είτε για λειτουργικούς λόγους, μπορούμε να τα συμπεριλάβουμε μέσα σε ένα πλαίσιο (Frame) ή μέσα σε ένα πλαίσιο εικόνας (PictureBox). Σ' αυτήν την περίπτωση, λέμε ότι ανήκουν σε έναν ***φορέα (container)***. Για να δημιουργήσουμε μια ομάδα πλήκτρων, σχεδιάζουμε πρώτα το φορέα και στη συνέχεια τοποθετούμε μέσα στον φορέα τα πλήκτρα σημείωσης και τα πλήκτρα επιλογής. Αν τα πλήκτρα προϋπάρχουν δεν μπορούμε να τα σύρουμε μέσα στο φορέα. Μπορούμε βέβαια να τα τοποθετήσουμε κατευθείαν πάνω στη φόρμα. Σε αυτήν την περίπτωση λειτουργούν σαν μια και μόνον ομάδα.

**Παράδειγμα 14-2.**

Στο παράθυρο της εικόνας 14-4 υπάρχουν δύο πλαίσια, το πλαίσιο με **Caption** "Ìïíüääò" και το πλαίσιο με **Caption** "×ñùìáôéóìüò". Από το πρώτο πλαίσιο ο χρήστης μπορεί να επιλέξει τις μονάδες μέτρησης ενός παραθύρου και από το δεύτερο πλαίσιο μπορεί να επιλέξει το αν το παράθυρο θα παρουσιάζει ασπρόμαυρο ή έγχρωμο το περιεχόμενό του.




*Εικόνα 14-4.*




*Εικόνα 14-5.*

Στο παράθυρο της εικόνας 14-5 υπάρχουν τέσσερα πλήκτρα σημείωσης. Παρατηρήστε ότι μαρκάροντας ένα από αυτά δεν αποκλείεται ένα άλλο, όπως συμβαίνει με τα πλήκτρα επιλογής.

## Ράβδοι κύλισης

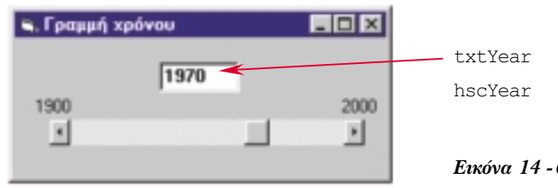
Οι ράβδοι κύλισης (scrollbars) χρησιμοποιούνται συνήθως σε συνδυασμό με άλλα αντικείμενα, όπως πλαίσια κειμένου, πλαίσια εικόνας κ.ά. για να διευκολύνουν την ανάγνωση μιας μεγάλης λίστας στοιχείων, για να δείξουν την πρόοδο μιας εργασίας ή για να προσφέρουν έναν εναλλακτικό τρόπο εισαγωγής στοιχείων. Οι ιδιότητες **Min** (ελάχιστο) και **Max** (μέγιστο)



καθορίζουν τα όρια των τιμών που μπορούν να παραχθούν από τη ράβδο κύλισης. Η τιμή **Min** αντιστοιχεί στην τιμή που θέλουμε να παίρνει η ιδιότητα **Value** της ράβδου όταν το ορθογώνιο κύλισης βρίσκεται στην πάνω (για κατακόρυφη ράβδο) ή αριστερή θέση (για οριζόντια ράβδο). Η τιμή **Max** αντιστοιχεί στην τιμή που θέλουμε να παίρνει η ιδιότητα **Value** της ράβδου όταν το πλαίσιο βρίσκεται στην κάτω (για κατακόρυφη ράβδο) ή τη δεξιά θέση (για οριζόντια ράβδο). Η ιδιότητα **Value** δίνει την τρέχουσα τιμή, που αντιστοιχεί στην τρέχουσα θέση του ορθογωνίου κύλισης μέσα στη ράβδο. Η ιδιότητα **SmallChange** καθορίζει τη μεταβολή της **Value**, όταν γίνεται κλικ στα βέλη κύλισης που βρίσκονται στα άκρα της ράβδου, ενώ η ιδιότητα **LargeChange** καθορίζει τη μεταβολή της **Value**, όταν γίνεται κλικ πάνω στη ράβδο, δηλαδή ανάμεσα σε ένα βέλος κύλισης και στο ορθογώνιο κύλισης.

**Άσκηση 14-2.**

Συνδυάστε μια οριζόντια ράβδο κύλισης με ένα πλαίσιο κειμένου, ώστε η τρέχουσα τιμή της ράβδου να εμφανίζεται σ το πλαίσιο κειμένου. Στο πλαίσιο κειμένου θέλουμε να εμφανίζουμε τα έτη από το 1900 ως το 2000 ανά δεκαετία και πενταετία.




*Εικόνα 14-6.*

Στο συμβάν **Load** της φόρμας θα αρχικοποιήσουμε κατάλληλα τις τιμές των ιδιοτήτων της ράβδου, ενώ στο συμβάν **Change** της ράβδου θα αποδίδουμε στο πλαίσιο κειμένου την τρέχουσα τιμή της ράβδου. Γράφουμε τον κώδικα:

```
Private Sub Form_Load()
    hscYear.Min = 1900          'ÅëÜ÷éóôç ôéìÞ
    hscYear.Max = 2000          'ÌÝãéóôç ôéìÞ
    hscYear.SmallChange = 5     'ÄëÜ÷éóôï Üëéá áýîçóçò
    hscYear.LargeChange = 10    'ÌÝãéóôï Üëéá áýîçóçò
End Sub
Private Sub hscYear_Change()
    txtYear = hscYear.Value
End Sub
```

## Αντικείμενα ελέγχου ActiveX

Εκτός από τα τυπικά εργαλεία που αναφέραμε, τα οποία εμφανίζονται απαρχής στην εργαλειοθήκη της VB, υπάρχουν και αρκετά πρόσθετα τα οποία μπορούν να δημιουργήσουν αντικείμενα ελέγχου τύπου ***ActiveX.*** Πρόκειται για εξειδικευμένα εργαλεία, τα οποία αυξάνουν σημαντικά τις δυνατότητες της VB, αφού μπορεί να χρησιμοποιήσει ο χρήστης για να τοποθετήσει πάνω σε μια φόρμα, αντικείμενα ελέγχου, τα οποία εκτελούν πιο σύνθετες λειτουργίες. Μερικά από αυτά εγκαθίστανται με την εγκατάσταση της VB, ενώ άλλα προσφέρονται από τρίτους προμηθευτές και διανέμονται σε ξεχωριστά αρχεία που έχουν κατάληξη .ocx. Για να προσθέσουμε ένα εργαλείο ActiveX στην εργαλειοθήκη:

1. Επιλέγουμε **Project ¦ Components** από τη γραμμή μενού.
2. Επιλέγουμε τον καρτελοδείκτη **Controls.**
3. Από το πινάκιο με τις ομάδες των εργαλείων (δείτε εικόνα 13-8) επιλέγουμε αυτές των οποίων τα εργαλεία θέλουμε να προσθέσουμε στην εργαλειοθήκη.

Μερικά από τα πιο κοινά εργαλεία βρίσκονται στην ομάδα ***Microsoft Windows Common Controls***, και περιγράφονται στη συνέχεια.

***TabStrip (Συστοιχία καρτελών)***: Δημιουργεί μια συστοιχία επικαλυπτόμενων καρτελών, που έχουν έναν καρτελοδείκτη στο πάνω άκρο τους και μπορούν να περιλάβουν διαφορετικά αντικείμενα ελέγχου. Ο χρήστης επιλέγοντας έναν καρτελοδείκτη βλέπει το περιεχόμενο της αντίστοιχης καρτέλας.

Το παράθυρο **Components** (εικόνα 13-8) εισαγωγής νέων εργαλείων περιέχει TabStrip.

***ToolBar (Γραμμή εργαλείων)***: Χρησιμοποιείται για τη δημιουργία γραμμών εργαλείων που περιέχουν ένα σύνολο ανεξάρτητων πλήκτρων. Κάθε ένα πλήκτρο μπορεί να περιέχει εικόνα ή κείμενο. Τα πλήκτρα εκτελούν συνήθως λειτουργίες που είναι δυνατόν να εκτελεστούν και από το μενού της εφαρμογής.

***StatusBar (Γραμμή κατάστασης)***: Χρησιμοποιείται για τη δημιουργία γραμμών κατάστασης στο κάτω μέρος του παραθύρου της εφαρμογής. Οι γραμμές κατάστασης παρέχουν πληροφορίες για τη λειτουργία των εφαρμογών και υποδείξεις για τη σημασία των αντικειμένων που επιλέγει ο χρήστης.

***ProgressBar (Γραμμή προόδου)***: Δημιουργεί ένα ορθογώνιο στο εσωτερικό του οποίου προβάλλεται, με μια σειρά από τετραγωνίδια, η πρόοδος μιας εργασίας. Έτσι, ο χρήστης έχει μια εικόνα του χρόνου που απαιτείται για την πλήρη εκτέλεση της εργασίας.

***TreeView (Δενδροειδής προβολή)***: Χρησιμοποιείται για την παρουσίαση της ιεραρχικής σχέσης μεταξύ των στοιχείων ενός συνόλου, για παράδειγμα για την παρουσίαση των περιεχομένων ενός κειμένου, τη σχέση φακέλων και αρχείων ενός δίσκου κ.ά. Σε κάθε κόμβο μπορεί να περιέχει μια ετικέτα ή ένα γραφικό.

***ImageList (Λίστα εικόνων):*** Δημιουργεί τον αποθηκευτικό χώρο μιας λίστας εικόνων, οι οποίες προβάλλονται στο εσωτερικό άλλων αντικειμένων ελέγχου. Για παράδειγμα, το εργαλείο ToolBar μπορεί να αντλεί τα εικονίδια των πλήκτρων του μέσα από μια λίστα εικόνων.

***ListView (Όψη λίστας)***: Επιτρέπει την πολυμορφική παρουσίαση μιας συλλογής στοιχείων, για παράδειγμα αρχείων και φακέλων, σε μορφή μικρών ή μεγάλων εικονιδίων, επικεφαλίδων ή γραμμών. Το δεξί τμήμα του παραθύρου του Explorer του λειτουργικού συστήματος λειτουργεί σαν αντικείμενο ListView.

***Slider (Δείκτης ολίσθησης)***: Δημιουργεί αντικείμενα παρόμοια με τη ράβδο κύλισης. Ο χρήστης μπορεί να σύρει έναν δείκτη κατά μήκος ενός άξονα και να επιλέξει μια τιμή από ένα σύνολο συνεχόμενων διακριτών τιμών. Στο Control Panel κάποιες από τις ιδιότητες του ποντικιού ρυθμίζονται με αυτόν τον τρόπο.

***ImageCombo (Συνδυασμένη λίστα εικόνων)***: Δημιουργεί μια βελτιωμένη μορφή της συνδυασμένης λίστας. Στο εσωτερικό αυτής της λίστας είναι δυνατόν να τοποθετηθούν εικονίδια δίπλα από κάθε στοιχείο, ώστε ο χρήστης να έχει και μια πιο οικεία σε αυτόν γραφική απεικόνιση.

**Άσκηση 14-3.**

Ένα αρκετά χρήσιμο εργαλείο, το ***όργανο (Gauge)***, με το οποίο είναι δυνατή η παραγωγή αντικειμένων ελέγχου που έχουν τη μορφή οργάνων μέτρησης (θερμομέτρων, ταχυμέτρων, βολτομέτρων κ.ά.), βρίσκεται μέσα στην ομάδα ***MicroHelp MhGauge Control.*** Να δημιουργηθεί εφαρμογή στην οποία ο χρήστης να κινεί το δείκτη ενός Slider (δείκτη ολίσθησης) και να μετατοπίζεται αναλόγως και ταυτόχρονα η βελόνα σε ένα όργανο μέτρησης.

1. Εισάγουμε στην εργαλειοθήκη τις ομάδες εργαλείων Microsoft Windows Common Controls και MicroHelp MhGauge Control.
2. Τοποθετούμε πάνω σε μια φόρμα ένα αντικείμενο οργάνου με όνομα `Gauge` και ένα αντικείμενο δείκτη ολίσθησης με όνομα `Slider`.
3. Στις ιδιότητες **`Min`** και **`Max`** των αντικειμένων `Slider` και `Gauge` δίνουμε τις τιμές 0 και 10 αντίστοιχα για να έχουν και τα δύο αντικείμενα την ίδια κλίμακα τιμών.
4. Για την ιδιότητα **`Picture`** του οργάνου επιλέγουμε το αρχείο `Circlock.bmp` που βρίσκεται στον φάκελο `C:\...\Vb\Graphics\Bitmaps\Gauge\`.


*Εικόνα 14-7*



5. Στο αντικείμενο `Gauge` και στην ιδιότητα **`NeedleWidth`** (πάχος βελόνας), εκχωρούμε την τιμή 3, ώστε η βελόνα του οργάνου να έχει κάποιο πάχος, και στην ιδιότητα **`Style`** την τιμή **3-Full,** ώστε η βελόνα να περιστρέφεται περί το κέντρο του οργάνου.
6. Γράφουμε τον κώδικα:

```
Private Sub Slider_Change()
  Gauge.Value = Slider.Value
End Sub
```

Κάθε αλλαγή τιμής (πρόκληση συμβάντος **`Change`**) στο αντικείμενο Slider, προκαλεί και την αντίστοιχη μεταβολή στην τιμή του αντικειμένου Gauge.

## Το αντικειμένο Εκτυπωτής

Το λειτουργικό σύστημα των Windows, διαχειρίζεται τις εκτυπώσεις μέσω του ***διαχειριστή εκτυπώσεων (Printer Manager)***. Ο διαχειριστής εκτυπώσεων δέχεται τις εκτυπώσεις από πολλά διαφορετικά προγράμματα και αναλαμβάνει τη σωστή εκτύπωσή τους. Αν δε μεσολαβούσε ο διαχειριστής εκτυπώσεων, και το κάθε πρόγραμμα έστελνε απευθείας την εκτύπωσή του στον εκτυπωτή, ή θα μπλέκονταν οι εκτυπώσεις, ή τα προγράμματα με μικρές εκτυπώσεις θα ανέμεναν τον τερματισμό προγραμμάτων με μεγάλες εκτυπώσεις που θα είχαν δεσμεύσει πρώτα τον εκτυπωτή. Όταν ένα πρόγραμμα δημιουργεί μια εκτύπωση, τα τμήματά της συγκεντρώνονται από τον διαχειριστή εκτυπώσεων (πρώτη φάση της εκτύπωσης) και όταν συμπληρωθεί όλη η διαδικασία, ο διαχειριστής εκτυπώσεων δεσμεύει τον κατάλληλο εκτυπωτή και την πραγματοποιεί (δεύτερη φάση της εκτύπωσης).

Η VB, μέσω του αντικειμένου `Printer`, μας δίνει τη δυνατότητα να εκμεταλλευτούμε όλα τα πλεονεκτήματα εκτύπωσης του γραφικού περιβάλλοντος των Windows. Έτσι, είναι δυνατή η εκτύπωση κειμένου σε διάφορες γραμματοσειρές και στυλ (τονισμένο, με πλάγιους χαρακτήρες κ.ά.), η εκτύπωση γραφικών και η συνδυασμένη εκτύπωση κειμένου και γραφικών. Η εκτύπωση κειμένου στην VB γίνεται με τη μέθοδο `Print`, που ήδη έχουμε γνωρίσει στα μαθήματα 9 και 11 και η εκτύπωση γραφικών με τις μεθόδους γραφικών `Pset, Line, Circle` που θα γνωρίσουμε στο μάθημα 15.

Αν μέσα στο πρόγραμμα δεν υπάρχει εντολή που να ζητά την εκτέλεση της μεθόδου `EndDoc` η εκτύπωση πραγματοποιείται με τον τερματισμό του προγράμματος.

Ένα σύνολο μεθόδων εκτύπωσης τερματίζεται πάντα με μια εντολή που ζητά την εκτέλεση της μεθόδου `EndDoc` (τέλος του εγγράφου). Η μέθοδος `EndDoc` ειδοποιεί τον διαχειριστή εκτυπώσεων ότι το πρόγραμμα έχει ολοκληρώσει την πρώτη φάση της εκτύπωσης και ότι μπορεί να προχωρήσει στη φάση αποστολής στον εκτυπωτή των προς εκτύπωση δεδομένων.

Το πλήθος των παραμέτρων της μεθόδου `Print` δεν είναι καθορισμένο. Δεξιά από την κωδική λέξη `Print` είναι δυνατόν να θέσουμε όσες παραμέτρους θέλουμε. Οι τιμές τους θα τυπωθούν στην ίδια γραμμή. Αν δεν χωρούν, η εκτύπωση συνεχίζεται στην επόμενη γραμμή. Ο διαχωρισμός τους δεν είναι υποχρεωτικό να γίνει με κόμματα (,), όπως γίνεται συνήθως σε όλες τις μεθόδους. Είναι δυνατόν να γίνει με κενά διαστήματα ή το σύμβολο semicolon (; η αγγλική άνω τελεία). Τα σύμβολα αυτά δεν παίζουν μόνο διαχωριστικό ρόλο μεταξύ των παραμέτρων, αλλά επιδρούν στη μορφή της εκτύπωσης.

**Παράδειγμα 14-3.**

α) Οι εντολές:

```
Printer.Print "ÄÕÁÄÉÊÏÓ"; "ÄÅÊÁÄÉÊÏÓ"; "ÄÅÊÁÅÎÁÄÉÊÏÓ"
Printer.Print "10101101"; -173; Hex(97)
Printer.EndDoc
```

θα τυπώσουν:

```
ÄÕÁÄÉÊÏÓÄÅÊÁÄÉÊÏÓÄÅÊÁÅÎÁÄÉÊÏÓ
10101101-173 61
```

Μεταξύ των παραμέτρων, το semicolon ή το κενό διάστημα δε δημιουργεί κανένα διαχωρισμό μεταξύ των εκτυπούμενων τιμών. Γι' αυτό και η εκτύπωση των συμβολοσειρών της πρώτης γραμμής φαίνεται συνεχόμενη. Στη δεύτερη γραμμή, υπάρχει μεταξύ των

δύο τελευταίων αριθμών ένα κενό διάστημα, διότι ο δεύτερος αριθμός είναι θετικός και στους θετικούς αριθμούς παραλείπεται το πρόσημο.

β) Οι εντολές:

```
Printer.Print "ÄÕÁÄÉÊÏÓ", "ÄÅÊÁÄÉÊÏÓ", "ÄÅÊÁÅÎÁÄÉÊÏÓ"
Printer.Print "10101101",-173, Hex(97)
Printer.EndDoc
```

θα τυπώσουν:

```
ÄÕÁÄÉÊÏÓ      ÄÅÊÁÄÉÊÏÓ      ÄÅÊÁÅÎÁÄÉÊÏÓ
10101101      -173            61
```

Το κόμμα ως διαχωριστικό οριοθετεί, την αρχή των περιοχών που έχουν μέγεθος 14 χαρακτήρες.

Αν θέλουμε να αφήσουμε κενή μια περιοχή, εισάγουμε μεταξύ των αντίστοιχων παραμέτρων δύο συνεχόμενα κόμματα. Ας σημειωθεί ότι αν, μετά από όλες τις παραμέτρους, υπάρχει κόμμα ή semicolon στο τέλος, η επόμενη μέθοδος `Print` θα τυπώσει στην ίδια σειρά.

Όπως ήδη έχουμε αναφέρει μπορούμε να καθορίσουμε τη γραμματοσειρά του κειμένου και το στυλ των γραμμάτων που θα χρησιμοποιηθούν. Ο καθορισμός γίνεται από ιδιότητες του αντικειμένου `Printer`. Συγκεκριμένα, η ιδιότητα:

**`FontName`**: Γραμματοσειρά. Η τιμή της ιδιότητας είναι μια συμβολοσειρά. Το όνομα της γραμματοσειράς είναι αυτό ακριβώς που αναγνωρίζουν τα Windows. Για παράδειγμα, "Times New Roman", "Courier New", "Wingdings" κ.ά.

**`FontSize`**: Μέγεθος. Το μέγεθος της γραμματοσειράς πρέπει να καθορίζεται σε στιγμές (points) που είναι το 1/72 της ίντσας.

**`FontBold`**: Τονισμός. Αν η τιμή της ιδιότητας είναι `True`, τα γράμματα τυπώνονται τονισμένα.

**`FontItalic`**: Πλάγια γράμματα. Αν η τιμή της είναι `True`, τα γράμματα τυπώνονται με κλίση.

**`FontUndeline`**: Υπογράμμιση. Αν η τιμή της της ιδιότητας είναι `True`, τα γράμματα τυπώνονται υπογραμμισμένα.

**`FontStrikethru`**: Διαγραφή. Αν η τιμή της της ιδιότητας είναι `True`, τα γράμματα τυπώνονται με μια γραμμή στο μέσον τους σαν να έχουν σβηστεί.

**Παράδειγμα 14-4.**

Το τμήμα προγράμματος:

```
Printer.FontName = "Times New Roman"
Printer.FontSize = 11
Printer.FontBold = False:Printer.FontItalic = False
Printer.Print "Normal"
Printer.FontBold = True:  Printer.FontItalic = False
Printer.Print "Bold"
Printer.FontBold = False:       Printer.FontItalic = True
Printer.Print "Italic"
Printer.FontBold = True:  Printer.FontItalic = True
Printer.Print "Bold Italic"
Printer.EndDoc
```

θα προκαλέσει την εκτύπωση:

Normal
**Bold**
*Italic*
***Bold Italic***

Στην περίπτωση που θέλουμε να γίνει αλλαγή σελίδας στον εκτυπωτή, χρησιμοποιούμε τη μέθοδο `NewPage`. Σε όποιο σημείο κι αν βρίσκεται η εκτύπωση, η σελίδα που έχει γραφτεί προωθείται στην έξοδο και οι μέθοδοι `Print` που θα εκτελεστούν στη συνέχεια, εκτυπώνουν στην επόμενη σελίδα.



Αν θέλουμε να κάνουμε εκτύπωση σε μια συγκεκριμένη θέση μιας σελίδας χρησιμοποιούμε τις ιδιότητες **`CurrentX`** και **`CurrentY`**. Οι τιμές αυτών των ιδιοτήτων ορίζουν σε twip τη x και τη y συντεταγμένη αντίστοιχα, του σημείου από το οποίο θα αρχίσει μια εκτύπωση.

Για να ακυρώσουμε μια εκτύπωση χρησιμοποιούμε τη μέθοδο `KillDoc`. Αυτή η μέθοδος ακυρώνει την εκτύπωση, ακόμα κι αν ένα μέρος της έχει υλοποιηθεί από τον διαχειριστή εκτυπώσεων.

## Ανακεφαλαίωση

Η λίστα είναι αντικείμενο που δεσμεύει το χρήστη να επιλέξει μέσα από έναν προκαθορισμένο σύνολο επιλογών. Με τη συνδυασμένη λίστα δημιουργούμε μια ανοιχτή λίστα, δηλαδή μια λίστα την οποία να μπορεί να ενημερώσει ο χρήστης. Το πλήκτρο σημείωσης και το πλήκτρο επιλογής λειτουργούν όπως ακριβώς και ένας διακόπτης. Από μια ομάδα πλήκτρων σημείωσης μπορούν να είναι ταυτόχρονα επιλεγμένα δύο ή περισσότερα πλήκτρα, ενώ απόμια ομάδα πλήκτρων επιλογής μόνο το ένα επιτρέπεται να είναι επιλεγμένο κάθε φορά. Οι ράβδοι κύλισης χρησιμοποιούνται σε συνδυασμό με άλλα αντικείμενα όπως πλαίσια κειμένου, πλαίσια εικόνας κ.ά. για να διευκολύνουν την ανάγνωση μιας μεγάλης λίστας στοιχείων, για να δείχνουν την πρόοδο μιας εργασίας ή για να προσφέρουν έναν εναλλακτικό τρόπο εισαγωγής στοιχείων.

Τα ActiveX είναι πρόσθετα εργαλεία με τα οποία δημιουργούνται αντικείμενα ελέγχου, που εκτελούν εξειδικευμένες λειτουργίες. Διατίθενται από τρίτους προμηθευτές και διανέμονται σε αρχεία τύπου `.ocx`.Η VB, μέσω του αντικειμένου `Printer`, δίνει τη δυνατότητα εκμετάλλευσης όλων των πλεονεκτημάτων εκτύπωσης του γραφικού περιβάλλοντος των Windows. Έτσι, είναι δυνατή η εκτύπωση κειμένου σε διάφορες γραμματοσειρές και στυλ, η εκτύπωση γραφικών και η συνδυασμένη εκτύπωση κειμένου και γραφικών.

## Εργαστηριακές Ασκήσεις

1. Σε μια φόρμα τοποθετήστε μια λίστα και μια συνδυασμένη λίστα. Σε κατάσταση σχεδίασης συμπληρώστε τα στοιχεία τους με τις τιμές που έχουν τα αντίστοιχα αντικείμενα της εικόνας 14-1.
2. Να τοποθετηθεί μια λίστα σε μια φόρμα, η οποία να συμπληρώνεται με τις τυπικές τιμές των αντιστάσεων ανοχής 5% που κυκλοφορούν στην αγορά από 1Ω έως 1ΜΩ.

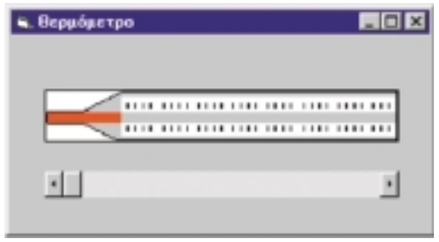
   Υπόδειξη: Χρησιμοποιήστε τον πίνακα 2-1 του μαθήματος 2.
3. Να εκτελεστεί η άσκηση 14-1. Να συμπληρωθεί η φόρμα, ώστε να γίνεται διαγραφή όλων των στοιχείων της λίστας `Order` με μέθοδο `Clear`.
4. Να εκτελεστεί η άσκηση 14-2.
5. Να δημιουργηθεί φόρμα στην οποία να τοποθετηθεί μια κατακόρυφη και μια οριζόντια ράβδος κύλισης. Όταν ο χρήστης κινεί το ορθογώνιο κύλισης κάθε ράβδου, ένα τετράγωνο που έχει σχεδιαστεί με το εργαλείο "γεωμετρικό σχήμα" να αλλάζει θέση και να τοποθετείται στις αντίστοιχες συντεταγμένες.


*Εικόνα 14-8.*

Υπόδειξη: Οι ιδιότητες **`Min`** και **`Max`** της κατακόρυφης ράβδου να υπολογιστούν από τις ιδιότητές της **`Top`** και **`Height`** του γεωμετρικού σχήματος και οι ιδιότητες **`Min`** και **`Max`** της οριζόντιας ράβδου να υπολογιστούν από τις ιδιότητές της **`Left`** και **`Width`**.

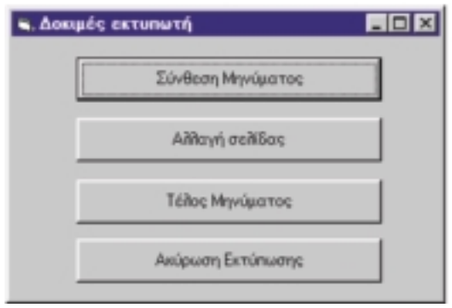
6. Να εκτελεστεί η άσκηση 14-3.

7. Στον κατάλογο `C:\…\Vb\Graphics\Bitmaps\Gauge\` υπάρχει αρχείο με όνομα `Horz.bmp`. Το αρχείο αυτό περιέχει την εικόνα ενός θερμομέτρου. Να δημιουργηθεί πρόγραμμα, στο οποίο ο χρήστης να ολισθαίνει το ορθογώνιο κύλισης μιας ράβδου κύλισης και να κινείται ο υδράργυρος του θερμομέτρου.




*Εικόνα 14-9.*

8. Σε μια οθόνη τοποθετήστε τέσσερα πλήκτρα. Επισυνάψτε κώδικα στις υπορουτίνες διαχείρισης συμβάντων των πλήκτρων, ώστε το ένα πλήκτρο να τυπώνει στον εκτυπωτή ένα μήνυμα, το δεύτερο να ζητά την αλλαγή σελίδας, το τρίτο να τερματίζει μια εκτύπωση και το τέταρτο να τη διακόπτει. Πραγματοποιήστε δοκιμές. Σχολιάστε το πότε ακριβώς αντιδρά ο εκτυπωτής μετά την απαίτηση εκτέλεσης της αντίστοιχης μεθόδου.




*Εικόνα 14-10.*

## Σημειώσεις:



# Μάθημα 15

# Γραφικά

Τελειώνοντας αυτό το μάθημα οι μαθητές θα μπορούν:

- Να χρησιμοποιούν τα εργαλεία γραφικών για να σχεδιάζουν απλά γεωμετρικά σχήματα πάνω σε μια φόρμα.
- Να περιγράφουν τον τρόπο διαχείρισης του χρώματος στη σχεδίαση των βασικών γεωμετρικών αντικειμένων.
- Να περιγράφουν τις διαφορές μεταξύ του εργαλείου "πλαίσιο εικόνας" και του εργαλείου "εικόνα".
- Να προγραμματίζουν τη δημιουργία γραφικών κάνοντας κλήση μεθόδων σχεδίασης γραφικών.
- Να καθορίζουν τις κλίμακες σχεδίασης

Στους υπολογιστές οι εικόνες, τα εικονίδια, τα σχήματα, τα σχέδια και τα διαγράμματα χαρακτηρίζονται με τον όρο ***γραφικά (graphics)***. Τα γραφικά αποτελούν απαραίτητο στοιχείο κάθε σύγχρονης εφαρμογής, ανεξάρτητα από το αν η εικόνα αποτελεί ή όχι το θεματικό της αντικείμενο. Από τη φύση του ο άνθρωπος μπορεί να αντλήσει πολύ πιο γρήγορα και εύκολα τις πληροφορίες που απεικονίζονται με γραφικό τρόπο, από ότι τις πληροφορίες που περιέχονται σε ένα κείμενο ή σε έναν πίνακα αριθμών.

Η VB διαθέτει δύο διαφορετικούς τρόπους για τη δημιουργία γραφικών. Στον έναν τρόπο χρησιμοποιούνται τα εργαλεία γραφικών της παλέτας εργαλείων ενώ στον άλλο χρησιμοποιούνται οι μέθοδοι γραφικών της γλώσσας προγραμματισμού. Το ποιος από τους δύο τρόπους θα χρησιμοποιηθεί, εξαρτάται από το τι ακριβώς θέλουμε να πετύχουμε. Κάποια απλά γραφικά (γραμμές, ορθογώνια, κύκλους, ελλείψεις κλπ) μπορούμε να τα "ζωγραφίσουμε" εξαρχής πάνω σε μια φόρμα κατά τη δημιουργία της διεπαφής, χρησιμοποιώντας τα εργαλεία γραφικών. Έτσι, χωρίς να γράψουμε καθόλου κώδικα βλέπουμε το τελικό οπτικό αποτέλεσμα. Κάποια άλλα γραφικά, τα οποία είναι πιο σύνθετα και μπορούν να περιγραφούν μόνον από αλγορίθμους ή για τα οποία υπάρχει η απαίτηση να εμφανιστούν κατά τη διάρκεια εκτέλεσης του προγράμματος ως εφέ, επιβάλλεται να δημιουργηθούν από μεθόδους γραφικών, που καλούνται μέσα από κώδικα προγράμματος.

Γραφικά μπορούμε να δημιουργήσουμε μέσα σε μια φόρμα ή μέσα σε ένα πλαίσιο εικόνας χρησιμοποιώντας τα εργαλεία ή/και τις μεθόδους γραφικών. Επίσης, μπορούμε να εκτυπώσουμε γραφικά στον εκτυπωτή, αξιοποιώντας τις μεθόδους γραφικών που υποστηρίζει το ειδικό αντικείμενο `Printer`.

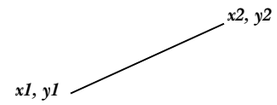
## Εργαλεία γραφικών

Όπως έχουμε ήδη δει στο μάθημα 4, τα εργαλεία γραφικών του περιβάλλοντος εργασίας της VB είναι: το εργαλείο "γραμμή" (line), το εργαλείο "γεωμετρικό σχήμα" (shape), το εργαλείο "εικόνα" (image) και το εργαλείο "πλαίσιο εικόνας" (picture box).

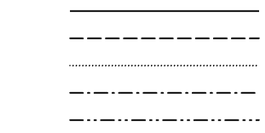
Με το εργαλείο "γραμμή" σχεδιάζουμε ευθύγραμμα τμήματα. Χαρακτηριστικές οπτικές ιδιότητες των ευθύγραμμων τμημάτων είναι οι:

**`X1`**, **`Y1`** και **`X2`**, **`Y2`**: Οι συντεταγμένες του σημείου αρχής (x1,y1) και οι συντεταγμένες του σημείου τερματισμού (x2,y2).
**`BorderWidth`**: Το πάχος της γραμμής.
**`BorderStyle`**: Το είδος της γραμμής, όταν το πάχος της έχει τιμή 1, με τιμές:

| | |
|---|---|
| **0 - Transparent** | Διαφανής |
| **1 - Solid** | Συμπαγής |
| **2 - Dash** | Διακεκομμένη |
| **3 - Dot** | Με κουκκίδες |
| **4 - Dash-Dot** | Διακεκομμένη με κουκκίδες |




***Σχήμα 15-1.*** *Οι συντεταγμένες αρχής και τέλους*



***Σχήμα 15 - 2.*** *Είδη γραμμής*

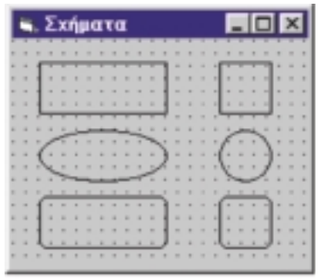
**BorderColor:** Το χρώμα της γραμμής.

**DrawMode:** Ο τρόπος σχεδίασης. Η ιδιότητα αυτή δέχεται 16 διαφορετικές τιμές καικαθορίζει τον τρόπο, με τον οποίο εμφανίζονται και αλληλεπιδρούν οι γραμμές, και γενικάόλα τα γραφικά, που δημιουργούνται πάνω στην περιοχή σχεδίασης, με αυτά που έχουν σχεδιαστεί ήδη. Έτσι, η εξορισμού τιμή της **DrawMode (13-Copy Pen (Default))** σχεδιάζει πάνω από τα γραφικά που υπάρχουν ήδη, με το χρώμα που ορίζει η τιμή της ιδιότητας **ForeColor** του αντικειμένου που περιγράφει την περιοχή σχεδίασης, η τιμή **6-Invert** αντιστρέφει το χρώμα, ενώ άλλες τιμές της **DrawMode** συνδυάζουν τα χρώματα με διάφορους τρόπους και επιτρέπουν τη δημιουργία οπτικών εφέ.

Το εργαλείο "γεωμετρικό σχήμα"

Με το εργαλείο "γεωμετρικό σχήμα", ανάλογα με την τιμή της ιδιότητας **Shape**, σχεδιάζουμε ορθογώνια, ή παράγωγα των ορθογωνίων, κύκλους, ελλείψεις κ.ά. Συγκεκριμένα, η ιδιότητα **Shape** μπορεί να πάρει τις τιμές:

| | |
|---|---|
| **0 - Rectangle** | Ορθογώνιο |
| **1 - Square** | Τετράγωνο |
| **2 - Oval** | Έλλειψη |
| **3 - Circle** | Κύκλο |
| **4 - Rounded Rectangle** | Ορθογώνιο με στρογγυλεμένες γωνίες |
| **5 - Rounded Square** | Τετράγωνο με στρογγυλεμένες γωνίες |




***Εικόνα 15 - 1.*** *Γεωμετρικά σχήματα*

Τα γεωμετρικά σχήματα χαρακτηρίζονται από τις οπτικές ιδιότητες **BorderWidth**, **BorderStyle**, **BorderColor**, **DrawMode**, τις οποίες ήδη μελετήσαμε στο εργαλείο "γραμμή". Επιπλέον, τα γεωμετρικά σχήματα έχουν και τις ιδιότητες:

**BackStyle:** Είδος φόντου, που καθορίζει αν το φόντο του εσωτερικού του σχήματος θα είναι γεμισμένο με χρώμα ή διαφανές. Συγκεκριμένα, η ιδιότητα μπορεί να πάρει τις τιμές:

| | |
|---|---|
| **0 - Transparent** | Διαφανές |
| **1 - Opaque** | Γεμισμένο με χρώμα |

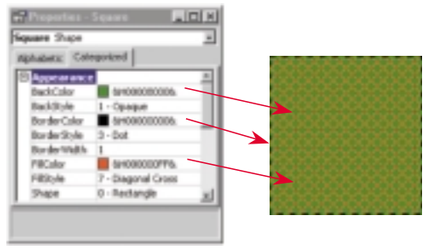
**FillStyle:** Είδος γεμίσματος. Καθορίζει τον τρόπο με τον οποίο θα γραμμοσκιαστεί το εσωτερικό του σχήματος.

| | |
|---|---|
| **0 - Solid** | Συμπαγές |
| **1 - Transparent** | Διαφανές |
| **2 - Horizontal Line** | Με οριζόντιες γραμμές |
| **3 - Vertical Line** | Με κατακόρυφες γραμμές |
| **4 - Upward Diagonal** | Με διαγώνιες γραμμές προς τα πάνω |
| **5 - Downward Diagonal** | Με διαγώνιες γραμμές προς τα κάτω |
| **6 - Cross** | Με σταυρωτές γραμμές |
| **7 - Diagonal Cross** | Με διαγώνιες σταυρωτές γραμμές |

**BackColor**, **FillColor**: Χρώμα φόντου και χρώμα γραμμοσκίασης, αντίστοιχα.

**Παράδειγμα 15-1.**

Σε ένα γεωμετρικό σχήμα μπορούμε να καθορίσουμε διαφορετικό χρώμα για το περίγραμμα, το χρώμα γραμμοσκίασης και το χρώμα του εσωτερικού φόντου.



***Εικόνα 15-2.*** *Το παράθυρο ιδιοτήτων ενός ορθογωνίου, στο οποίο διακρίνονται οι οπτικές του ιδιότητες και οι τιμές τους.*

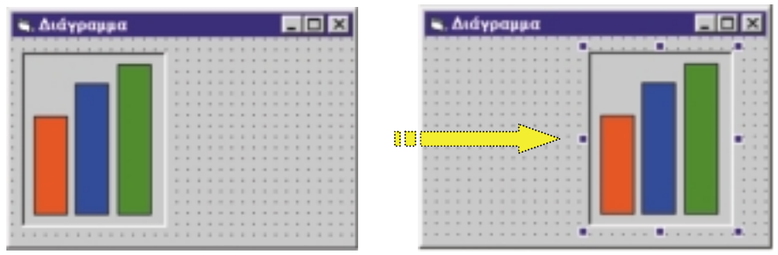


Τα εργαλεία "εικόνα" και "πλαίσιο εικόνας"

Με το εργαλείο "εικόνα" σχεδιάζουμε ορθογώνιες περιοχές, στις οποίες είναι δυνατόν να τοποθετηθούν εικόνες που περιέχονται μέσα σε αρχεία. Με το εργαλείο "πλαίσιο εικόνας" σχεδιάζουμε ορθογώνιες περιοχές, στις οποίες είναι δυνατόν να τοποθετηθούν εικόνες που περιέχονται μέσα σε αρχεία, αλλά επιπλέον μπορούν να τοποθετηθούν και άλλα γραφικά αντικείμενα. Το αντικείμενο πλαίσιο εικόνας είναι πολύ πιο ισχυρό αντικείμενο από ότι το αντικείμενο εικόνας. Έχει πολλές κοινές ιδιότητες με τις φόρμες και είναι δυνατή η τοποθέτηση στο εσωτερικό του γραμμών και γεωμετρικών σχημάτων. Το πλαίσιο εικόνας δρα σαν ***φορέας (container)*** των αντικειμένων που θα τοποθετηθούν στο εσωτερικό του. Τα συμπαρασέρνει κατά τη μετακίνησή του. Θα μπορούσε να πει κανείς ότι συμπεριφέρεται σαν υποπαράθυρο μέσα στο παράθυρο μιας φόρμας.

**Άσκηση 15-1.**

Στο εσωτερικό ενός πλαισίου εικόνας τοποθετούμε τρία ορθογώνια. Αν μετακινήσουμε το πλαίσιο εικόνας μέσα στη φόρμα θα συμπαρασύρει και τα τρία ορθογώνια.




**(α)** **(β)**

***Εικόνα 15-3.*** *(α) Αρχική θέση πλαισίου εικόνας*
*(β) Το πλαίσιο εικόνας στη νέα του θέση συμπαρασύρει το περιεχόμενό του*

Αντίθετα, ένα αντικείμενο εικόνας δεν μπορεί να συμπεριφερθεί σαν φορέας άλλων αντικειμένων.

Τα αντικείμενα πλαισίου εικόνας δε διαθέτουν την ιδιότητα **Strech**, την οποία διαθέτουν τα αντικείμενα εικόνας, την οποία περιγράψαμε στην άσκηση 5-3. Αντίθετα, διαθέτουν την ιδιότητα **AutoSize**. Η ιδιότητα αυτή μπορεί να καθορίσει το μέγεθος του πλαισίου εικόνας ανάλογα με το μέγεθος της εικόνας που θα υποδεχθεί. Αν η τιμή της ιδιότητας **AutoSize** είναι True γίνεται προσαρμογή του μεγέθους του πλαισίου εικόνας στις διαστάσεις στις εικόνας που τοποθετείται στο εσωτερικό του. Αντίθετα, αν η τιμή αυτής της ιδιότητας είναι False, το πλαίσιο εικόνας διατηρεί τις διαστάσεις του και η εικόνα, ή φαίνεται αποκομμένη στο εσωτερικό του ή καταλαμβάνει ένα τμήμα του ανάλογα με το αν έχει μεγαλύτερες ή μικρότερες διαστάσεις, αντίστοιχα.

Ένα αντικείμενο εικόνας ή πλαισίου εικόνας είναι δυνατόν να υποδεχθεί εικόνες που περιέχονται σε αρχεία μορφοποίησης:

**.bmp** ή **.dib** ***Window bitmap.*** Τα αρχεία αυτά περιέχουν ***ψηφιογραφικά γραφικά (bitmap graphics).*** Αυτός ο τρόπος μορφοποίησης δημιουργήθηκε για να αποθηκεύει αρχεία του προγράμματος Paint των Windows.

**.gif** ***Graphic Interchange Format***. Τα αρχεία αυτά περιέχουν συμπιεσμένα ψηφιογραφικά γραφικά με βάθος χρώματος 8 bits. Η μορφοποίηση αυτή αναπτύχθηκε από την Compuserve για τη μετάδοση εικόνων στο Internet.

**.jpg** ***Joint Photographic expert Group.*** Αυτής της μορφοποίησης τα αρχεία περιέχουν ψηφιογραφικά γραφικά σε συμπιεσμένη μορφή.

**.wmf** ή **.emf** ***Windows metafiles.*** Πρόκειται για αρχεία που περιέχουν ***διανυσματικά γραφικά (vector graphics)***. Υπάρχει όμως η δυνατότητα να αποθηκεύσουν και ψηφιογραφικά γραφικά.

**.ico** ***Icons.*** Τα αρχεία αυτ περιέχουν εικονίδια διαστάσεων 32x32 εικονοστοιχείων (pixels).

**.cur** ***Cursors.*** Πρόκειται για αρχεία τα οποία περιέχουν εικονίδια που μπορεί να χρησιμοποιήσει για μορφή ο δρομέας του ποντικιού.

Ψηφιογραφικά είναι τα γραφικά που συντίθενται κουκκίδα κουκκίδα όπως ακριβώς και τα ψηφιδωτά. Διανυσματικά είναι τα γραφικά που περιγράφονται από συντομογραφίες μαθηματικών προτάσεων και σχέσεων απλών γραμμών.
Εικονοστοιχεία είναι οι κουκκίδες από τις οποίες αποτελείται ένα ψηφιογραφικό γραφικό ή το πλέγμα απεικόνισης της οθόνης.

## Η σχεδίαση με μεθόδους γραφικών

Η σχεδίαση με μεθόδους γραφικών μπορεί να γίνει στο εσωτερικό μιας φόρμας, στο εσωτερικό ενός πλαισίου εικόνας, στο χαρτί του εκτυπωτή. Οι πιο απλές μέθοδοι γραφικών είναι οι ακόλουθες:

**Pset:** αποδίζει το χρώμα που ορίζουμε σε συγκεκριμένο σημείο (point) του αντικειμένου στο οποίο αναφέρεται (φόρμα, πλαίσιο εικόνας, εκτυπωτής). Η σύνταξή της είναι:

*αντικείμενο*.Pset Step *(x, y), χρώμα*

**Line:** σχεδιάζει γραμμές και ορθογώνια σχήματα στο εσωτερικό του αντικειμένου στο οποίο αναφέρεται (φόρμα, πλαίσιο εικόνας, εκτυπωτής). Η σύνταξή της είναι:

*αντικείμενο*.Line Step *(x1, y1)* - Step *(x2, y2), χρώμα*, BF

**Circle:** σχεδιάζει καμπυλόγραμμα σχήματα (κύκλους, ελλείψεις, τόξα) στο εσωτερικό του αντικειμένου στο οποίο αναφέρεται (φόρμα, πλαίσιο εικόνας, εκτυπωτής). Η σύνταξή της

*αντικείμενο*.Circle Step *(x, y), ακτίνα, χρώμα, αρχή, τέλος, λόγος_μορφής*

**Cls:** καθαρίζει όλα τα γραφικά και το κείμενο, τα οποία έχουν δημιουργηθεί πάνω σε μια φόρμα ή ένα πλαίσιο εικόνας με μεθόδους σχεδίασης κατά την εκτέλεση της εφαρμογής. Η σύνταξή της είναι:

*αντικείμενο*.Cls

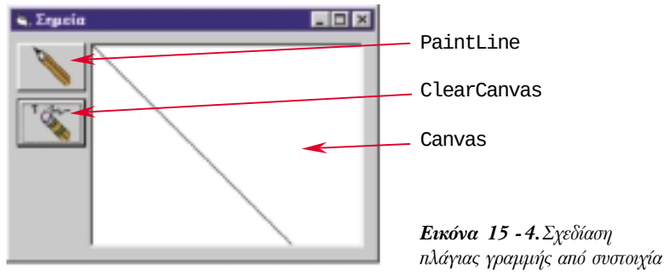
Σε όλες τις μεθόδους, η αναφορά του ονόματος του αντικειμένου σχεδίασης είναι προαιρετική. Αν απουσιάζει το αντικείμενο, η μέθοδος αναφέρεται στη φόρμα, στην οποία υπάρχει ο αντίστοιχος κώδικας. Επίσης, προαιρετικές είναι και οι αναφορές κάποιων από τις παραμέτρους, για παράδειγμα της παραμέτρου *χρώμα, λόγος_μορφής* κ.ά. Όταν έχει παραλειφθεί μια παράμετρος, ενώ δεν παραλείπονται οι παράμετροι που την ακολουθούν, πρέπει να γίνεται δέσμευση της θέσης της με ένα κόμμα. Τέλος, προαιρετική είναι και η χρήση της κωδικής λέξης Step (βήμα). Αν υπάρχει η κωδική λέξη Step η σχεδίαση αρχίζει με συντεταγμένες σχετικές ως προς το σημείο που έχει γίνει η τελευταία σχεδίαση.

## Σχεδίαση σημείων

Για να σχεδιάσουμε ένα σημείο σε ένα αντικείμενο πρέπει να αναφέρουμε υποχρεωτικά τις συντεταγμένες του *(x, y)*. Η δήλωση του χρώματος, όπως έχει αναφερθεί, είναι προαιρετική. Αν η παράμετρος χρώμα δεν είναι συμπληρωμένη, χρησιμοποιείται ως χρώμα αυτό που έχει δηλωθεί στην ιδιότητα ForeColor του αντικειμένου (φόρμας, πλαισίου εικόνας, εκτυπωτή) στο εσωτερικό του οποίου γίνεται η σχεδίαση.

**Παράδειγμα 15-2.**

Σε μια φόρμα τοποθετούμε ένα πλαίσιο εικόνας με το όνομα Canvas και δύο πλήκτρα. Το ένα πλήκτρο, με το όνομα PaintLine, σχεδιάζει μια γραμμή. Το άλλο πλήκτρο, με το όνομα ClearCanva, διαγράφει το περιεχόμενο του πλαισίου εικόνας.




***Εικόνα 15 - 4.*** *Σχεδίαση πλάγιας γραμμής από συστοιχία*

Ο κώδικας για το πλήκτρο ClearCanva είναι:

```
Private Sub ClearCanvas_Click()
    Canvas.Cls
End Sub
```



Ο κώδικας για το πλήκτρο PaintLine μπορεί να γραφεί ως:

```
Private Sub PaintLine_Click()
    Dim i As Long
    For i = 0 To 5000
        Canvas.PSet (i, i), vbBlack
    Next
End Sub
```

ή με τη χρήση της κωδικής λέξης Step στη μέθοδο Pset ως:

```
Private Sub PaintLine_Click()
  Dim i As Long
   Canvas.PSet (0, 0)     'Áñ÷éêÞ èÝóç
   For i = 1 To 5000
          Canvas.PSet Step (1, 1), vbBlack
   Next
End Sub
```

Οι γραμμές δε γίνονται με την παράθεση μερών αλλά με τη συνεχή κίνηση των σημείων.

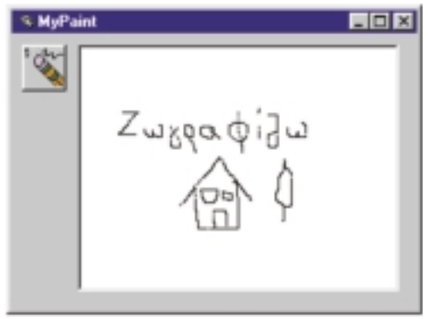
*Νεύτων*

Και μόνο με τη μέθοδο Pset μπορούμε να δημιουργήσουμε όλες τις υπορουτίνες σχεδίασης γραμμών και σχημάτων και να σχεδιάσουμε οποιαδήποτε εικόνα σε ένα αντικείμενο σχεδίασης.

**Άσκηση 15-2.**

Όλοι έχουν χρησιμοποιήσει το πρόγραμμα της Ζωγραφικής (Paint ή Paintbrush) των Windows, λίγοι όμως γνωρίζουν πόσο εύκολα μπορεί να δομηθεί από απλές υπορουτίνες που περιέχουν μεθόδους γραφικών. Σε αυτήν την άσκηση θα δημιουργήσουμε το τμήμα του προγράμματος Ζωγραφική, το οποίο σχεδιάζει ελεύθερο σχέδιο με τη βοήθεια του ποντικιού.

1. Σε μια φόρμα δημιουργούμε μια περιοχή σχεδίασης με το εργαλείο "πλαίσιο εικόνας" και της δίνουμε το όνομα Canvas. Επίσης, δημιουργούμε και ένα πλήκτρο με το όνομα ClearCanvas για να καθαρίζουμε την περιοχή σχεδίασης.




***Εικόνα 15 - 5.*** *Το πρόγραμμα ζωγραφική. Η ζωγραφική με το ποντίκι δεν είναι και τόσο εύκολη.*

2. Καθορίζουμε τη μορφή του δρομέα του ποντικιού στο πλαίσιο εικόνας, από τις ιδιότητες **MousePointer** και **MouseIcon**. Δίνουμε την τιμή **99 - Custom** στην ιδιότητα **MousePointer** και επιλέγουμε μια μορφή ποντικιού σε σχήμα μολυβιού από τον φάκελο C:\….\Graphics\Icons\Writing.
3. Γράφουμε τον κώδικα:

```
Option Explicit
Dim PenDown As Boolean
Private Sub Form_Load()
    PenDown = False
End Sub
Private Sub Canvas_MouseDown(Button As Integer, _
                             Shift As Integer, X As Single, Y As Single)
    PenDown = True
End Sub
```

```
Private Sub Canvas_MouseUp(Button As Integer, _
                           Shift As Integer, X As Single, Y As Single)
    PenDown = False
End Sub

Private Sub Canvas_MouseMove(Button As Integer, _
                             Shift As Integer, X As Single, Y As Single)
    If PenDown Then Canvas.PSet (X, Y)
End Sub
```

Η μεταβλητή PenDown χρησιμοποιείται ως ***σημαφόρος (semaphor)*** μεταξύ των υπορουτινών. Με τη φόρτωση της φόρμας, στην PenDown δίνεται η τιμή ή False, από την υπορουτίνα Form_Load. Έτσι, όταν το ποντίκι εισέλθει στην περιοχή Canvas δεν αφήνει ίχνος, επειδή η υπορουτίνα Canvas_MouseMove δεν εκτελεί τη μέθοδο Pset που βρίσκεται στο εσωτερικό της δομής επιλογής. Όμως, όταν πατηθεί το πλήκτρο του ποντικιού μέσα στην περιοχή Canvas, η υπορουτίνα Canvas_MouseDown δίνει στην PenDown την τιμή True. Τότε, και για όσο διάστημα το πλήκτρο του ποντικιού είναι πατημένο, με την κίνηση του ποντικιού η Canvas_MouseMove σχεδιάζει σημεία στην τρέχουσα θέση του δρομέα του ποντικιού. Όταν απελευθερωθεί το πλήκτρο του ποντικιού, η Canvas_MouseUp δίνει στην PenDown εκνέου την τιμή False, οπότε πάλι το ποντίκι δεν θα αφήνει ίχνος.

## Σχεδίαση ευθύγραμμων τμημάτων, τετραγώνων και ορθογώνιων

Για να σχεδιάσουμε ένα ευθύγραμμο τμήμα σε ένα αντικείμενο σχεδίασης καθορίζουμε, προαιρετικά τις συντεταγμένες (x1,y1) του σημείου αρχής και υποχρεωτικά τις συντεταγμένες (x2,y2) του σημείου τερματισμού. Αν παραλείψουμε το σημείο αρχής, η VB θεωρεί ότι αυτό είναι το τρέχον σημείο γραφής επί του αντικειμένου σχεδίασης, που προσδιορίζεται μέσω των ιδιοτήτων του **CurrentX** και **CurrentY** (τρέχουσα συντεταγμένη x και τρέχουσα συντεταγμένη y) ή από την τελευταία μέθοδο γραφικών, που έχουμε χρησιμοποιήσει. Διαφορετικά, η σχεδίαση της γραμμής ξεκινάει από την πάνω αριστερή γωνία του αντικειμένου.

Όπως και με τη σχεδίαση σημείων, έτσι και με τη σχεδίαση γραμμών, αν η παράμετρος χρώμα δεν είναι συμπληρωμένη, χρησιμοποιείται ως χρώμα αυτό που έχει δηλωθεί στην ιδιότητα **ForeColor** του αντικειμένου (φόρμας, πλαισίου εικόνας, εκτυπωτή) στο εσωτερικό του οποίου γίνεται η σχεδίαση. Επίσης, το πάχος της γραμμής καθορίζεται από την ιδιότητα **DrawWidth** του αντικειμένου σχεδίασης, το είδος της γραμμής από την ιδιότητα του **DrawStyle** και ο τρόπος σχεδίασης από την ιδιότητα **DrawMode**.

Από το παράθυρο επισκόπισης αντικειμένων (Object Browser) μπορούμε να δούμε τις εσωτερικές σταθερές που έχουν οριστεί για τις τιμές των ιδιοτήτων **DrawWidth**, **DrawStyle**, **DrawMode**.

Αν χρησιμοποιήσουμε την κωδική λέξη **Step** του σημείου αρχής, δηλώνουμε ότι οι συντεταγμένες του ορίζονται σε σχέση με την τρέχουσα θέση **(CurrentX, CurrentY)**. Αν χρησιμοποιήσουμε την κωδική λέξη **Step** του σημείου τερματισμού, δηλώνουμε ότι οι συντεταγμένες του ορίζονται σε σχέση με το σημείο αρχής.

**Παράδειγμα 15-3.**

Οι εντολές:

```
Line (500, 0)-(500, 500), vbBlue
Line (0, 500)-(500, 500), vbRed
```

σχεδιάζουν στη φόρμα (αφού δεν αναφέρεται όνομα αντικειμένου) μια οριζόντια μπλε γραμμή και μια κατακόρυφη κόκκινη γραμμή αντίστοιχα. Αν η εντολή:

```
Line -Step(500, 500), vbGreen
```

είναι η πρώτη εντολή σχεδίασης, σχεδιάζει μια πλάγια πράσινη γραμμή στη φόρμα, αρχίζοντας από το πάνω αριστερό σημείο της. Την ίδια λειτουργία πραγματοποιούν και οι εντολές:

```
Me.ForeColor=vbGreen    'Êáèüñéóå ÷ñþìá
PSet (0, 0)             'Óçìåßïí áñ÷Þò
Line -Step(500, 500)    'Ãñáììþ ìÝ÷ñé ôï óçìåßïí (500,500)
```



Για να σχεδιάσουμε ένα ορθογώνιο σχήμα σε ένα αντικείμενο ακολουθούμε τα βήματα που αφορούν το σχεδιασμό γραμμής και συμπληρώνουμε τη σύνταξη με την παράμετρο B (Box).

**Παράδειγμα 15-4.**

Η εντολή:

*αντικείμενο*.Line (x1, y1) - (x2, y2), , B

είναι ισοδύναμη με το σύνολο των εντολών:

*αντικείμενο*.Line (x1, y1) - (x2, y1)
*αντικείμενο*.Line (x2, y1) - (x2, y2)
*αντικείμενο*.Line (x2, y2) - (x1, y2)
*αντικείμενο*.Line (x1, y2) - (x1, y1)

Μαζί με την παράμετρο B μπορούμε να χρησιμοποιήσουμε, προαιρετικά, και την παράμετρο F (Fill). Σε αυτήν την περίπτωση το εσωτερικό του ορθογώνιου χρωματίζεται με το ίδιο χρώμα που χρησιμοποιείται για τις πλευρές.

**Παράδειγμα 15-5.**

Η εντολή:

```
Line (2000, 500)-Step(500, 500), , B
```

σχεδιάζει χρησιμοποιώντας το χρώμα, που καθορίζει η ιδιότητα **ForeColor** της φόρμας, ένα τετράγωνο του οποίου η μια κορυφή βρίσκεται στο σημείο (2000, 500) και το οποίο έχει μήκος πλευράς 500. Για να σχεδιαστεί το ίδιο τετράγωνο με χρωματισμένο το εσωτερικό του, η εντολή πρέπει να γραφτεί ως εξής:

```
Line (2000, 500)-Step(500, 500), , BF
```

**Άσκηση 15-3.**

Στα πακέτα λογισμικού δημιουργίας γραφικών υπάρχει μια λειτουργία για τη σχεδίαση ευθύγραμμων τμημάτων. Χρησιμοποιώντας αυτήν τη λειτουργία ο χρήστης, κάνει κλικ στο σημείο της αρχής μέσα στην περιοχή σχεδίασης, σύρει το ποντίκι προς το σημείο τερματισμού και εκεί το αφήνει. Όσο διάστημα ο χρήστης κρατά πατημένο το πλήκτρο του ποντικιού στην περιοχή σχεδίασης επανασχεδιάζεται το ευθύγραμμο τμήμα με κάθε αλλαγή της θέσης του ποντικιού. Στην πραγματικότητα δε σχεδιάζεται ένα μόνο ευθύγραμμο τμήμα αλλά δύο. Σε κάθε κίνηση του ποντικιού σβήνεται το ίχνος του ευθύγραμμου τμήματος και σχεδιάζεται νέο ευθύγραμμο τμήμα. Η διαγραφή του παλαιού ευθύγραμμου τμήματος γίνεται με επανασχεδιασμό του. Για να λειτουργήσει σωστά η διαγραφή του ευθύγραμμου τμήματος και ο επανασχεδιασμός πρέπει η σχεδιάση να γίνει με τον ειδικό τρόπο DrawMode **10 - Not Xor Pen.** Πρόκειται για έναν τρόπο σχεδίασης, στον οποίο όταν σε ένα εικονοστοιχείο ζητείται να αποκτήσει το χρώμα, το οποίο ήδη έχει, το εικονοστοιχείο επανακτά το χρώμα του φόντου. Για παράδειγμα, αν το εικονοστοιχείο είναι μαύρο σε γκρι φόντο και ζητηθεί να επανασχεδιαστεί με μαύρο χρώμα, αντί να διατηρήσει το χρώμα του γίνεται γκρι. Οι υπορουτίνες που πραγματοποιούν τη δυναμική σχεδίαση του ευθύγραμμου τμήματος είναι:


*Εικόνα 15-6.*

```
Option Explicit
Dim PenDown    As Boolean
Dim x1, y1     As Single
Dim x2, y2     As Single

Private Sub Form_Load()
    DrawMode = vbNotXorPen  'Áéäáéêüò ôñüðïò ó÷åäßáóçò ãñáììþí
End Sub
```

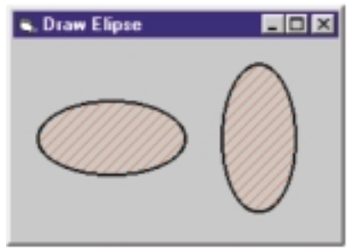
```
Private Sub Form_MouseDown(Button As Integer, _
                           Shift As Integer, X As Single, Y As Single)
    PenDown = True                'Ôï ðïíôßêé åßíáé ðáôçìÝíï
    x1 = X:     y1 = Y            'Óõíôåôáãìýíåò áñ÷Þò
    x2 = X:     y2 = Y            'Óõíôåôáãìýíåò ôýëïõò
End Sub
Private Sub Form_MouseMove(Button As Integer, _
                           Shift As Integer, X As Single, Y As Single)
    If PenDown Then
        Line (x1, y1)-(x2, y2)  'ÄéáãñáöÞ ðñïçãïýìåíïõ ß÷íïõò
        Line (x1, y1)-(X, Y)    'Äçìéïõñãßá íÝïõ ß÷íïõò
    End If
    x2 = X:     y2 = Y            'ÍÝåò óõíôåôáãìýíåò ôýëïõò
End Sub
Private Sub Form_MouseUp(Button As Integer, _
                         Shift As Integer, X As Single, Y As Single)
    PenDown = False
End Sub
```

## Σχεδίαση κύκλων, ελλείψεων, τόξων και τομέων

Για τη σχεδίαση ενός κύκλου πρέπει να καθορίσουμε, υποχρεωτικά, τις συντεταγμένες (x,y) του κέντρου του και το μήκος της ακτίνας του. Αν χρησιμοποιήσουμε την κωδική λέξη `Step` του σημείου αρχής, δηλώνουμε ότι οι συντεταγμένες του ορίζονται σε σχέση με την τρέχουσα θέση (**`CurrentX`**, **`CurrentY`**).

Στη γεωμετρία για τη σχεδίαση μιας έλλειψης χρειάζεται η αναφορά του κέντρου συμμετρίας της και τα μήκη του μεγάλου και του μικρού άξονά της. Στη μέθοδο `Circle` η παράμετρος ακτίνα αντιστοιχεί στο μεγάλο άξονα. Το μήκος της ακτίνας του μικρού άξονα υπολογίζεται διαιρώντας την ακτίνα με το *λόγο_μορφής (**aspect ratio**)*. Αν ο *λόγος_μορφής* έχει τιμή μικρότερη του 1, η *ακτίνα* είναι η ακτίνα του x-άξονα και η έλλειψη εκτείνεται κατά μήκος του οριζόντιου άξονα, ενώ αν ο λόγος έχει τιμή μεγαλύτερη ή ίση με 1 η ακτίνα είναι η ακτίνα του y-άξονα και η έλλειψη εκτείνεται κατά μήκος του κατακόρυφου άξονα. Αν δεν υπάρχει η κωδική λέξη `Step`, το ζευγάρι (x, y) είναι οι απόλυτες συντεταγμένες του άξονα συμμετρίας, ενώ αν υπάρχει είναι οι σχετικές του συντεταγμένες.

**Παράδειγμα 15-6.**




*Εικόνα 15-7. Δείγματα λόγου μορφής*

Η εντολή:

```
Circle (3000, 1000), 750
```

σχεδιάζει έναν κύκλο με κέντρο το σημείο (3000, 1000) και ακτίνα 750. Επίσης, οι εντολές:

```
'Ëüãïò < 1
Circle (1000, 1000), 750, , , , 1/2
'Ëüãïò > 1
Circle (2500, 1000), 750, , , , 2
```

σχεδίαζουν την αριστερή και τη δεξιά έλλειψη, αντίστοιχα, της εικόνας 15-7.

Με τη μέθοδο `Circle` μπορούμε να σχεδιάσουμε τόξα και τομείς κύκλων ή ελλείψεων. Σε αυτήν την περίπτωση πρέπει να συμπληρώσουμε τις παραμέτρους *αρχή* και *τέλος* που καθορίζουν τη γωνία από την οποία θα αρχίσει και τη γωνία στην οποία θα τελειώσει, αντίστοιχα, η σχεδίαση της έλλειψης ή του κύκλου. Οι γωνίες πρέπει να είναι εκφρασμένες σε ακτίνια και οι τιμές τους πρέπει να είναι μεταξύ -2π και 2π. Η μέτρηση των γωνιών αρχίζει από το δεξιότερο σημείο του κύκλου και η φορά διαγραφής τους είναι αντίθετη της φοράς κίνησης των δεικτών του ρολογιού. Αυτό έχει κάποια δόση ασυμφωνίας με τα μαθηματικά, μια και το σημείο (0,0) βρίσκεται στην πάνω αριστερη γωνία, αλλά βολεύει γιατί με αυτή τη φορά έχουμε συνηθίσει να προσανατολιζόμαστε στον τριγωνομετρικό κύκλο.



**Παράδειγμα 15-7.**

Το τόξο που σχεδιάζεται με την εντολή:

```
Circle (3500, 2500), 1200, , 3.141593, 0
```

είναι το κάτω ημικύκλιο, ενώ το τόξο που σχεδιάζεται με την εντολή:

```
Circle (3500, 2500), 1200, , 0, 3.141593
```

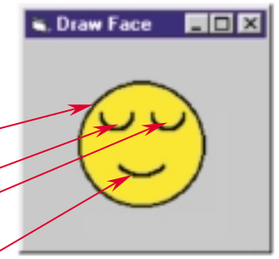
είναι το πάνω ημικύκλιο.

Αν η τιμή ενός άκρου του τόξου είναι αρνητική, σχεδιάζεται και η ακτίνα που συνδέει το άκρο με το κέντρο. Αν και τα δυο άκρα έχουν αρνητικές τιμές, σχεδιάζονται και οι δυο ακτίνες και το σχήμα που προκύπτει είναι ένας κυκλικός ή ένας ελλειπτικός τομέας.

**Παράδειγμα 15-8**.

Το τμήμα προγράμματος:

```
Dim ð, r       As Single
ð = 4 * Atn(1)
r = 600
FillColor = vbYellow
FillStyle = vbSolid
Circle (1250,1300), r
Circle (1000,1000), r/4, , ð, 0
Circle (1500,1000), r/4, , ð, 0
Circle (1250,1300), r/2, , 5/4*ð, 7/4*ð
```




*Εικόνα 15-8.*

σχεδιάζει ένα συνεσταλμένο χαρούμενο προσωπάκι.

Επίσης, το τμήμα προγράμματος:

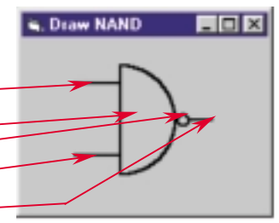
```
Dim ð, r      As Single
ð = 4 * Atn(1)
r = 600
FillStyle = vbSolid
FillColor = vbYellow
Circle (1500, 1000), r, , -ð/8, -15*ð/8
FillColor = vbWhite
Circle (1500, 1000 - r/3), r/10
```


*Εικόνα 15-9.*

σχεδιάζει το γνωστό "πραγματευτή", τον packman.

Τέλος, το τμήμα προγράμματος:

```
Dim ð, r      As Single
Line (1000, 600)-(1500, 600)
Circle (1500, 1000), r, , -3*ð/2, -ð/2
Circle (2190, 1200 - r/3), r/10
Line (1000, 1400)-(1500, 1400)
Line (2250, 1000)-(2500, 1000)
```




*Εικόνα 15-10.*

σχεδιάζει μια πύλη NAND.

## Διαχείριση χρώματος

Στο μάθημα 4 είδαμε ότι τα χρώματα μπορούμε να τα αναπαραστήσουμε ως δεκαεξαδικούς αριθμούς. Επίσης, στο μάθημα 6 είδαμε ότι υπάρχουν κάποιες εσωτερικές σταθερές που αντιστοιχούν στα χρώματα. Για τον ακριβή καθορισμό του χρώματος η VB διαθέτει δύο συναρτήσεις, την RGB και την `QBColor`.

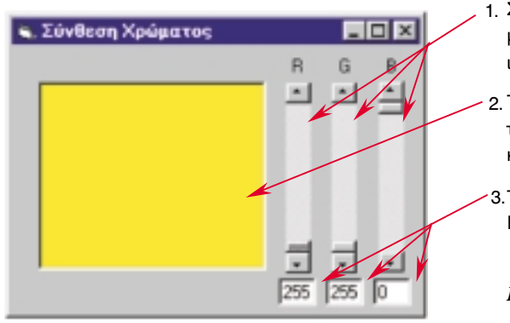
### Η συνάρτηση `RGB (red, green, blue)`

Το όνομα της συνάρτησης παράγεται από τα αρχικά των λέξεων `Red` (κόκκινο), `Green` (πράσινο) και `Blue` (μπλε). Απαιτεί τρία ορίσματα με τιμές από 0-255, που καθορίζουν τη συμμετοχή των χρωμάτων κόκκινου, πράσινου και μπλε στη δημιουργία του επιθυμητού

χρώματος. Το κάτω όριο 0 υποδηλώνει την έλλειψη του αντίστοιχου χρώματος, ενώ το πάνω όριο 255 υποδηλώνει τη μέγιστη δυνατή συμμετοχή του. Η συνάρτηση επιστρέφει ένα μεγάλο ακέραιο ο οποίος αντιπροσωπεύει την τιμή του χρώματος.

**Άσκηση 15-4.**

Σε πολλά προγράμματα γραφικών και επεξεργασίας εικόνας (π.χ. CorelDraw, PhotoShop), ο χρήστης μπορεί να καθορίσει από ένα παράθυρο και με τη βοήθεια ράβδων κύλισης τους χρωματικούς τόνους. Θα δημιουργήσουμε μια παρόμοια φόρμα καθορισμού του χρώματος για μια εφαρμογή.




1. Σε μια φόρμα τοποθετούμε τρεις ράβδους κύλισης με ονόματα Red, Green και Blue και δίνουμε στην ιδιότητα **Max** την τιμή 255.
2. Τοποθετούμε ένα πλαίσιο εικόνας με όνομα Sample του οποίου το χρώμα υποβάθρου θα αλλάζει σε κάθε αλλαγή της τιμής των ράβδων κύλισης.
3. Τοποθετούμε και τρία πλαίσια κειμένου με ονόματα RedIndi, GreenIndi και BlueIndi

***Εικόνα 15-11.*** *Φόρμα επιλογής χρώματος*

Στο παράθυρο κώδικα γράφουμε τις υπορουτίνες:

```
Private Sub Blue_Change()
      Sample.BackColor = RGB(Red, Green, Blue)
      BlueIndi = Blue
End Sub
Private Sub Green_Change()
      Sample.BackColor = RGB(Red, Green, Blue)
      GreenIndi = Green
End Sub
Private Sub Red_Change()
      Sample.BackColor = RGB(Red, Green, Blue)
      RedIndi = Red
End Sub
```

## Η συνάρτηση QBColor (χρώμα)

Η συνάρτηση αυτή δέχεται ως παράμετρο έναν ακέραιο από 0 έως 15, ο οποίος αντιστοιχεί σε ένα από τα 16 χρώματα που χρησιμοποιούσε η QuickBasic (πρόγονος της VB) και επιστρέφει τον αντίστοιχο κωδικό του RGB χρώματος. Συγκεκριμένα, οι τιμές που δέχεται στην παράμετρο χρώμα η QBColor και τα αντίστοιχα χρώματα είναι:

| | |
|---|---|
| **0** - Μαύρο | **8** - Γκρι |
| **1** - Μπλε | **9** - Ανοιχτό μπλε |
| **2** - Πράσινο | **10** - Ανοιχτό πράσινο |
| **3** - Κυανό | **11** - Ανοιχτό κυανό |
| **4** - Κόκκινο | **12** - Ανοιχτό κόκκινο |
| **5** - Μωβ | **13** - Ανοιχτό μωβ |
| **6** - Κίτρινο | **14** - Ανοιχτό κίτρινο |
| **7** - Άσπρο | **15** - Έντονο άσπρο |

## Η μέθοδος Point

Η μέθοδος Point επιστρέφει την RGB τιμή του χρώματος ενός συγκεκριμένου σημείου (point). Η γενική μορφή της μεθόδου είναι:

*αντικείμενο*.Point*(x, y)*

Αν το σημείο που καθορίζουν οι συντεταγμένες x, y βρίσκεται εκτός του αντικειμένου, η μέθοδος επιστρέφει την τιμή -1

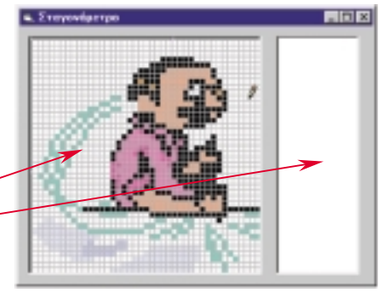


**Παράδειγμα 15-9.**

Η υπορουτίνα:

```
Private Sub Canvas_MouseDown(Button As Integer, _
                             Shift As Integer, X As Single, Y As Single)
    Sample.BackColor = Canvas.Point(X, Y)
End Sub
```

με κάθε κλικ του ποντικιού πάνω στο πλαίσιο εικόνας Canvas, λαμβάνει δείγμα χρώματος και το παρουσιάζει ως χρώμα υποβάθρου στο πλαίσιο εικόνας Sample.




***Εικόνα 15-12.***

## Οι ιδιότητες AutoRedraw και ClipControls

Κάθε γραφικό που δημιουργείται με χρήση μεθόδων γραφικών στη φόρμα ή στο πλαίσιο εικόνας, αν καλυφθεί προσωρινά από ένα άλλο αντικείμενο θα χαθεί. Το ίδιο συμβαίνει αν ελαχιστοποιήσουμε μια φόρμα που περιέχει γραφικά και στη συνέχεια την επαναφέρουμε στο κανονικό της μέγεθος.

Το πρόβλημα μπορούμε να το λύσουμε με το συμβάν **Paint** και να ξαναδημιουργήσουμε την εικόνα στην οθόνη. Ωστόσο, η VB προσφέρει μια πιο απλή τεχνική. Κάθε φόρμα και πλαίσιο εικόνας διαθέτει την ιδιότητα **AutoRedraw** (αυτόματη επανασχεδίαση) που εξορισμού έχει την τιμή False. Αν στην ιδιότητα αυτή δώσουμε την τιμή True, η VB χειρίζεται μόνη της την επανασχεδίαση της οθόνης. Αυτό το πετυχαίνει αποθηκεύοντας στη μνήμη ένα αντίγραφο. Όταν χρειαστεί να αποκαταστήσει τη φόρμα ανασύρει το αντίγραφο και το επανασχεδιάζει στην οθόνη. Αυτό βέβαια στην περίπτωση μεγάλων και σύνθετων εφαρμογών μπορεί να αποβεί σε βάρος της απόδοσης. Επομένως, για απλές εφαρμογές μπορούμε να αξιοποιούμε την ιδιότητα **AutoRedraw** αλλά για σύνθετες εφαρμογές καλό είναι να γράφουμε τις διαδικασίες σχεδίασης στο συμβάν **Paint.**

Τέλος, τα γραφικά και το κείμενο που τοποθετούνται σε μια φόρμα ή ένα πλαίσιο εικόνας όσο η ιδιότητα **AutoRedraw** έχει τιμή True δεν επηρεάζονται, αν πριν καλέσουμε τη μέθοδο Cls, αποδώσουμε στην ιδιότητα **AutoRedraw** την τιμή False. Αυτό σημαίνει ότι μπορούμε να διατηρήσουμε κείμενο και γραφικά σε μια φόρμα ή πλαίσιο εικόνας, αν χειριστούμε κατάλληλα την ιδιότητά τους **AutoRedraw.**

Μια άλλη σημαντική ιδιότητα της περιοχής σχεδίασης είναι η ιδιότητα **ClipControls.** Η ιδιότητα αυτή, καθορίζει αν οι μέθοδοι γραφικών που χρησιμοποιούνται στις διαδικασίες συμβάντος **Paint** θα επανασχεδιάσουν ολόκληρο το αντικείμενο ή μόνον τις πρόσφατα σχεδιασμένες περιοχές. Ο τύπος της ιδιότητας είναι boolean και εξορισμού έχει την τιμή True, που σημαίνει ότι επανασχεδιάζεται ολόκληρο το αντικείμενο.

## Σύστημα συντεταγμένων

Για να δημιουργήσουμε γραφικά χρειάζεται να προσδιορίσουμε σημεία στην οθόνη, στη φόρμα ή σε ένα πλαίσιο εικόνας. Για το λόγο αυτό χρειαζόμαστε ένα σύστημα συντεταγμένων. Το εξορισμού σύστημα που προσφέρει η VB και προσδιορίζει τη θέση των αντικειμένων πάνω στην οθόνη, τη φόρμα ή το πλαίσιο εικόνας, δίνει τις συντεταγμένες (0,0) στην επάνω αριστερή γωνία τους και αυξάνει τις τιμές στον οριζόντιο άξονα κατά την προς τα δεξιά μετακίνηση, και στον κατακόρυφο κατά την προς τα κάτω μετακίνηση. Οι μονάδες μέτρησης που χρησιμοποιούνται

κατά μήκος των δυο αξόνων καθορίζουν και την κλίμακα σχεδίασης και η προκαθορισμένη κλίμακα είναι σε twips. Η VB μπορεί να χρησιμοποιήσει και άλλες κλίμακες, ανάλογα με τις ανάγκες μας, αρκεί να το προκαθορίσουμε μέσω της ιδιότητας του αντικειμένου σχεδίασης **ScaleMode**. Οι τιμές που μπορούμε να δώσουμε στην ιδιότητα **ScaleMode** είναι:

| | |
|---|---|
| **0 - User** | Ορίζεται απο το χρήστη |
| **1 - Twip** | Twips (20 ανά στιγμή, 1440 ανά ίντσα, 567 ανά εκατοστό) |
| **2 - Point** | Στιγμές (72 ανά ίντσα) |
| **3 - Pixel** | Εικονοστοιχεία |
| **4 - Character** | Χαρακτήρες (κατακόρυφα 6 ανά ίντσα, οριζόντια 12 ανά ίντσα) |
| **5 - Inch** | ΄Ιντσες |
| **6 - Millimeter** | Χιλιοστά |
| **7-Centimeter** | Εκατοστά |

Τέλος, μπορούμε να αλλάξουμε τόσο το σημείο εκκίνησης των αξόνων όσο και τη διεύθυνσή τους - για παράδειγμα, να ορίσουμε τις συντεταγμένες για την πάνω αριστερή γωνία σε (3,-4). Προς τούτο χρησιμοποιούμε τις ιδιότητες **ScaleHeight**, **ScaleWidth**, **ScaleLeft** και **ScaleTop**. Αν αποδώσουμε τιμή σε μια τουλάχιστον από αυτές, δημιουργούμε δικό μας σύστημα συντεταγμένων για το αντικείμενο στο οποίο αναφερόμαστε, ενώ η τιμή της **ScaleMode** γίνεται αυτόματα 0.

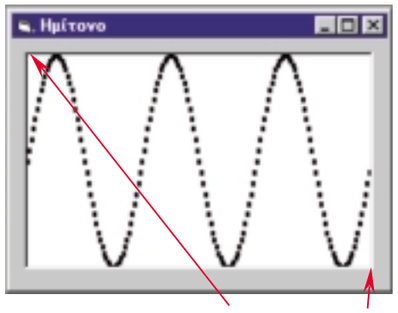
Οι ιδιότητες **ScaleHeight** (ύψος κλίμακας) και **ScaleWidth** (πλάτος κλίμακας) επιστρέφουν ή θέτουν τιμή στην κατακόρυφη ή οριζόντια, αντίστοιχα, διάσταση της περιοχής σχεδίασης. Οι ιδιότητες **ScaleLeft** (αριστερό κλίμακας) και **ScaleTop** (κορυφή κλίμακας) αποδίδουν τιμή στην οριζόντια και κατακόρυφη, αντίστοιχα, συντεταγμένη της πάνω αριστερής γωνίας της περιοχής σχεδίασης. Και οι δυο έχουν εξορισμού την τιμή 0, δηλαδή ορίζουν το σημείο αρχής (της επάνω αριστερής γωνίας) ως (0,0).
Ακόμη, μπορούμε να ορίσουμε το σύστημα συντεταγμένων κατά το χρόνο εκτέλεσης του προγράμματος χρησιμοποιώντας τη μέθοδο **Scale**, η οποία έχει γενική σύνταξη:

*αντικείμενο*.**Scale** *(x1, y1) - (x2, y2)*

με (x1, y1) την πάνω αριστερή γωνία του αντικειμένου και (x2, y2) την κάτω δεξιά γωνία. Αν χρησιμοποιήσουμε τη μέθοδο `Scale` χωρίς ορίσματα, επαναφέρουμε το προκαθορισμένο σύστημα αξόνων.

**Άσκηση 15-5.**

Θα δημιουργήσουμε ένα πρόγραμμα που θα σχεδιάζει σε ένα πλαίσιο εικόνας μια ημιτονική καμπύλη στο διάστημα [0, 6π].
Όπως γνωρίζουμε, η ημιτονική συνάρτηση παίρνει τιμές από -1 έως 1. Η πάνω αριστερή γωνία πρέπει λοιπόν να έχει συντεταγμένες (0,1) και η κάτω δεξιά γωνία πρέπει να έχει συντεταγμένες (6π,-1).
Αν `SinusArea` είναι το όνομα του πλαισίου εικόνας μέσα στο οποίο γίνεται η σχεδίαση, το πιο κάτω τμήμα προγράμματος σχεδιάζει την καμπύλη.




***Εικόνα 15-13.*** *Σημείο (0,1), σημείο (6π,-1)*

```
Dim ð            As Double
Dim Angle        As Double
ð = 4 * Atn(1)
SinusArea.Scale (0, 1)-(6 * ð, -1)
SinusArea.DrawWidth = 3
For Angle = 0 To 6 * ð Step ð / 30
     SinusArea.PSet (Angle, Sin(Angle))
Next A
```



Ας σημειωθεί ότι ο κώδικας δε χρειάζεται καμία αλλαγή αν αλλάξουμε τις διαστάσεις του αντικειμένου `SinusArea`. Η VB θα πραγματοποιήσει αυτόματα την απαραίτητη αλλαγή κλίμακας. Αυτό επιτρέπει την ομαλή εκτέλεση των εφαρμογών VB σε περιβάλλοντα με διαφορετικές αναλύσεις οθόνης χωρίς σχεδόν καμία αλλαγή στον κώδικα.

## Ανακεφαλαίωση

Η VB διαθέτει δύο διαφορετικούς τρόπους για τη δημιουργία γραφικών. Στον έναν τρόπο χρησιμοποιούνται τα εργαλεία γραφικών της παλέτας εργαλείων και στον άλλο τρόπο χρησιμοποιούνται οι μέθοδοι γραφικών της γλώσσας προγραμματισμού. Τα εργαλεία γραφικών του περιβάλλοντος εργασίας της VB είναι το εργαλείο "γραμμή", το εργαλείο "γεωμετρικό σχήμα", το εργαλείο "εικόνα" και το εργαλείο "πλαίσιο εικόνας". Η σχεδίαση με μεθόδους γραφικών μπορεί να γίνει στο εσωτερικό μιας φόρμας, στο εσωτερικό ενός πλαισίου εικόνας, στο χαρτί του εκτυπωτή. Οι πιο απλές μέθοδοι γραφικών είναι οι `Pset`, η οποία αποδίζει το χρώμα που ορίζουμε σε συγκεκριμένο σημείο, η `Line`, η οποία σχεδιάζει γραμμές και ορθογώνια σχήματα, η `Circle`, η οποία σχεδιάζει καμπυλόγραμμα σχήματα (κύκλους, ελλείψεις, τόξα, τομείς) και η `Cls`, η οποία καθαρίζει όλα τα γραφικά και το κείμενο, τα οποία έχουν δημιουργηθεί με μεθόδους κατά την εκτέλεση της εφαρμογής, σε μια φόρμα ή ένα πλαίσιο εικόνας.

## Εργαστηριακές Ασκήσεις

1. Εκτελέστε την άσκηση 15-1 με αντικείμενα εικόνας και πλαισίου εικόνας. Τι παρατηρείτε σε σχέση με τα σχήματα και το αντικείμενο εικόνας;
2. Σε μια φόρμα να δημιουργήσετε ένα αντικείμενο πλαισίου εικόνας. Από την ιδιότητα **Picture** καθορίστε να εμφανίζεται μια εικόνα μικρότερων διαστάσεων στο εσωτερικό του. Δώστε την τιμή `True` στην ιδιότητα **AutoSize**. Τι παρατηρείτε; Αλλάξτε πάλι τις διαστάσεις του αντικειμένου πλαισίου εικόνας και ζητήστε την εκτέλεση του προγράμματος. Τι παρατηρείτε;
3. Εκτελέστε την άσκηση 15-2. Σε τι οφείλονται οι ασυνέχειες κατά τη σχεδίαση; Πώς μπορείτε να βελτιώσετε το οπτικό αποτέλεσμα;
4. Εκτελέστε την άσκηση 15-3. Επαναλάβετε το πρόγραμμα δίνοντας διαφορετικές τιμές στην ιδιότητα **DrawMode**. Τι παρατηρείτε; Σχολιάστε ειδικά την περίπτωση για την τιμή `vbCopyPen`.
5. Δημιουργήστε το τμήμα προγράμματος που σχεδιάζει μια πύλη OR.
6. Δημιουργήστε μια φόρμα στα αριστερά της οποίας να υπάρχει μια εργαλειοθήκη με εργαλεία σχεδίασης: μολύβι για ελεύθερο σχέδιο, ευθύγραμμο τμήμα, ορθογώνιο, κύκλος. Δώστε τη δυνατότητα στον χρήστη να μπορεί να επιλέξει εργαλείο, να μπορεί να καθορίσει χρώμα και να μπορεί να σχεδιάζει σε μια περιοχή σχεδίασης.
7. Εκτελέστε την άσκηση 15-5. Επαναλάβετε την άσκηση για τη συνάρτηση συνημιτόνου. Πώς πρέπει να διαμορφωθούν οι κλίμακες για ημίτονο πλάτους a;
8. Επαναλάβετε την άσκηση 15-5 για τη συνάρτηση του τετραγωνικού παλμού που περιγράφεται στο παράδειγμα 7-5. Πώς μπορείτε να δημιουργήσετε και να παρουσιάσετε τριγωνικό παλμό;

## Σημειώσεις:

## Μάθημα 16
# Κινούμενο σχέδιο
# Τεχνική σύρε κι άσε

Τελειώνοντας αυτό το μάθημα οι μαθητές θα μπορούν:

- Να δημιουργούν απλό κινούμενο σχέδιο με εναλλαγή ή και με μετάθεση εικόνων.
- Να χρησιμοποιούν την τεχνική "σύρε κι άσε" στις εφαρμογές τους.

Το σύστημα της όρασης παρουσιάζει αδράνεια στην επεξεργασία μιας εικόνας με αποτέλεσμα, αν μας προβληθούν γρήγορα πολλές διαδοχικές εικόνες, οι οποίες διαφέρουν λίγο μεταξύ τους, να μας δημιουργείται η εντύπωση της συνεχούς κίνησης των αντικειμένων που έχουν αλλάξει θέση. Αυτό το φαινόμενο είναι γνωστό ως μετείκασμα και το εκμεταλλευόμαστε στη δημιουργία του κινούμενου σχεδίου. Το κινούμενο σχέδιο δημιουργείται από μια σειρά διαδοχικών σχεδίων. Η γρήγορη παρουσίαση των σχεδίων με ρυθμό 10 -15 σχέδια το δευτερόλεπτο προκαλεί την ψευδαίσθηση της ομαλής μετακίνησης των αντικειμένων ή της αλλαγής του σχήματός τους. Βασικά στη δημιουργία του κινούμενου σχεδίου χρησιμοποιούνται δύο τεχνικές: ***η κίνηση πάνω σε τροχιά (path animation)*** και η ***διαδοχή διαφορετικών στάσεων (cel animation).***

Το δεύτερο θέμα με το οποίο θα ασχοληθούμε σε αυτό το μάθημα αφορά τη λειτουργία ***σύρε κι άσε (drag & drop)***. Στη λειτουργία αυτή πατάμε το πλήκτρο του ποντικιού και επιλέγουμε ένα αντικείμενο στην οθόνη, σύρουμε (drag) το ποντίκι ενώ εξακολουθούμε να κρατάμε πατημένο το πλήκτρο, παρασύροντας το αντικείμενο σε μια άλλη θέση, και αφήνουμε το πλήκτρο του ποντικιού, οπότε ταυτόχρονα αφήνεται (drop) και το αντικείμενο. Λίγο πολύ όλοι μας έχουμε χρησιμοποιήσει αυτόν τον χειρισμό για να μεταφέρουμε ή να αντιγράψουμε ένα αρχείο στον Explorer των Windows ή για να μεταφέρουμε το περιεχόμενο μιας σειράς κελιών ενός υπολογιστικού φύλλου του Excel σε μιαν άλλη θέση.

## Κινούμενο σχέδιο

Κίνηση πάνω σε τροχιά

Για να δημιουργήσουμε κίνηση μιας εικόνας πάνω σε τροχιά, τοποθετούμε την εικόνα πάνω στο φόντο μιας φόρμας. Επίσης, στη φόρμα τοποθετούμε και ένα χρονόμετρο ως βηματοδότη της κίνησης. Σε διαδοχικές χρονικές στιγμές, που καθορίζονται από την ιδιότητα **`Interval`** του χρονομέτρου, μετατοπίζουμε την εικόνα σε μια νέα θέση πάνω στην ευθεία, την τεθλασμένη ή την καμπύλη που μας έχει υποδειχθεί ως τροχιά της κίνησης. Οι αποστάσεις των διαδοχικών θέσεων πάνω στη γραμμή, σε συνάρτηση με το χρονικό διάστημα που μεσολαβεί, καθορίζουν και την ταχύτητα της κίνησης. Για την κίνηση ενός αντικειμένου ελέγχου σε μια νέα θέση χρησιμοποιούμε τη μέθοδο `Move`. Η γενική μορφή της μεθόδου `Move` είναι:

*αντικείμενο*.`Move` *x, y*

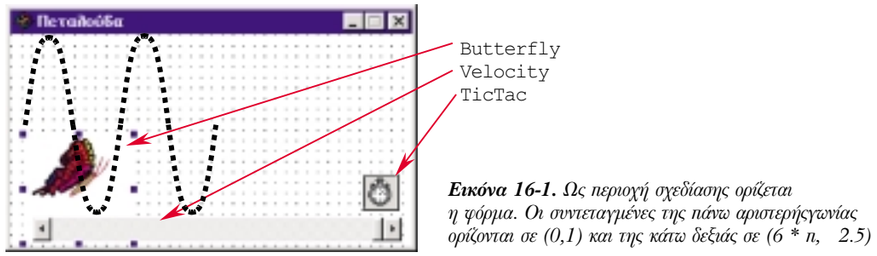
με x, y τις συντεταγμένες της νέας θέσης της πάνω αριστερής κορυφής του αντικειμένου.

**Άσκηση 16-1.**

Στον φάκελο `"C:\Program Files\DevStudio\VB\Samples\Pguide\VCR"` υπάρχει αρχείο με όνομα `Bfly2.bmp`, το οποίο περιέχει την εικόνα μιας πεταλούδας. Να δημιουργηθεί πρόγραμμα που να παρουσιάζει την πεταλούδα να πετά ακολουθώντας μια ημιτονική τροχιά. Η ταχύτητα πτήσης της πεταλούδας να καθορίζεται από οριζόντια ράβδο

1. Πάνω στη φόρμα τοποθετούμε αντικείμενο εικόνας με όνομα `Butterfly`. Προτιμάμε τα αντικείμενα εικόνας, διότι, όπως έχουμε αναφέρει, σχεδιάζονται γρηγορότερα και είναι δυνατή η κάλυψη όλου του εσωτερικού τους με το περιεχόμενο του αρχείου (ιδιότητα `Stretch = True`). Φορτώνουμε την εικόνα της πεταλούδας στο αντικείμενο, μεταβάλλοντας την ιδιότητα **`Picture`**.

2. Τοποθετούμε αντικείμενο χρονομέτρου, το οποίο ονομάζουμε TicTac.
3. Τέλος, τοποθετούμε ράβδο κύλισης με όνομα Velocity. Στις ιδιότητες **Min** και **Max** δίνουμε τις τιμές 30 και 200, που αντιστοιχούν σε 33 και 5 αλλαγές το δευτερόλεπτο.




***Εικόνα 16-1.*** *Ως περιοχή σχεδίασης ορίζεται η φόρμα. Οι συντεταγμένες της πάνω αριστερής γωνίας ορίζονται σε (0,1) και της κάτω δεξιάς σε (6 * π, 2.5)*

4. Γράφουμε τον κώδικα:

```
Dim ð    As Double              'Ï áñéèìüò ð
Dim x    As Double              'ÓõíôåôáãìÝíç x
Dim y    As Double              'ÓõíôåôáãìÝíç y

Private Sub Form_Load()
   'Áñ÷éêïðïßçóç
    ð = 4 * Atn(1)              'Ïñéóå ôïí áñéèìü ð
    Scale (0, 1)-(6 * ð, -2.5) 'Êáèüñéóå ôçí êëßìáêá
    x = 0:   y = 0              'Áñ÷éêÞ èÝóç áíôéêåéìÝíïõ
    TicTac.Interval = 50        '20 áëëáãÝò ôï äåõôåñüëåðôï
End Sub

Private Sub Velocity_Change()
TicTac.Interval = Velocity.Value
End Sub

Private Sub TicTac_Timer()
   'ÍÝá èÝóç ôïõ áíôéêåéìÝíïõ
    Butterfly.Move x, y
   'Õðïëüãéóå ôçí åðüìåíç èÝóç
     x = x + ð / 20             'Ìåôáôüðéóç óôïí x
    If x > 6 * ð Then x = 0     'Áí âãåéò åêôüò öüñìáò åðáíÜöåñå
    y = Sin(x)                  'Ìåôáôüðéóç óôïí y
End Sub
```

Με τη φόρτωση της φόρμας γίνονται οι αρχικοποιήσεις. Η κλίμακα της φόρμας ορίζεται ώστε να μην κρύβεται κανένα μέρος της εικόνας κατά την κίνηση της πεταλούδας. Σε τακτά χρονικά διαστήματα, που καθορίζονται από την τιμή της ιδιότητας **Interval** του αντικειμένου TicTac, υπολογίζεται η μετατόπιση της εικόνας ως προς τον άξονα των x και η μετατόπιση της εικόνας ως προς τον άξονα των y.

Διαδοχή διαφορετικών στάσεων

Στην τεχνική της εναλλαγής διαφορετικών στάσεων τοποθετούμε πάνω στη φόρμα ένα αντικείμενο ελέγχου που μπορεί να δεχτεί μια εικόνα (εικόνα, πλαίσιο εικόνας, πλήκτρο διαταγής). Επίσης, στη φόρμα τοποθετούμε και ένα χρονόμετρο. Σε διαδοχικές χρονικές στιγμές, που καθορίζονται από την ιδιότητα **Interval** του χρονομέτρου, προβάλλουμε στο εσωτερικό του αντικειμένου ελέγχου, διαφορετικές στάσεις ή καταστάσεις των αντικειμένων του σχεδίου. Η γρήγορη εναλλαγή των στιγμιοτύπων δημιουργεί την εντύπωση της μεταβολής. Τα στιγμιότυπα είτε φορτώνονται από αρχείο είτε είναι φορτωμένα πάνω στη φόρμα σε άλλα αντικείμενα και ζητείται η μεταφόρτωσή τους.

Η φόρτωση μιας εικόνας στην ιδιότητα **Picture** κατά τη στιγμή της εκτέλεσης του προγράμματος γίνεται με τη συνάρτηση LoadPicture. Ο τρόπος εκχώρησης και η γενική σύνταξη της συνάρτησης είναι:

*αντικείμενο*.Picture = LoadPicture(*όνομα_αρχείου*)



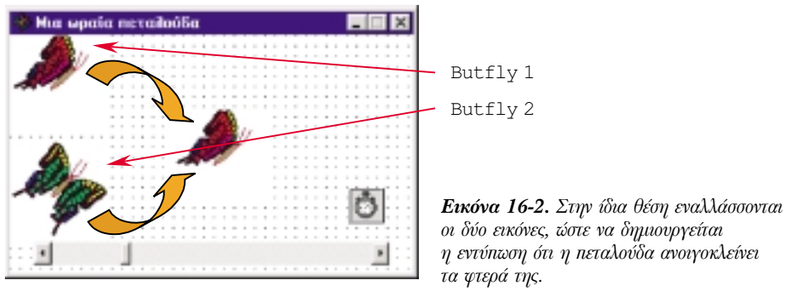
Η φόρτωση απευθείας από αρχείο θέτει περιορισμούς ταχύτητας. Στην περίπτωση λοιπόν που τα στιγμιότυπα είναι λίγα στον αριθμό, συμφέρει να τα έχουμε προφορτώσει σε άλλα μη ορατά αντικείμενα της φόρμας και να τα μεταφορτώνουμε τη στιγμή της εκτέλεσης του προγράμματος στο αντικείμενο εικόνα. Η μεταφόρτωση γίνεται με απλή εκχώρηση της τιμής της ιδιότητας **Picture** του ενός αντικειμένου, στην ιδιότητα **Picture** του άλλου, π.χ.

*αντικείμενο1*.Picture = *αντικείμενο2*.Picture

**Άσκηση 16-2.**

Στον φάκελο "C:\Program Files\DevStudio\VB\Samples\Pguide\VCR" εκτός από το αρχείο Bfly2.bmp υπάρχει και το αρχείο Bfly1.bmp με μια διαφορετική στάση πτήσης της πεταλούδας. Να δημιουργηθεί πρόγραμμα που να παρουσιάζει την πεταλούδα να πετά ανοιγοκλείνοντας τα φτερά της. Η ταχύτητα πτήσης της πεταλούδας να καθορίζεται από οριζόντια ράβδο κύλισης.

1. Πάνω στη φόρμα τοποθετούμε αντικείμενο εικόνας με όνομα Butterfly. Δε χρειάζεται να του εκχωρήσουμε κάποια εικόνα, αυτό θα γίνει κατά τη διάρκεια εκτέλεσης του προγράμματος.
2. Επίσης, τοποθετούμε δύο αντικείμενα εικόνας με ονόματα Butfly1 και Butfly2 αντίστοιχα, τα οποία θα διατηρήσουμε αόρατα κατά τη διάρκεια εκτέλεσης του προγράμματος, γι' αυτό και δίνουμε την τιμή False στην ιδιότητα τους **Visible**.
3. Τοποθετούμε αντικείμενο χρονομέτρου, το οποίο ονομάζουμε TicTac.
4. Τέλος, τοποθετούμε ράβδο κύλισης με όνομα Velocity. Στις ιδιότητες **Min** και **Max** δίνουμε τις τιμές 10 και 1000, που αντιστοιχούν σε 10 και 1 αλλαγές το δευτερόλεπτο.




***Εικόνα 16-2.*** *Στην ίδια θέση εναλλάσσονται οι δύο εικόνες, ώστε να δημιουργείται η εντύπωση ότι η πεταλούδα ανοιγοκλείνει τα φτερά της.*

5. Γράφουμε τον κώδικα:

```
Dim FlipFlop      As Integer      'Óçìáöüñïò åíáëëáãÞò
Private Sub Form_Load()
   Const Folder = "C:\Program Files\DevStudio\VB\Samples\Pguide\VCR\"
     'Áñ÷éêïðïßçóç
   Butfly1.Picture = LoadPicture(Folder & "Bfly1.bmp")
   Butfly2.Picture = LoadPicture(Folder & "Bfly2.bmp")
   FlipFlop = 0
End Sub

Private Sub TicTac_Timer()
   '¢ëëáîå ôç óôÜóç áíÜëïãá ìå ôçí ôéìÞ ôïõ óçìáöüñïõ
    If FlipFlop Then
         Butterfly.Picture = Butfly1.Picture
    Else
         Butterfly.Picture = Butfly2.Picture
    End If
    FlipFlop = Not FlipFlop
End Sub

Private Sub HScroll1_Change()
    TicTac.Interval = Velocity.Value
End Sub
```

Με τη φόρτωση της φόρμας γίνονται οι αρχικοποιήσεις. Στα αντικείμενα `Butfly1` και `Butfly2` φορτώνονται τα στιγμιότυπα της πεταλούδας και δίνεται αρχική τιμή στη μεταβλητή `FlipFlop`, που θα λειτουργήσει ως σημαφόρος εναλλαγής. Σε τακτά χρονικά διαστήματα, που καθορίζονται από την τιμή της ιδιότητας **`Interval`** του αντικειμένου `TicTac` και ανάλογα με την τιμή του σημαφόρου `FlipFlop`, φορτώνεται στο ορατό αντικείμενο `Butterfly`, η μια ή η άλλη εικόνα και αλλάζει η τιμή του σημαφόρου.

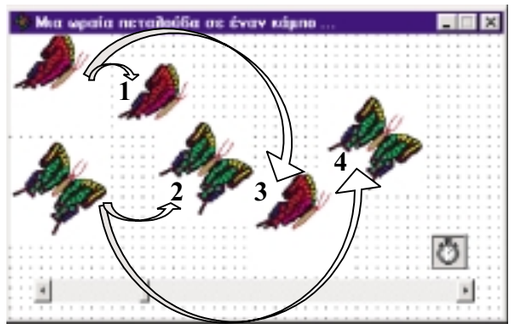
Συνδυασμός κίνησης πάνω σε τροχιά και διαδοχής διαφορετικών στάσεων

Συνήθως, χρησιμοποιείται ο συνδυασμός των δύο τεχνικών. Δημιουργείται εναλλαγή εικόνων μαζί με παράλληλη μετατόπιση.

**Άσκηση 16-3.**

Να δημιουργηθεί πρόγραμμα που να παρουσιάζει την πεταλούδα να πετά ανοιγοκλείνοντας τα φτερά της, ενώ μετακινείται πάνω σε ημιτονική τροχιά.

**1.** Χρησιμοποιούμε τη φόρμα με τα αντικείμενα της άσκησης 16-2.




***Εικόνα 16-3.*** *Διαδοχικά στιγμιότυπα εναλλαγής της εικόνας και μετατόπισης*

**2.** Συνδυάζουμε τους κώδικες των ασκήσεων 16-1 και 16-2 σε:

```
Dim FlipFlop As Integer   'Óçìáöüñïò åíáëëáãÞò
Dim ð    As Double       'Ï áñéèìüò ð
Dim x    As Double       'Óõíôåôáãìýíç x
Dim y    As Double       'Óõíôåôáãìýíç y

Private Sub Form_Load()
    Const Folder = "C:\Program Files\DevStudio\VB\Samples\Pguide\VCR\"
   'Áñ÷éêïðïßçóç
    ð = 4 * Atn(1)             'Ïñéóå ôïí áñéèìü ð
    Scale (0, 1)-(6 * ð, -2.5) 'Êáèüñéóå ôçí êëßìáêá
    x = 0:   y = 0             'Áñ÷éêÞ èÝóç áíôéêåéìýíïõ
    btrfly1.Picture = LoadPicture(Folder & "Bfly1.bmp")
    btrfly2.Picture = LoadPicture(Folder & "Bfly2.bmp")
    FlipFlop = 0
End Sub

Private Sub TicTac_Timer()
   'Ćëëáîå ôç óôÜóç
    If FlipFlop = 0 Then
       btrfly.Picture = btrfly1.Picture
       FlipFlop = 1
    Else
       btrfly.Picture = btrfly2.Picture
       FlipFlop = 0
    End If
   'Ćëëáîå ôç èÝóç
    btrfly.Move x, y
    x = x + ð / 20
    If x > 6 * ð Then x = 0
    y = Sin(x)
End Sub
```



```
Private Sub HScroll1_Change()
   Timer2.Interval = HScroll1.Value
End Sub
```

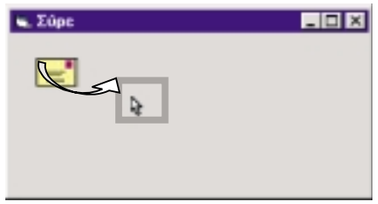
## Η τεχνική σύρε κι άσε

Για να "απελευθερώσει" ο προγραμματιστής ένα αντικείμενο, ώστε να έχει τη δυνατότητα ο χρήστης να το σύρει πάνω σε μια φόρμα μπορεί:

- είτε να δώσει στην ιδιότητα **`DragMode`** του αντικειμένου την τιμή **1 - Automatic** (εσωτερική σταθερά `vbAutomatic` στην περίπτωση, που κάνει την εκχώρηση της τιμής μέσα από κώδικα),
- είτε να αφήσει στην ιδιότητα **`DragMode`** την τιμή **0 - Manual** (εσωτερική σταθερά `vbManual` στην περίπτωση που κάνει την εκχώρηση της τιμής μέσα από κώδικα) και να χρησιμοποιήσει τη μέθοδο **`Drag`**.

Ο πρώτος τρόπος, το ***αυτόματο σύρε κι άσε***, είναι και ο πιο απλός, αλλά έχει το μειονέκτημα ότι απομονώνει το αντικείμενο από τα συμβάντα του ποντικιού. Για παράδειγμα, το αντικείμενο παύει να αντιδρά στο συμβάν **`Click`**.

**Άσκηση 16-4.**

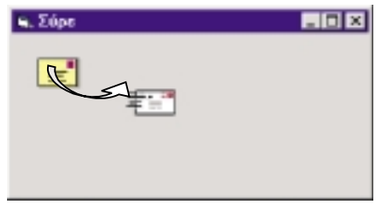



***Εικόνα 16-4.*** *Είδωλο χωρίς εικόνα*

1. Σε μια φόρμα τοποθετούμε ένα αντικείμενο εικόνας και του επισυνάπτουμε την εικόνα του αρχείου `C:\...\Graphics\Icons\Mail01a.ico`.
2. Στην ιδιότητα **`DragMode`** δίνουμε την τιμή **1 - Automatic.**
3. Τρέχουμε το πρόγραμμα.
4. Σύρουμε το αντικείμενο. Παρατηρούμε ότι ένα γκρι πλαίσιο, που αναπαριστά το αντικείμενο, κινείται μαζί με το ποντίκι.

Για να έχει τη μορφή εικόνας το αντικείμενο τη στιγμή που σύρεται, πρέπει να δώσουμε τιμή στην ιδιότητα **`DragIcon`**.

**Άσκηση 16-5.**




***Εικόνα 16-5.*** *Είδωλο με εικόνα*

1. Στην ιδιότητα **`DragIcon`** του αντικειμένου δίνουμε την τιμή `C:\...\Graphics\Icons\Mail03.ico`.
2. Τρέχουμε το πρόγραμμα.
3. Σέρνουμε το αντικείμενο. Παρατηρούμε ότι ο δείκτης του ποντικιού έχει αντικατασταθεί από την εικόνα που έχουμε εκχωρήσει στην ιδιότητα **`DragIcon`**.

Μπορούμε να εκχωρήσουμε τιμή στην ιδιότητα **`DragIcon`** και κατά τη στιγμή της εκτέλεσης του προγράμματος. Τότε είτε χρησιμοποιούμε τη συνάρτηση `LoadPicture`, είτε την ιδιότητα **`Picture`** άλλου αντικειμένου (δείτε προηγούμενη παράγραφο).

Η μετάθεση του ειδώλου του αντικειμένου σε μιαν άλλη θέση, δε συνεπάγεται και τη μετάθεση του αντικειμένου σε αυτήν τη θέση, όταν τελικά αφήσουμε το πλήκτρο του ποντικιού. Για να γίνει αυτό πρέπει να έχουμε κάνει κατάλληλη κωδικοποίηση στο συμβάν **`DragDrop`**. Το συμβάν **`DragDrop`** προκύπτει σε όλα τα ***αντικείμενα στόχους*** (π.χ. φόρμες, αντικείμενα ελέγχου), πάνω στα οποία ο χρήστης αφήνει ένα αντικείμενο πηγή, δηλαδή αντικείμενο που έσερνε με το ποντίκι. Η υπορουτίνα διαχείρισης συμβάντος **`DragDrop`** για ένα αντικείμενο συντάσσεται ως:

```
Sub αντικείμενο_DragDrop(Source As Control, X As Single, Y As Single)
```

Η παράμετρος `Source` παριστάνει στο αντικείμενο πηγή. Επίσης, οι παράμετροι `X` και `Y` αντιστοιχούν στις συντεταγμένες του σημείου που αφέθηκε το πλήκτρο του ποντικιού.

**Άσκηση 16-6.**

Θέλουμε, τα αντικείμενα που σύρονται πάνω σε μια φόρμα, να μετακινούνται στο σημείο που αφήνουμε το είδωλό τους.

Γράφουμε τον κώδικα:

```
Sub Form_DragDrop(Source As Control, X As Single, Y As Single)
   Source.Move X, Y
End Sub
```

Ένα άλλο εξίσου σημαντικό συμβάν που έχει σχέση με την τεχνική σύρε κι άσε, είναι το συμβάν. Η σύνταξη της υπορουτίνας διαχείρισης συμβάντος **DragOver** είναι παρόμοια με τη σύνταξη της υπορουτίνας διαχείρισης συμβάντος **DragDrop**. Συγκεκριμένα, γράφουμε:

```
Sub αντικείμενο_DragOver(Source As Control, X As Single, Y As Single, _
                         State As Integer)
```

Το συμβάν αυτό προκύπτει όταν ένα αντικείμενο σέρνεται πάνω από ένα άλλο. Η παράμετρος `State` μπορεί να πάρει τις τιμές:

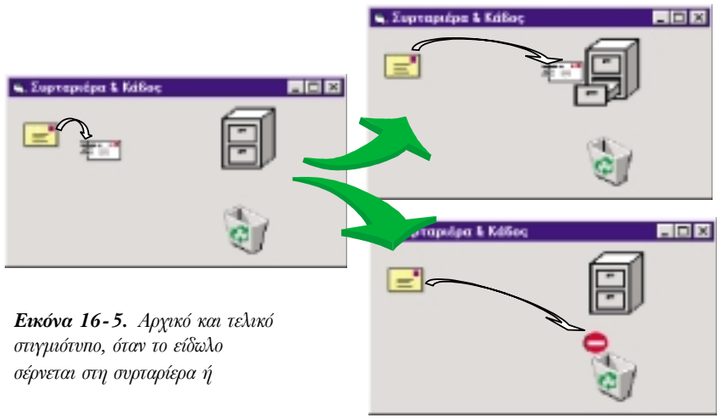
`VbEnter` το ποντίκι μόλις έχει μπει στα όρια του αντικειμένου στόχου,

`VbLeave` το ποντίκι μόλις έχει αφήσει τα όρια του αντικειμένου στόχου,

`vbOver` το ποντίκι βρίσκεται πάνω στο αντικείμενο στόχο.

**Άσκηση 16-7.**

Στη φόρμα που χρησιμοποιήσαμε στις πιο πάνω ασκήσεις, να τοποθετηθεί ένα αντικείμενο εικόνας με όνομα `Filer`, στο οποίο να δοθεί η μορφή συρταριέρας και ένα αντικείμενο εικόνας με όνομα `Bin`, στο οποίο να δοθεί η μορφή κάδου απορριμάτων. Όταν σέρνουμε τον φάκελο πάνω από τη συρταριέρα, να ανοίγει το κάτω συρτάρι της και αν αφήσουμε το πλήκτρο του ποντικιού, το συρτάρι να κλείνει απορροφώντας το γράμμα. Επίσης, όταν σέρνουμε τον φάκελο πάνω από τον κάδο απορριμάτων να εμφανίζεται ένα απαγορευτικό σήμα που να μας ειδοποιεί ότι δεν μπορούμε να καταστρέψουμε αυτόν τον φάκελο.




***Εικόνα 16-5.*** *Αρχικό και τελικό στιγμιότυπο, όταν το είδωλο σέρνεται στη συρταριέρα ή*

Γράφουμε τον κώδικα:

```
'ÖÜêåëïò ìå åéêïíßäéá
Const Folder = "C:\Program Files\DevStudio\Vb\Graphics\Icons"

'Áí ôï åßäùëï âñßóêåôáé ðÜíù áðù ôç óõñôáñéÝñá
Private Sub Filer_DragOver(Source As Control, X As Single, Y As Single, _
                           State As Integer)
    Select Case State
        Case vbEnter    'Áíôüù ôçò óõñôáñéÝñáò. Åéêïíßäéï áíïé÷ôü óõñôÜñé
            Filer.Picture = LoadPicture(Folder & "\Office\Files03b.ico")
        Case vbLeave    'Âêôüù ôçò óõñôáñéÝñáò. Åéêïíßäéï êëåéóôü óõñôÜñé
            Filer.Picture = LoadPicture(Folder & "\Office\Files03a.ico")
    End Select
End Sub
```



```
'Ï ÷ñÞóôçò Üöçóå ôï ðïíôßêé
Private Sub Filer_DragDrop(Source As Control, X As Single, Y As Single)
   'Ìåôáêßíçóå ôïí öÜêåëï êáé êñýøå ôïí
    Source.Move X, Y
    Source.Visible = False
   'Áéêïíßäéï êëåéóôü óõñôÜñé
    Filer.Picture = LoadPicture(Folder & "\Office\Files03a.ico")
End Sub

'Áí ôï åßäùëï âñßóêåôáé ðÜíù áðù ôïí êÜäï
Private Sub Bin_DragOver(Source As Control, X As Single, Y As Single, _
                         State As Integer)
    Select Case State
        Case vbEnter    'Áíôüù ôïõ êÜäïõ. Åéêïíßäéï Stop
            Source.DragIcon = LoadPicture(Folder & "\Misc\Misc06.ico")
        Case vbLeave    'Âêôüù ôïõ êÜäïõ. Åéêïíßäéï öÜêåëïò
            Source.DragIcon = LoadPicture(Folder & "\Mail\Mail03.ico")
    End Select
End Sub
```

Όλα όσα αναφέραμε μέχρι αυτό το σημείο μπορούμε να τα εκμεταλλευτούμε τόσο στην αυτόματη, όσο και στην υπό έλεγχο τεχνική σύρε κι άσε. Η δεύτερη τεχνική, η ***σύρε κι άσε υπό έλεγχο*** (ιδιότητα **DragMode** με τιμή **0 - Manual**), είναι πιο ευέλικτη, αφού προσφέρει στον προγραμματιστή τη δυνατότητα ελέγχου των ενεργειών του χρήστη, σε ότι αφορά τον χειρισμό του ποντικιού. Ο προγραμματιστής, ανάλογα με την περίπτωση, μπορεί να επιτρέπει ή να μην επιτρέπει στο χρήστη να σύρει ένα αντικείμενο. Η ενεργοποίηση της λειτουργίας σύρε κι άσε γίνεται προγραμματιστικά, κατά το συμβάν **MouseDown**, ζητώντας την εκτέλεση της μεθόδου **Drag** με τιμή παραμέτρου `vbBeginDrag`. Επίσης, η απενεργοποίηση γίνεται κατά το συμβάν **MouseUp**, ζητώντας την εκτέλεση της μεθόδου **Drag** με τιμή παραμέτρου `vbEndDrag`.

**Άσκηση 16-8.**

Αν εκτελέσουμε την άσκηση 16-6, παρατηρούμε ότι ναι μεν τα αντικείμενα μετακινούνται σε μια νέα θάση, αλλά αυτή δε συμπίπτει με το σημείο που δείχνει το ποντίκι. Αυτό οφείλεται στο ότι μπορούμε να πιάσουμε και να σύρουμε το αντικείμενο από οποιοδήποτε σημείο του. Όμως, κατά την απόθεση του αντικειμένου δε λαμβάνουμε υπ' όψιν μας τη σχετική θέση του δείκτη του ποντικιού ως προς την πάνω αριστερή γωνία του αντικειμένου κι γι' αυτό και παρατηρείται και η μετάπτωση. Μας ζητείται να βελτιώσουμε το αισθητικό αποτέλεσμα.
Αν το όνομα του αντικειμένου που πρόκειται να σύρουμε είναι Envelope:

1. Δίνουμε στην ιδιότητα **DragMode** την τιμή **0 - Manual.**
2. Γράφουμε τον κώδικα:

```
Dim StartX      As Single
Dim StartY      As Single
Sub Envelope_MouseDown (Button As Integer, Shift As Integer, _
                        X As Single, Y As Single)
    Envelope.Drag vbBeginDrag
    StartX = X
    StartY = Y
End Sub

Sub Form_DragDrop(Source As Control, X As Single, Y As Single)
    Source.Move (X - StartX), (Y - StartY)
End Sub

Sub Envelope_MouseUp (Button As Integer, Shift As Integer, _
                      X As Single, Y As Single)
    Envelope.Drag vbEndDrag
End Sub
```

Για να γίνει σωστά η μετάθεση πρέπει και οι κλίμακες του αντικειμένου και της φόρμας να έχουν τις ίδιες μονάδες.

Όταν ο χρήστης πατήσει το πλήκτρο του ποντικιού πάνω στο αντικείμενο, εκτελείται η υπορουτίνα διαχείρισης συμβάντος **MouseDown**, η οποία ενεργοποιεί τη λειτουργία σύρε κι άσε και συγκρατεί τις σχετικές συντεταγμένες του ποντικιού ως προς την πάνω αριστερή γωνία του αντικειμένου (αυτές ακριβώς τις τιμές παίρνουν οι παράμετροι X, Y της υπορουτίνας).

όταν ο χρήστης αφήσει το ποντίκι εκτελείται η υπορουτίνα `Form_DragDrop`, που μετακινεί το αντικείμενο, και η υπορουτίνα `Envelope_MouseUp` που απενεργοποιεί τη λειτουργία σύρε κι άσε.

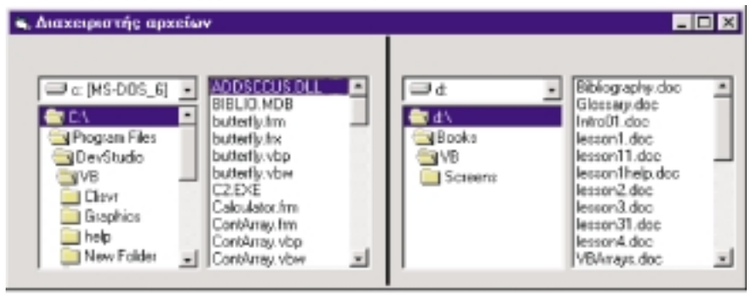
## Ανακεφαλαίωση

Η γρήγορη παρουσίαση των σχεδίων με ρυθμό 10 -15 σχέδια το δευτερόλεπτο προκαλεί την ψευδαίσθηση της ομαλής μετακίνησης των αντικειμένων ή της αλλαγής του σχήματός τους. Η δημιουργία του κινούμενου σχεδίου βασίζεται σε δύο τεχνικές: την κίνηση πάνω σε τροχιά και τη διαδοχή διαφορετικών στάσεων. Συνήθως στις εφαρμογές χρησιμοποιείται ο συνδυασμός τους. Σε κάθε περίπτωση πρέπει να υπάρχει ένας βηματοδότης που καθορίζει το ρυθμό μετατόπισης ή εναλλαγής των στάσεων.

Στη λειτουργία σύρε κι άσε πατάμε το πλήκτρο του ποντικιού και επιλέγουμε ένα αντικείμενο στην οθόνη. Σύρουμε (drag) το ποντίκι, ενώ εξακολουθούμε να κρατάμε πατημένο το πλήκτρο και παρασύρουμε το αντικείμενο σε μια άλλη θέση. Αφήνουμε το πλήκτρο του ποντικιού, οπότε ταυτόχρονα αφήνεται (drop) και το αντικείμενο. Υπάρχουν δύο τεχνικές σύρε κι άσε. Η αυτόματη και η υπό έλεγχο.

## Εργαστηριακές Ασκήσεις

1. Εκτελέστε την άσκηση 16-1. Δοκιμάστε διάφορες ταχύτητες κίνησης.
2. Αντί της ημιτονικής συνάρτησης χρησιμοποιήστε γεννήτρια τυχαίων αριθμών, ώστε η πεταλούδα στην άσκηση 16-1 να μετακινείται σε τυχαίες θέσεις στην κατακόρυφη διεύθυνση.
3. Εκτελέστε την άσκηση 16-2.
4. Εκτελέστε την άσκηση 16-3. Προσθέστε τυχαιότητα στην κίνηση με μια γεννήτρια τυχαίων αριθμών.
5. Εκτελέστε την άσκηση 16-4 και 16-5.
6. Εκτελέστε την άσκηση 16-4 πάνω στη φόρμα της άσκησης 16-5.
7. Εκτελέστε την άσκηση 16-7. Βελτιώστε τον κώδικα, ώστε να μπορείτε να διαχειριστείτε περισσότερα από ένα αντικείμενα.
8. Εκτελέστε την άσκηση 16-8.
9. Μέχρι τώρα χρησιμοποιήσαμε αντικείμενα εικόνας για να δείξουμε τον τρόπο με τον οποίο μπορούμε να εκμεταλλευτούμε την τεχνική σύρε κι άσε. Τα αντικείμενα αυτά δεν είναι δεσμευτικά, θα μπορούσαν να ήταν πλαίσια κειμένου, λίστες, συνδυασμένες λίστες ή άλλα αντικείμενα που δέχονται συμβάντα ποντικιού. Δημιουργήστε μια φόρμα χωρισμένη στη μέση. Σε κάθε περιοχή της φόρμας να έχουν τοποθετηθεί αντικείμενα πτυσσόμενης λίστας φακέλων, λίστας φακέλων, λίστας αρχείων.
   Να γραφτεί κώδικας, ώστε να μπορεί ο χρήστης να επιλέγει ένα αρχείο της μιας περιοχής της οθόνης, να το σύρει στη λίστα φακέλων της άλλης και αν το αφήνει σε αυτή να γίνεται και η αντιγραφή του αρχείου από τον ένα φάκελο στον άλλο.




***Εικόνα 16-7.***



## Μάθημα 17

# Πολυμέσα

Τελειώνοντας αυτό το μάθημα οι μαθητές θα μπορούν:

- Να περιγράφουν τι είναι η διεπαφή ελέγχου των μέσων.
- Να περιγράφουν τις ιδιότητες και τα συμβάντα του αντικειμένου πολυμέσων.
- Να χρησιμοποιούν το αντικείμενο πολυμέσων και το αντικείμενο ενσωμάτωσης και σύνδεσης για να δημιουργούν προγράμματα με δυνατότητες πολυμέσων.

Το λειτουργικό σύστημα των Windows είναι ένα λειτουργικό σύστημα ***ανεξάρτητο συσκευής (device-independent)***, με την έννοια ότι οι εφαρμογές μπορούν να χειριστούν τις συσκευές ενός συγκεκριμένου τύπου, ανεξάρτητα από τα ιδιαίτερα κατασκευαστικά χαρακτηριστικά που τους έχουν προσδόσει οι κατασκευαστές τους. Αυτή η ανεξαρτησία εξασφαλίζεται με την παραδοχή, ότι κάθε συσκευή συνοδεύεται από το κατάλληλο λογισμικό, το οποίο είναι γνωστό ως ***οδηγός συσκευής (device driver)*** και του οποίου η συμβατότητα με τα Windows είναι μέλημα του κατασκευαστή της.

## Η διεπαφή ελέγχου των μέσων

Τα Windows παρέχουν τη ***διεπαφή ελέγχου μέσων (Media Control Interface - MCI)***, μέσω της οποίας ο προγραμματιστής μπορεί να συμπεριλάβει στις εφαρμογές του ήχο, βίντεο και κινούμενο σχέδιο. Η MCI είναι ένα σύνολο από βιβλιοθήκες υπορουτινών και συναρτήσεων, οι οποίες επιτρέπουν στους προγραμματιστές να διαχειριστούν συσκευές, χωρίς να είναι υποχρεωμένοι να κατανοήσουν τις ειδικές και πολλές φορές σύνθετες υπορουτίνες, που περιέχονται στο λειτουργικό σύστημα ή στους οδηγούς συσκευών. Οι προγραμματιστές μπορούν να αναφέρονται στις συσκευές με συμβολικά ονόματα και να ζητούν την εκτέλεση συγκεκριμένων λειτουργιών, αποστέλοντας τυποποιημένες διαταγές σε μορφή συμβολοσειρών.

Οι συσκευές πολυμέσων, που μπορεί να υπάρχουν σε ένα υπολογιστικό σύστημα, είναι οι κάρτες ήχου, οι ακολουθητές MIDI (sequencers), οι συσκευές CD-ROM, οι συσκευές ακουστικών CD, οι συσκευές βιντεοταινιών και βιντεοδίσκων. Η διεπαφή MCI διακρίνει τις συσκευές πολυμέσων σε ***απλές (simple)*** και σε ***σύνθετες (compound)***. Απλές είναι οι συσκευές που δεν χρειάζονται κάποιο αρχείο δεδομένων και αναζητούν σε ίχνη (tracks) την πληροφορία πάνω στα φυσικά μέσα, π.χ. από το ακουστικό CD ή από τον βιντεοδίσκο. Οι συσκευές αυτές παίζουν κομμάτια και αντλούν την πληροφορία πάνω από το φυσικό μέσο (δίσκο) διαβάζοντας τα track. Οι σύνθετες συσκευές ωστόσο χρειάζονται ένα αρχείο δεδομένων. Για παράδειγμα, η κάρτα ήχου αντλεί τον ήχο από αρχείο ήχου.

Αν και οι κατασκευαστές υλικού συνεχώς βελτιώνουν τις συσκευές τους και εισάγουν καινούργιες στην αγορά, η ονοματολογία δεν είναι τυποποιημένη. Στον πίνακα 17-1 δίνονται τα τυποποιημένα ονόματα των συσκευών που μπορούν να ελεγχθούν από το MCI.

| Συμβολισμός | Συσκευή |
|---|---|
| **`CDAudio`** | Ακουστικό CD (audio CD player) |
| **`WaveAudio`** | Κάρτα ήχου (sound card) |
| **`Sequencer`** | MIDI sequencer (αρχεία `.mid` ή `.rmi`) |
| **`Dat`** | Συσκευή ψηφιακού ήχου σε ταινία (digital audio tape player) |
| **`Animation`** | "Συσκευή" για προβολή κινούμενου σχεδίου (animation device) |
| **`VCR`** | Συσκεύη βιντεοταινίας (videotape recorder or player) |
| **`VideoDisk`** | Βιντεοδίσκος |
| **`AVIVideo`** | Αρχείο που περιέχει βίντεο |

***Πίνακας 17-1.*** *Οι συσκευές MCI που μπορούν να ελεγχθούν από τα Windows.*

Οι περισσότερες συσκευές που ελέγχονται μέσω του MCI έχουν ένα σύνολο από κοινά χαρακτηριστικά που αφορούν τον τρόπο λειτουργίας τους, όπως ξεκίνημα, σταμάτημα, παύση, επιστροφή κ.ά. Για την πραγματοποίηση αυτών των λειτουργιών αποστέλλονται

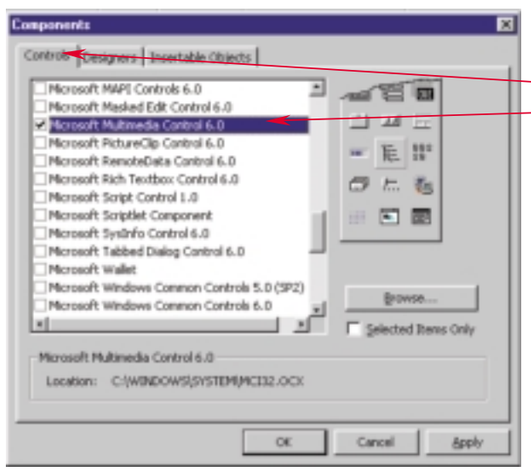
στη συσκευή διαταγές. Οι διαταγές αυτές έχουν τυποποιηθεί. Στον πίνακα 17-2 δίνονται οι τυποποιημένες κατά MCI διαταγές.

| Διαταγή | Λειτουργία |
|---|---|
| `Open` | Άνοιξε τη συσκευή |
| `Close` | Κλείσε τη συσκευή |
| `Play` | Παίξε το περιεχόμενο του μέσου (ήχο, βίντεο, ...) |
| `Pause` | Διέκοψε προσωρινά το παίξιμο |
| `Stop` | Σταμάτα |
| `Back` | Επέστρεψε σε προηγούμενα ίχνη (tracks) |
| `Step` | Προχώρησε σε επόμενα ίχνη |
| `Prev` | Επέστρεψε στην αρχή του προηγούμενου ίχνους |
| `Next` | Προχώρησε στην αρχή του επόμενου ίχνους |
| `Seek` | Κινήσου σε συγκεκριμένη θέση |
| `Record` | Κάνε εγγραφή |
| `Eject` | Απόβαλλε το φυσικό μέσο (π.χ δίσκο) από τη συσκευή |
| `Save` | Αποθήκευσε σε αρχείο |

***Πίνακας 17-2.*** *Οι διαταγές προς τις συσκευές MCI.*

## Το αντικείμενο πολυμέσων

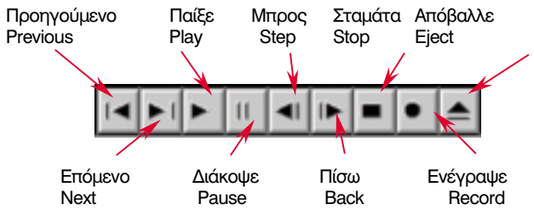
Η VB διαθέτει ένα ειδικό ***αντικείμενο ελέγχου, το αντικείμενο πολυμέσων (Microsoft Multimedia Control)***, με το οποίο οι προγραμματιστές μπορούν να χειρίζονται με απλό τρόπο τις συσκευές MCI. Αν στην εργαλειοθήκη δεν υπάρχει το αντίστοιχο εργαλείο, για την προσθήκη του κάνουμε τα εξής:




1. Επιλέγουμε **Project ¦ Components** από τη γραμμή μενού.
2. Στο παράθυρο Components που θα εμφανιστεί επιλέγουμε τον καρτελοδείκτη **Controls.**
3. Κάνουμε κλικ στη γραμμή **Microsoft Multimedia Control.**
4. Το εικονίδιο του αντικειμένου εμφανίζεται στην εργαλειοθήκη.

***Εικόνα 17-1.*** *Προσθήκη του Microsoft Multimedia Control στην εργαλειοθήκη*

Το αντικείμενο πολυμέσων περιλαμβάνει ένα σύνολο από πλήκτρα, τα οποία αντιστοιχούν στις λειτουργίες που διακρίνουμε σε μια τυπική συσκευή κασετοφώνου ή σε μια τυπική συσκευή βίντεο της τηλεόρασης.




***Εικόνα 17-2.*** *Τα πλήκτρα του αντικειμένου πολυμέσων*



Σε μια φόρμα μπορούμε να έχουμε πολλά αντικείμενα πολυμέσων προκειμένου να ελέγχουμε ταυτόχρονα πολλές συσκευές (χρησιμοποιούμε ένα αντικείμενο ανά συσκευή).

Ο τύπος της συσκευής που ελέγχει το αντικείμενο πολυμέσων ορίζεται στην ιδιότητα **`DeviceType`**. Η στήλη "Συμβολισμός" του πίνακα 17-1 περιέχει ενδεικτικές τιμές που μπορεί να πάρει αυτή η ιδιότητα. Έτσι, αν πρόκειται το αντικείμενο πολυμέσων να ελέγχει την κάρτα ήχου δίνουμε στην ιδιότητα αυτή την τιμή `waveaudio`, αν το αντικείμενο πολυμέσων ελέγχει τη συσκευή CD-ROM, στην οποία έχει τοποθετηθεί ένα ακουστικό CD, δίνουμε την τιμή `CDAudio` κλπ. Οι διαταγές για τον έλεγχο μιας συσκευής μπορούν να δοθούν μέσω της ιδιότητας **`Command`** του αντικειμένου πολυμέσων.

**Παραδείγματα 17-1.**

α) Οι εντολές:

```
MMControl1.DeviceType = "CDAudio"   'Ôýðïò óõóêåõÞò
MMControl1.Command = "Open"         'Aíïéîå ôç óõóêåõÞ
MMControl1.Command = "Play"         'Ðáßîå ôïí Þ÷ï ôïõ ðñþôïõ track
```

καθορίζουν ότι ο τύπος της συσκευής είναι `CDAudio` (ακουστικό CD), ανοίγουν τη συσκευή, με τη διαταγή `Open`, και εντολοδοτούν το παίξιμο του μουσικού κομματιού, που έχει σειρά, με τη διαταγή `Play`.

Η διαταγή `Play` γίνεται αποδεκτή από τη συσκευή μόνον αν έχει προηγηθεί η διαταγή `Open`.

β) Για να εκμεταλλευτούμε σωστά τους πόρους του συστήματος θα πρέπει, πριν τερματίσουμε την εφαρμογή μας, να κλείσουμε τις ανοιχτές συσκευές. Γι' αυτό στην υπορουτίνα διαχείρισης συμβάντος που ξεφορτώνει μια φόρμα γράφουμε:

```
Private Sub Form_Unload (Cancel Ás Integer)
    MMControl1.Command = "Close" 'Êëåßóå ôç óõóêåõÞ
End Sub
```

Κατά την εκτέλεση της εφαρμογής, το αντικείμενο πολυμέσων μπορεί να εμφανίζεται ή να είναι αόρατο. Αν επιθυμούμε την αλληλεπίδραση με το χρήστη δίνουμε στις ιδιότητές του **`Enabled`** και **`Visible`** την τιμή `True`. Αν δε θέλουμε ο χρήστης να έχει οποιονδήποτε έλεγχο πάνω στη συσκευή δίνουμε στην ιδιότητα **`Visible`** την τιμή `False` (μια εφαρμογή μπορεί να ελέγχει MCI συσκευές με ή χωρίς την αλληλεπίδραση του χρήστη). Αποκρύπτοντας το αντικείμενο πολυμέσων μπορούμε να το χρησιμοποιήσουμε για να κάνουμε απλές λειτουργίες, όπως να παίξουμε συγκεκριμένα αρχεία τύπου `.wav`. Όταν εμφανίζουμε το αντικείμενο πολυμέσων, τα πλήκτρα του μπορούν να παρουσιάζονται οριζόντια ή κατακόρυφα. Αυτό το ρυθμίζουμε μέσω της ιδιότητας **`Orientation`** (εξορισμού τιμή 0 για οριζόντια διευθέτηση ή τιμή 1 για κατακόρυφη διευθέτηση).

**Άσκηση 17-1.**

Σε μια φόρμα τοποθετούμε ένα αντικείμενο πολυμέσων, το οποίο ονομάζουμε `Applause`. Θέλουμε να δημιουργήσουμε τον κώδικα, ο οποίος να παίζει το περιεχόμενο του αρχείου `C:\Windows\Media\Office97\Applause.wav`, το οποίο περιέχει τον ήχο χειροκροτημάτων. Στο παράθυρο κώδικα γράφουμε:

```
Private Sub Form_Load()
    Applause.DeviceType = "WaveAudio"
    Applause.Visible = False
    Applause.Enabled = False
    Applause.Filename = "C:\Windows\Media\Office97\Applause.wav"
    Applause.Command = "Open"
    Applause.Command = "Play"
End Sub
```

Ο τύπος της συσκευής καθορίστηκε σε `WaveAudio` από την ιδιότητα **`DeviceType`** και το αντικείμενο πολυμέσων αποκρύφθηκε δίνοντας την τιμή `False` στην ιδιότητα **`Visible`**. Η συσκευή `WaveAudio` είναι μια σύνθετη συσκευή και απαιτεί και τον καθορισμό του αρχείου από το οποίο θα αντληθεί ο ήχος, γι' αυτό και δόθηκε η κατάλληλη τιμή στην ιδιότητα **`Filename`**. Τέλος, η συσκευή ανοίχθηκε με τη διαταγή `Open` και ζητήθηκε το παίξιμο του ήχου με τη διαταγή `Play`.

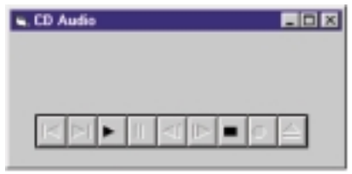
Εκτός από τη δυνατότητα ενεργοποίησης/απενεργοποίησης ολόκληρου του αντικειμένου πολυμέσων, μπορούμε επιλεκτικά να ενεργοποιήσουμε ή να απενεργοποιήσουμε μεμονωμένα πλήκτρα του. Αυτό το πετυχαίνουμε μέσω της ιδιότητας *πλήκτρο***Enabled**, όπου πλήκτρο μπορεί να είναι το συνθετικό **Back**, **Step**, **Play** κλπ ανάλογα με την αγγλική ονομασία του πλήκτρου που περιγράφεται στην εικόνα 17-2. Ανάλογη δυνατότητα έχουμε για την εμφάνιση ή απόκρυψη μεμονωμένων πλήκτρων μέσω της ιδιότητας *πλήκτρο***Visible**. Καθένα από τα 9 πλήκτρα έχει την ιδιότητα **Enabled** και **Visible.** Για παράδειγμα, για το πλήκτρο Back έχουμε τις ιδιότητες **BackEnabled** και **BackVisible**.

Όλες οι λειτουργίες δεν είναι διαθέσιμες σε όλες τις συσκευές. Για παράδειγμα, δεν μπορούμε να χρησιμοποιήσουμε τη διαταγή Record για ένα ακουστικό CD. Για το λόγο αυτό μπορούμε να θέσουμε την ιδιότητα **AutoEnable** σε τιμή True, ώστε η VB να απενεργοποιήσει αυτόματα τα πλήκτρα που αντιστοιχούν σε μη διαθέσιμες λειτουργίες. Η ρύθμιση True της **AutoEnable** αναιρεί όλες τις τιμές *πλήκτρο***Enabled**. Επομένως, για να μπορούμε να έχουμε τον πλήρη έλεγχο των πλήκτρων με τις ιδιότητες *πλήκτρο***Enabled**, η ιδιότητα **AutoEnable** πρέπει να έχει την τιμή False. Όταν το αντικείμενο πολυμέσων είναι ορατό και η ιδιότητά του **AutoEnable** έχει την τιμή True, η διαταγή Open ενεργοποιεί τα πλήκτρα που μπορεί να υποστηρίξει η συγκεκριμένη MCI συσκευή.

**Παράδεγμα 17-2.**

Στον κώδικα που ακολουθεί ενεργοποιούνται μόνον τα πλήκτρα Play και Stop και στο αντικείμενο πολυμέσων με όνομα CDPlayer εκχωρείται μια συσκευή τύπου CDAudio. Για να μπορούμε να έχουμε τον πλήρη έλεγχο με τις ιδιότητες *πλήκτρο***Enabled**, η ιδιότητα. **AutoEnable** τίθεται False.

```
Private Sub Form_Load()
    CDPlayer.DeviceType = "CDAudio"
    CDPlayer.AutoEnable = False
    CDPlayer.BackEnabled = False
    CDPlayer.StepEnabled = False
    CDPlayer.PlayEnabled = True
    CDPlayer.PauseEnabled = False
    CDPlayer.PrevEnabled = False
    CDPlayer.NextEnabled = False
    CDPlayer.StopEnabled = True
    CDPlayer.RecordEnabled = False
    CDPlayer.EjectEnabled = False
    CDPlayer.Command = "Open"
End Sub
```

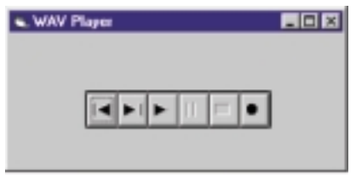



***Εικόνα 17-3.*** *Αντικείμενο πολυμέσων με ενεργοποιημένα και απενεργοποιημένα πλήκτρα*

**Παράδεγμα 17-3.**

Στον κώδικα που ακολουθεί, στο αντικείμενο πολυμέσων με όνομα WavePlayer, ενεργοποιούνται τα πλήκτρα Previous, Next, Play και Stop και αποκρύπτονται τα πλήκτρα Step, Back και Eject.

```
Private Sub Form_Load()
    WavePlayer.DeviceType = "WaveAudio"
    WavePlayer.filename = _
        "C:\WINDOWS\MEDIA\Ctmelody.wav"
    WavePlayer.AutoEnable = True
    WavePlayer.Command = "Open"
End Sub
```




***Εικόνα 17-4.*** *Αντικείμενο πολυμέσων με κρυμμένα πλήκτρα*

Στο παράδειγμά μας, το άνοιγμα της συσκευής ενεργοποιεί τα πλήκτρα Prev, Next, Play, Back και Step. Όταν πατάμε το πλήκτρο Play ενεργοποιείται το πλήκτρο Stop. Ανάλογα, το άνοιγμα μιας CDAudio συσκευής ενεργοποιεί τα πλήκτρα Prev, Next, Play και Eject, ενώ όταν πατάμε το πλήκτρο Play ενεργοποιούνται τα πλήκτρα Stop και Pause.



Αν θέλουμε να ελέγξουμε αν μια ανοιχτή συσκευή έχει ή όχι κάποιες συγκεκριμένες δυνατότητες, ελέγχουμε τις ιδιότητες **CanEject**, **CanPlay**, **CanRecord** και **CanStep.** Για παράδειγμα, η συσκευή CDAudio υποστηρίζει τη διαταγή Eject, που αποβάλλει το CD από τη συσκευή, αλλά η συσκευή WaveAudio δεν την υποστηρίζει, αφού όπως έχουμε ήδη αναφέρει παίζει ήχο από αρχείο, οπότε τι να αποβάλλει.

**Παραδείγματα 17-4.**

Οι εντολές:

```
MMC.StepVisible = MMC.CanStep
MMC.PlayVisible = MMC.CanPlay
MMC.RecordVisible = MMC.CanRecord
MMC.EjectVisible = MMC.CanEject
```

εμφανίζουν ή αποκρύπτουν τα περιττά πλήκτρα, ανάλογα με το είδος της συσκευής που έχουμε εκχωρήσει στο αντικείμενο πολυμέσων MMC.

Πατώντας τα πλήκτρα του αντικειμένου πολυμέσων προκύπτουν τα συμβάντα *πλήκτρο***Click**. Γράφοντας κώδικα στις αντίστοιχες υπορουτίνες διαχείρισης συμβάντων, μπορούμε να βελτιώσουμε την αλληλεπίδραση με τον χρήστη. Για παράδειγμα, να ζητήσουμε επιβεβαίωση της διαταγής και αν χρειαστεί ακύρωσή της. Αυτό το πετυχαίνουμε μέσω της παραμέτρου Cancel. Αν μέσα στην υπορουτίνα διαχείρισης συμβάντος δώσουμε στην παράμετρο Cancel την τιμή True, γίνεται απόρριψη του συμβάντος.

Όταν περατωθεί η λειτουργία που αντιστοιχεί στα πλήκτρα, προκύπτουν συμβάντα *πλήκτρο***Completed**, που μας επιτρέπουν μέσω της παραμέτρου ErrorCode να ελέγξουμε αν κατά την εκτέλεση της λειτουργίας προέκυψε κάποιο λάθος.

**Παραδείγματα 17-5.**

Με την υπορουτίνα διαχείρισης συμβάντος **PlayClick** ζητάμε επιβεβαίωση εκτέλεσης της διαταγής Play, που προέρχεται από το πάτημα του αντίστοιχου πλήκτρου του αντικειμένου πολυμέσων με όνομα MMC. Αν δοθεί θετική απάντηση (πλήκτρο Ok της συνάρτησης MsgBox), διεκπεραιώνεται η διαταγή και με τη συμπλήρωσή της εκτελείται η υπορουτίνα διαχείρισης συμβάντος **PlayCompleted**. Τότε, στην υπορουτίνα διαχείρισης συμβάντος **PlayCompleted**, εξετάζεται η παράμετρος ErrorCode, που μας ενημερώνει για την επιτυχή ή μη επιτυχή έκβαση της διαταγής Play. Αν όμως δοθεί αρνητική απάντηση (πλήκτρο Cancel της συνάρτησης MsgBox), δίνουμε στην παράμετρο Cancel την τιμή True και ακυρώνουμε τη διαταγή Play.

```
Private Sub MMC_PlayClick(Cancel As Integer)
    If MsgBox("ÐâÔÞóáôå ôï ðëÞêôñï", vbOKCancel)= vbOK Then
        Cancel = False
    Else
        MsgBox "Äéáêïðþ ôçò åêôÝëåóçò"
        Cancel = True
    End If
End Sub

Private Sub MMC_PlayCompleted(Errorcode As Long)
    If ErrorCode <> 0 Then
        MsgBox "ÐñïÝêõøå óöÜëìá êáôÜ ôçí åêôÝëåóç."
    Else
        MsgBox "Η äéáôáãÞ åêôåëÝóôçêå åðéôõ÷þò."
    End If
End Sub
```

Πολλές διαταγές MCI χρειάζονται ένα απροσδιόριστο χρονικό διάστημα για να ολοκληρώσουν τη λειτουργία τους. Για παράδειγμα, το παίξιμο ενός αρχείου ήχου απαιτεί χρόνο που εξαρτάται από τη διάρκεια του ψηφιοποιημένου ήχου. Αν θέλουμε να μας γνωστοποιηθεί το πέρας μιας λειτουργίας που αντιστοιχεί σε διαταγή MCI, δίνουμε στην ιδιότητα **Notify** την τιμή

True, πριν ζητήσουμε την εκτέλεση της διαταγής. Τότε μόνον, με την ολοκλήρωση της διαταγής προκαλείται το συμβάν **Done** και εκτελείται η αντίστοιχη υπορουτίνα. Η υπορουτίνα αυτή μέσω της παραμέτρου NotifyCode μας ενημερώνει για την επιτυχή ή μη έκβαση της διαταγής. Οι δυνατές τιμές της παραμέτρου NotifyCode και οι αντίστοιχες σταθερές τους είναι:

| | | |
|---|---|---|
| 1 | mciSuccessful | Η λειτουργία εκτελέστηκε ομαλώς |
| 2 | mciSuperseded | Η διαταγή εκτοπίστηκε από άλλη |
| 4 | mciAborted | Η διαταγή διακόπηκε από τον χρήστη |
| 8 | mciFailure | Η λειτουργία της διαταγής απέτυχε |

**Άσκηση 17-2.**

Να δημιουργηθεί φόρμα στην οποία το πάτημα ενός πλήκτρου διαταγής να προκαλεί το παίξιμο ενός βίντεο. Όταν ολοκληρωθεί το παίξιμο του βίντεο να εμφανίζεται σε διαλογικό παράθυρο το μήνυμα "ÔÝëïò".

Δημιουργούμε πλήκτρο διαταγής με όνομα PlayVideo και αντικείμενο πολυμέσων με όνομα MMC και γράφουμε τον κώδικα:

```
Private Sub Form_Load()
    MMC.DeviceType = "AVIVideo"
    MMC.Filename = "C:\program files\devstudio\vb\graphics\avis\count16.avi"
    MMC.Visible = False
End Sub
Private Sub PlayVideo_Click()
    MMC.Command = "Open"
    MMC.Notify = True
    MMC.Command = "Play"
End Sub
Private Sub MMC_Done(NotifyCode As
                     Integer)
    MsgBox "ÔÝëïò", , "Done"
End Sub
Private Sub Form_Unload(Cancel As
                        Integer)
    MMC.Command = "Close"
End Sub
```


***Εικόνα 17-5.*** *Το βίντεο του αρχείου count16.avi παρουσιάζεται σε ξεχωριστό παράθυρο.*

Σημαντικό συμβάν του αντικειμένου πολυμέσων είναι το **StatusUpdate**, που λειτουργεί σαν το συμβάν **Timer** του χρονομέτρου. Η VB καλεί συνεχώς την υπορουτίνα διαχείρισης συμβάντος **StatusUpdate** σε τακτά χρονικά διαστήματα, τα οποία ορίζουμε από την ιδιότητα **UpdateInterval**. Με αυτό το συμβάν είναι δυνατόν να ανιχνεύουμε την κατάσταση που βρίσκεται η συσκευή MCI, ελέγχοντας την ιδιότητα **Mode**. Οι δυνατές τιμές αυτής της ιδιότητας και οι αντίστοιχες σταθερές τους είναι:

| | | |
|---|---|---|
| 524 | mciModeNotOpen | Η συσκευή δεν είναι ανοιχτή |
| 525 | mciModeStop | Έχει γίνει διακοπή από διαταγή Stop |
| 526 | mciModePlay | Η συσκευή βρίσκεται σε λειτουργία |
| 527 | mciModeRecord | Γίνεται εγγραφή (ηχογράφηση, βιντεοσκόπηση) |
| 528 | mciModeSeek | Η μετάβαση σε νέα θέση βρίσκεται σε εξέλιξη |
| 529 | mciModePause | Έχει γίνει διακοπή από διαταγή Pause |
| 530 | mciModeReady | Η συσκευή είναι έτοιμη |

Η ιδιότητα **TimeFormat** καθορίζει τον τρόπο αναφοράς των μονάδων χρόνου που χρησιμοποιούνται από τις ιδιότητες του αντικειμένου πολυμέσων. Π.χ. για την ιδιότητα **Position** (θέση από την αρχή του κομματιού που παίζεται) και για την ιδιότητα **Length** (μέγεθος του κομματιού).

| | | |
|---|---|---|
| 0 | mciFormatMilliseconds | Χιλιοστά του δευτερολέπτου |
| 1 | mciFormatHms | Ώρες, λεπτά, δευτερόλεπτα |
| 2 | mciFormatMsf | Λεπτά, δευτερόλεπτα, πλαίσια |
| 3 | mciFormatFrames | Πλαίσια |



**Άσκηση 17-2.**

Να δημιουργηθεί φόρμα με την οποία να είναι δυνατή η επιλογή του αρχείου πολυμέσων, που το περιεχόμενο θέλουμε να ακούσουμε (αν είναι αρχείο ήχου) ή να δούμε (αν είναι αρχείο βίντεο). Αν επιλεγεί αρχείο ήχου να παρουσιάζεται η διάρκειά του σε χιλιοστά του δευτερολέπτου και αν είναι αρχείο βίντεο να παρουσιάζεται η διάρκειά του σε πλήθος πλαισίων.

1. Σε μια φόρμα τοποθετούμε ένα πλαίσιο (frame) και στο εσωτερικό του δύο πλήκτρα επιλογής, το OptWav και το OptAvi. Όταν επιλέγεται το πρώτο θα δηλώνεται ότι θέλουμε να παίζεται ήχος και όταν επιλέγεται το δεύτερο θα δηλώνεται ότι θέλουμε να παίζεται βίντεο.
2. Τοποθετούμε ένα αντικείμενο κοινού διαλογικού παραθύρου, στο οποίο δίνουμε το όνομα Dialog. Το διαλογικό παράθυρο που θα ανοίγεται θα είναι παράθυρο ανοίγματος αρχείου με κατάλληλο φίλτρο π.χ. *.wav ή *.avi.
3. Τοποθετούμε ένα αντικείμενο πολυμέσων με όνομα MMC. Από τα πλήκτρα αυτού του αντικειμένου θα παίζεται το περιεχόμενο του αρχείου, που θα επιλέγεται κάθε φορά.
4. Τοποθετούμε μια ετικέτα με όνομα Status. Την ιδιότητα **Caption** αυτής της ετικέτας θα την ενημερώνουμε κάθε φορά που εκτελείται η υπορουτίνα διαχείρισης του συμβάντος **StatusUpdate**.
5. Γράφουμε τον κώδικα:

```
Private Sub OptAvi_Click()
   Dialog.Filter = "all files (*.*)|*.*|.AVI files|*.AVI"
   Dialog.FilterIndex = 2  'Ðáñïõóßáóå ôï äåýôåñï óôïé÷åßï (AVI)
   Dialog.ShowOpen         'Äéáëïãéêü ðáñÜèõñï Open
   MMC.DeviceType = "AVIVideo"
   MMC.filename = Dialog.Filename
   MMC.UpdateInterval = 5  '5 msecs
   MMC.TimeFormat = mciformatframes
   MMC.Command = "Ïpen"
End Sub
Private Sub OptWav_Click()
   Dialog.Filter = "all files (*.*)|*.*|.WAV files|*.WAV"
   Dialog.FilterIndex = 2  'Ðáñïõóßáóå ôï äåýôåñï óôïé÷åßï (WAV)
   Dialog.ShowOpen         'Äéáëïãéêü ðáñÜèõñï Open
   MMC.DeviceType = "WaveAudio"
   MMC.filename = Dialog.Filename
   MMC.UpdateInterval = 5  '5 msecs
   MMC.TimeFormat = mciformatmilliseconds
   MMC.Command = "Ïpen"
End Sub
Private Sub MMC_StatusUpdate()
   If OptAvi Then
      Status.Caption = "Frame " & _
      MMC.Position & " of " & MMC.Length
   Else
      Status.Caption = "Milliseconds:" & _
      MMC.Position & " of " & MMC.Length
   End If
End Sub
Private Sub Form_Unload(Cancel As Integer)
    MMC.Command = "Close"
End Sub
```


***Εικόνα 17-6.*** *Το παράθυρο της εφαρμογής ενώ παίζεται αρχείο ήχου.*

## Σύνδεση και Ενσωμάτωση Αντικειμένων

Έστω ότι σε μια εφαρμογή πολυμέσων θέλουμε να δημιουργήσουμε μια περιοχή, στην οποία οι χρήστες θα μπορούν να ζωγραφίζουν. Αντί να γράψουμε κώδικα που υλοποιεί όλες τις λειτουργίες ενός ήδη υπάρχοντος προγράμματος μπορούμε να την περιγράψουμε ως περιοχή, που εξυπηρετείται από το πρόγραμμα της Ζωγραφικής (Paint). Στην περίπτωση λοιπόν που

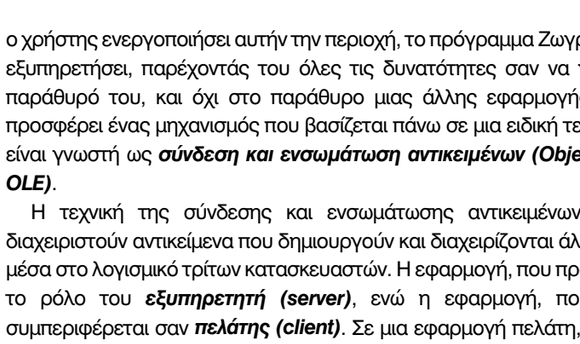
ο χρήστης ενεργοποιήσει αυτήν την περιοχή, το πρόγραμμα Ζωγραφική θα αναλαμβάνει να τον εξυπηρετήσει, παρέχοντάς του όλες τις δυνατότητες σαν να γινόταν η ζωγραφική σε ένα παράθυρό του, και όχι στο παράθυρο μιας άλλης εφαρμογής. Αυτήν τη δυνατότητα την προσφέρει ένας μηχανισμός που βασίζεται πάνω σε μια ειδική τεχνολογία λογισμικού, η οποία είναι γνωστή ως ***σύνδεση και ενσωμάτωση αντικειμένων (Object Linking and Embedding - OLE)***.

Η τεχνική της σύνδεσης και ενσωμάτωσης αντικειμένων βοηθά τις εφαρμογές να διαχειριστούν αντικείμενα που δημιουργούν και διαχειρίζονται άλλες εφαρμογές ή περιέχονται μέσα στο λογισμικό τρίτων κατασκευαστών. Η εφαρμογή, που προσφέρει το αντικείμενο, παίζει το ρόλο του ***εξυπηρετητή (server)***, ενώ η εφαρμογή, που κάνει χρήση του μέσου, συμπεριφέρεται σαν ***πελάτης (client)***. Σε μια εφαρμογή πελάτη, που κάνει εκμετάλλευση της τεχνολογίας σύνδεσης και ενσωμάτωσης αντικειμένων, μπορούμε να δούμε αντικείμενα, τα οποία ίσως να μην μπορούσαν να παρουσιαστούν με άλλο τρόπο ή να χειριστούμε αντικείμενα με πολύ απλό τρόπο χωρίς να χρειαστεί να επέμβουμε προγραμματιστικά.

Στην τεχνολογία σύνδεσης και ενσωμάτωσης αντικειμένων υπάρχουν δύο μηχανισμοί λειτουργίας. Ο ένας εξυπηρετεί την ενσωμάτωση και ο άλλος τη σύνδεση των αντικειμένων. Αν και ο χρήστης, κατά την εκτέλεση του προγράμματος πελάτη, δεν μπορεί να διακρίνει κάποια διαφορά μεταξύ των μηχανισμών, αυτή υπάρχει και είναι σημαντική. Με την ενσωμάτωση ενός αντικειμένου, το αντικείμενο που συνήθως βρίσκεται σε αρχείο, (π.χ. ένα αρχείο εικόνας), αντιγράφεται μέσα στην εφαρμογή πελάτη και εγκλωβίζεται (γίνεται στατικό μέλος της εφαρμογής). Με τη σύνδεση, όταν η εφαρμογή πελάτης χρειαστεί το αντικείμενο, το καλεί μέσα από το αρχέτυπο αρχείο. Η σύνδεση επομένως έχει μεγαλύτερη δυναμική. Το ίδιο αρχείο μπορούν να το εκμεταλλευτούν πολλές εφαρμογές που παίζουν το ρόλο του πελάτη και αν τυχόν γίνουν για κάποιο λόγο επάνω του αλλαγές από μία εφαρμογή, οι αλλαγές αντικατοπτρίζονται και στις υπόλοιπες εφαρμογές. Αντίθετα, αλλαγές σε ένα εγκλωβισμένο αντικείμενο φαίνονται μόνον μέσα στην εφαρμογή που το έχει εγκλωβίσει.

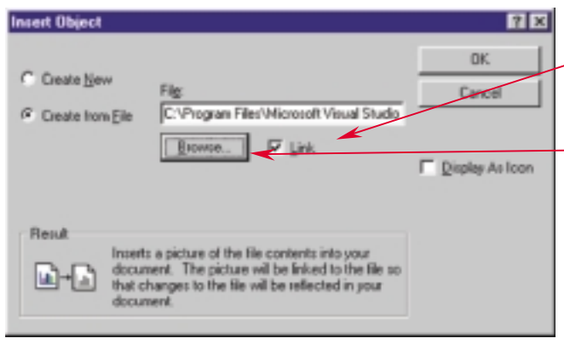
Για να εισάγουμε ένα αντικείμενο OLE σε μια φόρμα:


Το εργαλείο OLE




1. Επιλέγουμε από την εργαλειοθήκη το εργαλείο OLE και σχεδιάζουμε μια ορθογώνια περιοχή.
2. Από τη λίστα **Object Type** επιλέγουμε τον τύπο του αντικειμένου που θα υποδεχτεί το αντικείμενο OLE.
3. Κάνουμε την επιλογή **Create from File** οπότε το παράθυρο μετασχηματίζεται σε αυτό της εικόνας 17-8.

***Εικόνα 17-7.*** *Το διαλογικό παράθυρο για την επιλογή τύπου αντικειμένου*




4. Επιλέγουμε σύνδεση αντικειμένου σημειώνοντας στο τετραγωνίδιο του πλήκτρου σημείωσης **Link**. Αν δεν κάνουμε αυτήν την επιλογή θεωρείται ότι θα γίνει ενσωμάτωση του αντικειμένου.
5. Από το πλήκτρο **Browse** εμφανίζουμε το παράθυρο επιλογής αρχείων, από το οποίο επιλέγουμε και αρχείο.

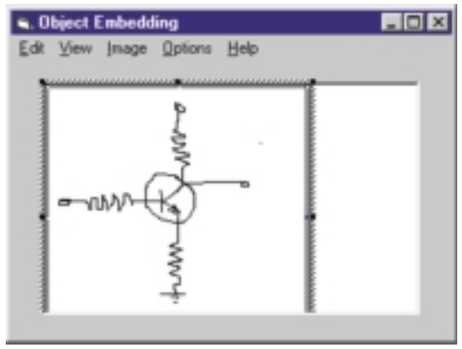
***Εικόνα 17-8.*** *Το διαλογικό παράθυρο για την επιλογή τύπου αντικειμένου*

Η ενεργοποίηση των αντικειμένων OLE γίνεται κατά τη διάρκεια εκτέλεσης του προγράμματος με διπλό κλικ πάνω στο αντικείμενο.



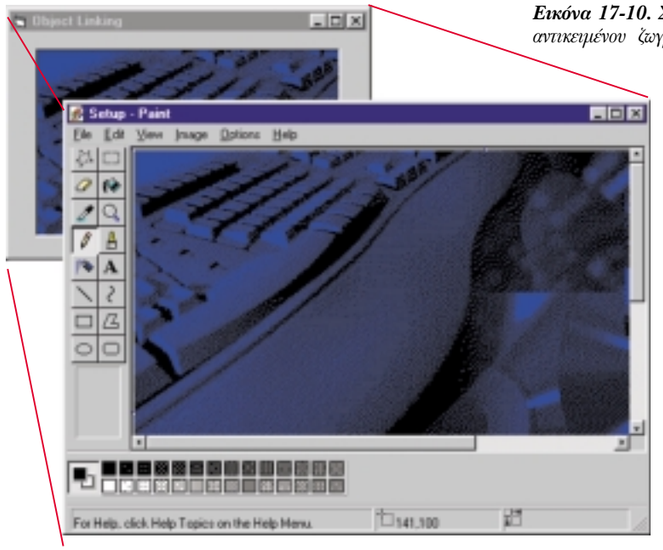
**Άσκηση 17-4.**

α) Σε μια φόρμα τοποθετούμε ένα αντικείμενο OLE τύπου Bitmap Object (ψηφιογραφικής σχεδίασης). Επιλέγουμε την τεχνική ενσωμάτωσης του αντικειμένου. Αν κατά τη διάρκεια εκτέλεσης του προγράμματος κάνουμε διπλό κλικ πάνω στο αντικείμενο, η περιοχή σχεδίασης τονίζεται με ένα πλαίσιο και στο πάνω μέρος του παραθύρου εμφανίζεται ένα μενού με τις βασικές επιλογές του προγράμματος της Ζωγραφικής.




***Εικόνα 17-9.*** *Ενσωμάτωση αντικειμένου ζωγραφικής*

β) Σε μια φόρμα τοποθετούμε ένα αντικείμενο OLE τύπου Bitmap Object. Επιλέγουμε την τεχνική σύνδεσης με αρχείο. Αν κατά τη διάρκεια εκτέλεσης του προγράμματος κάνουμε διπλό κλικ πάνω στο αντικείμενο, εμφανίζεται το παράθυρο του προγράμματος της Ζωγραφικής, δηλαδή του προγράμματος που είναι συνδεδεμένο με την περιοχή σχεδίασης.




***Εικόνα 17-10.*** *Σύνδεση με αρχείο αντικειμένου ζωγραφικής*

Η τεχνική OLE δεν εφαρμόζεται μόνο στη σύνδεση και ενσωμάτωση αντικειμένων πολυμέσων αλλά και άλλων τύπων αντικειμένων. Για παράδειγμα, σε μια φόρμα είναι δυνατή η ένθεση ενός εγγράφου επεξεργαστή κειμένου ή ενός φύλλου εργασίας ενός υπολογιστικού φύλλου κ.ά.

## Ανακεφαλαίωση

Τα Windows παρέχουν τη διεπαφή ελέγχου μέσων, μέσω της οποίας ο προγραμματιστής μπορεί να συμπεριλάβει στις εφαρμογές του ήχο, βίντεο, κινούμενο σχέδιο. Η ονοματολογία και οι διαταγές προς τις συσκευές είναι τυποποιημένες και γίνεται με τη βοήθεια του αντικειμένου πολυμέσων. Το αντικείμενο πολυμέσων περιλαμβάνει ένα σύνολο από πλήκτρα, τα οποία

αντιστοιχούν στις λειτουργίες που διακρίνουμε σε μια τυπική συσκευή κασετοφώνου ή σε μια τυπική συσκευή βίντεο της τηλεόρασης. Τυπικές συσκευές που μπορεί να χειριστεί το αντικείμενο πολυμέσων είναι οι `CDAudio`, `WaveAudio`, `Sequencer`, `Dat`, `Animation`, `VCR`, `VideoDisk`, `AVIVideo`. Εφαρμογές με πολυμεσικές δυνατότητες μπορούμε να δημιουργήσουμε και με τη βοήθεια του μηχανισμού σύνδεσης και ενσωμάτωσης αντικειμένων. Με την ενσωμάτωση ενός αντικειμένου, το αντικείμενο, που συνήθως βρίσκεται σε αρχείο, αντιγράφεται μέσα στην εφαρμογή πελάτη και εγκλωβίζεται. Με τη σύνδεση, όταν η εφαρμογή πελάτης χρειαστεί το αντικείμενο, το καλεί μέσα από το αρχέτυπο αρχείο.

## Εργαστηριακές Ασκήσεις

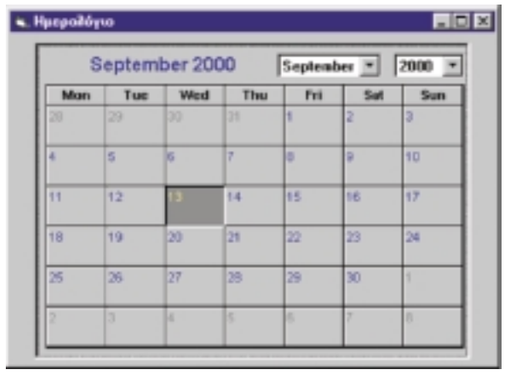
1. Δημιουργία ενός απλού χειριστηρίου για ακουσικά CD. Με την προϋπόθεση ότι στον υπολογιστή υπάρχει κάρτα ήχου με ηχεία και συσκευή CD-ROM, τοποθετήστε πάνω σε μια φόρμα ένα αντικείμενο πολυμέσων. Γράψτε πρόγραμμα ώστε το αντικείμενο πολυμέσων να εξομοιώνει την πρόσοψη ελέγχου ενός ακουστικού CD.
2. Πραγματοποιήστε βήμα βήμα την άσκηση 17-1.
3. Πραγματοποιήστε βήμα βήμα την άσκηση 17-2. Μετασχηματίστε την άσκηση, ώστε να είναι δυνατό και το παίξιμο αρχείων MIDI.
4. Τροποιήστε την άσκηση 17-3 ώστε αντί για μια ετικέτα ως ενδείκτη της προόδου στην εκτέλεση του κομματιού πολυμέσων να χρησιμοποιείται μια ράβδος κύλισης.


5. Η ιδιότητα **`From`** και η ιδιότητα **`To`** του αντικειμένου πολυμέσων συντάσσονται ως:

   *αντικείμενο*.**`From`** = *αρχή*

   *αντικείμενο*.**`To`** = *τέλος*

   και καθορίζουν, στη μονάδα μέτρησης που έχει προκαθοριστεί από την ιδιότητα **`TimeFormat`**, την *αρχή* από την οποία θα αρχίσει να παίζεται ένα πολυμεσικό κομμάτι (ήχος ή βίντεο) και το *τέλος*, στο οποίο θα τερματιστεί το παίξιμό του. Να γραφεί πρόγραμμα στο οποίο να γίνεται η επιλογή ενός κομματιού βίντεο και να καθορίζεται το αρχικό και το τελικό πλαίσιο που θα παρουσιαστεί.
6. Εκτελέστε την άσκηση 17-4. Στο δεύτερο μέλος της άσκησης πραγματοποιήστε αλλοιώσεις πάνω στο αντικείμενο σχεδίασης και αποθηκεύστε το αποτέλεσμα σε αρχείο. Ανοίξτε το αρχείο από την εφαρμογή που το υποστηρίζει.
7. Χρησιμοποιώντας το εργαλείο ένσωμάτωσης και σύνδεσης, δημιουργήστε οθόνη που να προβάλλει ένα ημερολόγιο (Calendar).






# Μάθημα 18
# Πίνακες (Συστοιχίες)

Τελειώνοντας αυτό το μάθημα οι μαθητές θα μπορούν:

- Να επεξηγούν τη διαφορά μεταξύ των πινάκων (arrays) και των απλών μεταβλητών.
- Να επεξηγούν τη σημασία των δεικτών στους πίνακες.
- Να πραγματοποιούν δηλώσεις πινάκων.
- Να χρησιμοποιούν πινάκες για την κωδικοποίηση δεδομένων όμοιας μορφής και μεγάλου όγκου.

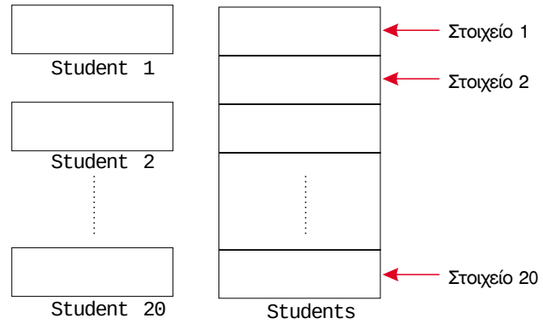
Στην πράξη, όταν θέλουμε να αναφερθούμε σε ένα σύνολο ομοειδών αντικειμένων χρησιμοποιούμε ένα γενικό όνομα, όταν όμως θέλουμε να αναφερθούμε σε ένα συγκεκριμένο στοιχείο του συνόλου, επισυνάπτουμε στο γενικό όνομα και ένα δεύτερο συνθετικό, που προσδιορίζει επακριβώς το στοιχείο. Για παράδειγμα, όλα τα τμήματα της δευτέρας λυκείου, έχουν το γενικό όνομα Β, για να αναφερθούμε όμως σε ένα από αυτά θέτουμε επιπλέον και έναν αριθμητικό δείκτη. Έτσι, με $B_1$ συμβολίζουμε το πρώτο τμήμα της δευτέρας λυκείου, με $B_2$ το δεύτερο τμήμα, με $B_3$ το τρίτο κτλ.

Όπως έχουμε δει για την αποθήκευση τιμών μέσα σε ένα πρόγραμμα χρησιμοποιούμε μεταβλητές. Στην περίπτωση που έχουμε ένα μεγάλο πλήθος ομοειδών μεταβλητών δεν είναι αναγκαίο να ορίσουμε και ένα μοναδικό όνομα για κάθε μία από αυτές. Η VB μας δίνει τη δυνατότητα να χρησιμοποιούμε το ίδιο γενικό όνομα για ένα σύνολο μεταβλητών και να αναφερόμαστε σε μια συγκεκριμένη από αυτές, γράφοντας το γενικό όνομα και έναν αριθμητικό ***δείκτη (index)***. Οι μεταβλητές αυτού του τύπου ονομάζονται ***μεταβλητές με δείκτη*** και συνθέτουν τις επονομαζόμενες ***συστοιχίες*** ή αλλιώς ***πίνακες (arrays).***

Έστω, ότι μέσα σε ένα πρόγραμμα θέλουμε να διαχειριστούμε τα ονόματα των 20 μαθητών μιας τάξης.Χρησιμοποιώντας απλές μεταβλητές και σύμφωνα με όσα γνωρίζουμε μέχρι τώρα, θα έπρεπε να δηλώσουμε τόσα ονόματα μεταβλητών, όσα αντιστοιχούν στο πλήθος των μαθητών.

Δημιουργώντας όμως έναν πίνακα, αρκεί να δηλώσουμε ένα μόνο όνομα. Τα στοιχεία του πίνακα μπορούν να προσδιοριστούν από έναν δείκτη που υποδηλώνει την αύξουσα σειρά τους μέσα στον πίνακα.

Η χρήση των πινάκων συμβάλλει στην απλοποίηση της κωδικοποιήσης των μεταβλητών μέσα σε ένα πρόγραμμα, κάνει τα προγράμματα πιο γενικά και δίνει λύσεις σε προβλήματα, τα οποία δεν μπορούν να λυθούν με τη χρήση απλών μεταβλητών.




***Σχήμα 18-1.*** *Απλές μεταβλητές και πίνακας*

## Δήλωση πίνακα. Αναφορά στοιχείων πίνακα

Τα ονόματα των πινάκων, ως ονόματα μεταβλητών, υπακούουν στους ίδιους ακριβώς κανόνες, στους οποίους υπακούουν και τα ονόματα των απλών μεταβλητών. Δηλαδή, πρέπει να αρχίζουν από γράμμα, να μη συντίθενται από περισσότερους από 255 χαρακτήρες, να μη συμπίπτουν με δεσμευμένες λέξεις. Πριν χρησιμοποιήσουμε έναν πίνακα στο πρόγραμμα, πρέπει πρώτα να τον δηλώσουμε, ώστε η VB να δεσμεύσει τον κατάλληλο χώρο στη μνήμη και να καθορίσει τον τρόπο με τον οποίο θα χειριστεί τα στοιχεία του. Αμέσως μετά το όνομα του πίνακα, και μέσα σε παρενθέσεις, δηλώνουμε τη μέγιστη τιμή που μπορεί να πάρει ο δείκτης. Συγκεκριμένα, η δήλωση ενός πίνακα γίνεται με τη δηλωτική εντολή `Dim` ως εξής:

`Dim` *όνομα_μεταβλητής(μέγιστος_δείκτης)* `As` *τύπος*

όπου όνομα_μεταβλητής, το όνομα του πίνακα, μέγιστος_δείκτης η μεγαλύτερη τιμή του δείκτη και τύπος, ένας από τους τύπους δεδομένων.

**Παράδειγμα 18-1.**

Έστω η περίπτωση των 20 μαθητών που αναφέραμε πιο πάνω. Χρησιμοποιώντας απλές μεταβλητές πρέπει να κάνουμε τις δηλώσεις:

```
Dim Student1      As String
Dim Student2      As String
Dim Student3      As String
       :      :      :      :
Dim Student20     As String
```

Χρησιμοποιώντας όμως ένα γενικό όνομα για έναν πίνακα, αρκεί να γράψουμε:

```
Dim Students(20)   As String
```

Είναι λοιπόν προφανές πόσο δύσχρηστο μπορεί να γίνει ένα πρόγραμμα, αν για όμοια δεδομένα χρησιμοποιήσουμε διαφορετικά ονόματα μεταβλητών και πόσο πιο μικρό και περιεκτικό, αν χρησιμοποιήσουμε έναν πίνακα. Στη συγκεκριμένη περίπτωση, ήδη από την περιοχή των δηλώσεων, έχουμε κερδίσει 19 γραμμές κώδικα και φανταστείτε πόσο θα κερδίζαμε, αν ήταν ανάγκη να επεξεργαστούμε τα στοιχεία όλων των μαθητών μιας σχολικής περιφέρειας ή και ολόκληρης της Ελλάδος! Η ανεξάρτητη δήλωση μεταβλητών θα ήταν ανεφάρμοστη και μόνο η δήλωση με πίνακα θα ήταν υλοποιήσιμη.

Στο μάθημα 2 είχαμε τονίσει τη σημασία της σωστής δομής δεδομένων

Η δήλωση ενός πίνακα μπορεί να γίνει και με μια από τις δηλωτικές εντολές `Private`, `Public`, `Static`. Το ποια από αυτές θα χρησιμοποιηθεί εξαρτάται από την εμβέλεια που θέλουμε να προσδώσουμε στον πίνακα. Όπως έχουμε δει οι κωδικές λέξεις `Private` και `Dim` είναι ισοδύναμες. Η κωδική λέξη `Public` χρησιμοποιείται, όταν θέλουμε ο πίνακας να έχει καθολικό χαρακτήρα και να είναι δυνατή η χρήση των στοιχείων του μέσα σε όλο το πρόγραμμα. Η κωδική λέξη `Static` διατηρεί το περιεχόμενο των στοιχείων του πίνακα μεταξύ των κλήσεων της υπορουτίνας.

Για τον τρόπο δήλωσης, την εμβέλεια και τον τύπο των πινάκων ισχύουν όσα έχουμε αναφέρει στο μάθημα 6 περί μεταβλητών.

Η μέγιστη τιμή δείκτη επιλέγεται με κριτήριο το πλήθος των στοιχείων του πίνακα. Για να είναι γενικό ένα πρόγραμμα και για να μην χρειάζεται να γίνονται αλλαγές κάθε φορά που πρόκειται να χρησιμοποιηθεί με διαφορετικό πλήθος δεδομένων, συνήθως επιλέγεται μια ικανοποιητικά μεγάλη τιμή ως τιμή μέγιστου δείκτη, που είναι αρκετή να καλύψει ένα μεγάλο πλήθος περιπτώσεων για τις οποίες πρόκειται να χρησιμοποιηθεί ένα πρόγραμμα. Για παράδειγμα, αν το πρόγραμμα του πιο πάνω παραδείγματος πρόκειται να καλύψει τις ανάγκες μιας μόνο τάξης, ο μέγιστος δείκτης μπορεί να έχει τιμή μέχρι 40, αν πρόκειται όμως να καλύψει τις ανάγκες ενός σχολείου ο μέγιστος δείκτης μπορεί να πάρει τιμή μέχρι 1000, που σημαίνει ότι είναι δυνατόν να καλύπτει και τις ανάγκες μιας τάξης κ.ο.κ. Θα έλεγε λοιπόν κανείς, ότι πρέπει να επιλέγουμε μεγάλες τιμές για να μπορεί ένα πρόγραμμα να καλύπτει από τις πιο γενικές μέχρι και τις πιο ειδικές περιπτώσεις. Όμως, όπως θα δούμε στην επόμενη παράγραφο, κάτι τέτοιο δεν είναι δυνατόν.

Κάθε πρόγραμμα, όπως και κάθε κατασκευή, δημιουργείται να να μπορεί να καλύψει ανάγκες που έχουν προδιαγραφεί.

Μπορούμε να δημιουργήσουμε πίνακα οποιουδήποτε τύπου δεδομένων, για παράδειγμα `Boolean`, `Integer`, `Long`, `Single`, `String` κλπ. Το μέγεθος μνήμης, που δεσμεύεται κατά τη δήλωση ενός πίνακα, είναι ακέραιο πολλαπλάσιο του πλήθους των bytes, που δεσμεύονται για κάθε τύπο μεταβλητής και εξαρτάται από την τιμή του μέγιστου δείκτη. Γι' αυτό και δεν πρέπει να δηλώνεται τιμή μέγιστου δείκτη πολύ μεγαλύτερη από αυτήν που πρόκειται να χρησιμοποιηθεί στην χειρότερη των περιπτώσεων.

**Παραδείγματα 18-2.**

α) Σε ένα δίκτυο υπάρχουν 300 συνδεδεμένοι υπολογιστές. Σε ένα πρόγραμμα διαχείρισης των υπολογιστών, ο πίνακας που περιέχει τις IP διευθύνσεις τους μπορεί να δηλωθεί ως:

```
Dim WorkStationIP(300)    As String
```

Παρ' όλο που την IP διεύθυνση τη θεωρούμε αριθμό, πρέπει να τη χειριστούμε σαν συμβολοσειρά μέσα στο πρόγραμμα, αφού αποτελείται από τέσσερα αριθμητικά τμήματα που χωρίζονται με τελείες.

β) Σε ένα πρόγραμμα ζητείται να αποθηκεύσουμε τις τιμές 30 αντιστάσεων που υπάρχουν σε ένα ηλεκτρονικό κύκλωμα. Κάνουμε τη δήλωση:

```
Dim Resistance(70)        As Single
```



Ας σημειωθεί ότι τον μέγιστο δείκτη του πίνακα, τον επιλέξαμε κατά πολύ μεγαλύτερο των απαιτήσεων του προβλήματος, για να είναι δυνατή η χρήση του προγράμματος και σε μια μελλοντική αύξηση των απαιτήσεων.

Η αναφορά σε ένα στοιχείο του πίνακα γίνεται με τη βοήθεια αριθμητικού δείκτη. Ο δείκτης ενός πίνακα γράφεται αμέσως μετά το όνομα του και μέσα σε παρενθέσεις. Ο δείκτης δεν αποτελεί τμήμα του ονόματος αλλά ένα πρόσθετο χαρακτηριστικό, που διαφοροποιεί ένα στοιχείο του πίνακα από όλα τα άλλα στοιχεία. Ο δείκτης μπορεί να είναι ακέραια αριθμητική σταθερά, μεταβλητή ή παράσταση.

Ο δείκτης ενός πίνακα δεν μπορεί να γραφτεί στο κάτω δεξιό μέρος του ονόματος για καθαρά τεχνικούς λόγους

**Παράδειγμα 18-3.**

Συνεχίζοντας την ανάλυση του προβλήματος της κωδικοποίησης των ονομάτων των μαθητών, θέλουμε να αναφερθούμε στο πρόγραμμά μας σε κάποιους μαθητές. Για να αναφερθούμε στον 1ο μαθητή θα γράφαμε:

```
Students(1)
```

επίσης, για να αναφερθούμε στον 2ο μαθητή θα γράφαμε:

```
Students(2)
```

Στην περίπτωση που θέλουμε να αναφερθούμε στον μαθητή i, όπου το i ακέραια αριθμητική μεταβλητή θα γράφαμε:

```
Students(i)
```

Τέλος, στην περίπτωση που θα θέλαμε να αναφερθούμε στον μετά τον i μαθητή θα γράφαμε:

```
Students(i+1)
```

Η αναφορά σε στοιχείο του πίνακα με δείκτη μεγαλύτερο από την τιμή του μεγαλύτερου δείκτη προκαλεί διακοπή του προγράμματος και εμφάνιση του μηνύματος **"Run-time error '9' Subscript out of range".**

## Εισαγωγή, εκτύπωση, επεξεργασία στοιχείων πίνακα

Στις περισσότερες εφαρμογές σήμερα ζητείται η διαχείριση τεράστιων όγκων δεδομένων. Η είσοδος, η επεξεργασία και η εμφάνισή τους απαιτούν ειδικούς τρόπους κωδικοποίησης, ώστε να προκύπτουν μικρά, απλά και εύκολα στη συντήρηση προγράμματα. Η βασική τεχνική πάνω στην οποία στηρίζονται οι τρόποι κωδικοποίησης είναι ο συνδυασμός των πινάκων με βρόχους. Όπως θα δούμε, συσχετίζοντας το μετρητή ενός βρόχου με τον δείκτη ενός πίνακα μπορούμε να επεξεργαστούμε όλα τα στοιχεία του πίνακα μέσα σε ένα βρόχο.

Συνήθως, η εισαγωγή στοιχείων γίνεται από αρχεία ή παράθυρα. Στο επόμενο παράδειγμα χρησιμοποιούμε το απλούστερο παράθυρο εισαγωγής, που είναι το παράθυρο της συνάρτησης `InputBox`. Επίσης, η εκτύπωση στοιχείων γίνεται σε παράθυρα ή στον εκτυπωτή. Στο επόμενο παράδειγμα χρησιμοποιούμε ένα δεύτερο παράθυρο για την παρουσίαση των στοιχείων.

**Παράδειγμα 18-4.**

Για να εισάγουμε τα ονόματα των μαθητών μιας τάξης μέσα σε ένα πρόγραμμα κάνουμε τις δηλώσεις και γράφουμε τον κώδικα:

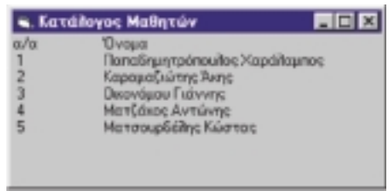
```
'ÄÞëùóç ìåôáâëçôþí
 Dim Students(40) As String    'ÐßíáêáÒ ìáèçôþí
 Dim No           As Integer   'ÐëÞèïò ìáèçôþí ôçò ôÜîçò
 Dim i            As Integer   'Ìåôñçôþò âñü÷ïõ
'Åéóáãùãþ ðëÞèïõò ìáèçôþí
 No = InputBox("ÐëÞèïò ìáèçôþí", "Åéóáãùãþ")
 If No > 40 Then End
'Åéóáãùãþ ïíïìÜôùí ìáèçôþí
 For i = 1 To No
    Students(i) = InputBox("¼íïìá " & i & "ïõ ìáèçôÞ", "Åéóáãùãþ")
 Next i
```

Η πρώτη συνάρτηση `InputBox` μεταφέρει στο πρόγραμμα το πλήθος των μαθητών της τάξης. Αν στη μεταβλητή `No` δοθεί τιμή μεγαλύτερη του 40, που είναι η μεγαλύτερη επιτρεπτή τιμή για τον δείκτη (τόσο έχει δηλωθεί στη δηλωτική εντολή `Dim`), το πρόγραμμα διακόπτεται, ώστε να προστατευτεί η μη ομαλή διακοπή του από μελλοντική υπερχείλιση του πίνακα. Στο δεύτερο βρόχο, για κάθε μαθητή, από τον πρώτο μέχρι και τον τελευταίο, λαμβάνεται το όνομά του και γίνεται η καταχώρισή του στο αντίστοιχο στοιχείο του πίνακα που έχει δηλωθεί ότι θα περιέχει τα ονόματα των μαθητών. Την πρώτη φορά ο μετρητής `i` του βρόχου παίρνει την τιμή 1, άρα με την αναφορά `Students(i)` υπονοείται το στοιχείο `Students(1)` που έχει συσχετιστεί με τον πρώτο μαθητή.


***Εικόνα 18-1.*** *Η InputBox για εισαγωγή ονόματος μαθητή*

Το πιο κάτω τμήμα προγράμματος, ανοίγει το παράθυρο της φόρμας `OutputForm` και εμφανίζει στο εσωτερικό του τα ονόματα των μαθητών που έχουν εισαχθεί με τον κώδικα που

```
'Öüñôùóå ôç öüñìá
OutputForm.Show
'ÀìöÜíéóå ôéò åðéêåöáëßäåò
OutputForm.Print "á/á", "¼íìá"
'ÀìöÜíéóå ôá ïíüìáôá
For i = 1 To No
    OutputForm.Print i, Students(i)
Next i
```




***Εικόνα 18-2.*** *Το παράθυρο με τον κατάλογο των ονομάτων που εμφανίζεται μέσα από το πρόγραμμα.*

Η επεξεργασία των στοιχείων ενός πίνακα γίνεται και αυτή κατά κανόνα με τη βοήθεια βρόχων. Τα στοιχεία του πίνακα μπορούν να προσπελαστούν με τη σειρά που έχουν εισαχθεί, με αντίστροφη σειρά ή με τυχαία αναφορά του δείκτη.

**Παράδειγμα 18-5.**

Έστω, ότι θέλουμε να απομονώσουμε τα ψηφία ενός θετικού ακεραίου αριθμού N και να τα τοποθετήσουμε στα στοιχεία ενός πίνακα με όνομα `Digit`, καταχωρίζοντας το ψηφίο των μονάδων στο στοιχείο 1, των δεκάδων στο στοιχείο 2 κ.ο.κ. Γράφουμε το τμήμα προγράμματος:

```
'ÄÞëùóç ìåôáâëçôþí
 Dim Digit(10)     As Integer     'Ðßíáêáò øçößùí
 Dim N             As Long        'Ï áñéèìüò
 Dim NStr          As String      'Ï áñéèìüò Í óáí óõìâïëïóåéñÜ
 Dim DigitStr      As String * 1  'Øçößï ôïõ áñéèìïý
 Dim i             As Integer     'Ìåôñçôþò âñü÷ïõ
 N = InputBox("Ðëçêôñïëïãþóå áñéèìü", "ÅéóáãùãÞ")
 NStr = Str(N)                    'Ìåôáôñïðþ áñéèìïý óå óõìâïëïóåéñÜ
 For i = 2 To Len(NStr)
     DigitStr = Mid( NStr, i, 1)
     Digit(i - 1) = DigitStr
 Next i
```

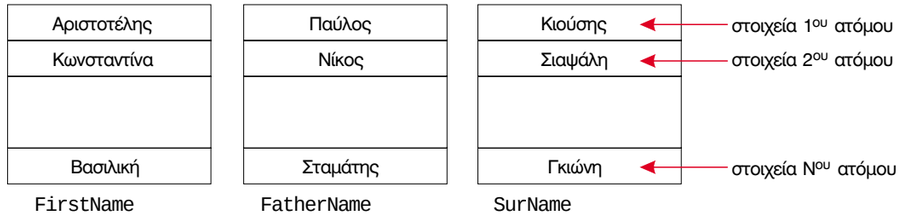
Η συνάρτηση `Str` μετατρέπει τον αριθμό σε συμβολοσειρά, της οποίας ο πρώτος χαρακτήρας είναι το κενό διάστημα. Ο βρόχος, για να αγνοηθεί το κενό διάστημα,



σαρώνει τους χαρακτήρες (ψηφία) της αλφαριθμητικής μεταβλητής `NStr` από τον δεύτερο προς τον τελευταίο και τους τοποθετεί στα στοιχεία του πίνακα από το πρώτο στοιχείο προς το τελευταίο.

**Παράδειγμα 18-6.**

Έστω ότι το όνομα, το πατρώνυμο και το επώνυμο N ατόμων έχουν καταχωριστεί στα στοιχεία των πινάκων `FirstName`, `FatherName` και `SurName`, αντίστοιχα. Π.χ.




***Σχήμα 18-1.*** *Τα στοιχεία του πρώτου ατόμου βρίσκονται στην πρώτη θέση των πινάκων, τα στοιχεία του δεύτερου ατόμου στη δεύτερη θέση των πινάκων κ.ο.κ.*

Το τμήμα προγράμματος:

```
For i = 1 To N
Printer.Print Left( FirstName(i), 1 ) & "." & _
              Left( FatherName(i), 1 ) & "." & _
              SurName(i)
Next i
```

τυπώνει τη σύντμηση των ονομάτων, δηλαδή:

```
Á.Ð.Êéïýóçò
Ê.Í.ÓéáøÜëç
    :
Â.Ó.Ãêéþíç
```

**Άσκηση 18-1.**

Έστω, ότι μας δίνουν τις θερμοκρασίες που σημειώθηκαν μια μέρα σε 30, το πολύ, πόλεις της Ελλάδας στις 12 το μεσημέρι. Μας ζητούν να υπολογίσουμε και να εμφανίσουμε τη μέση θερμοκρασία, προσθέτοντας όλες τις θερμοκρασίες της λίστας και διαιρώντας με το συνολικό αριθμό τους. Γράφουμε τον κώδικα:

```
'ÄÞëùóç ìåôáâëçôþí
 Dim Temp(30)     As Single     'ÌÝ÷ñé 30 èåñìïêñáóßåò
 Dim No           As Integer    'ÐëÞèìü èåñìïêñáóéþí áðü ôïí ÷ñÞóç
 Dim i            As Integer    'Ìåôñçôþò âñü÷ïõ
 Dim TotalTemp    As Single     'Áèñïéóìá èåñìïêñáóéþí
 Dim AvergTemp    As Single     'ÌÝóïò üñïò
'Åéóáãùãþ ðëÞèïõò èåñìïêñáóéþí
 No = InputBox("ÐëÞèïò ðüëåùí", "Åéóáãùãþ")
 If (No < 1) Or (No > 30) Then End
'Åéóáãùãþ èåñìïêñáóéþí
 For i = 1 To No
    Temp(i) = InputBox("Èåñìïêñáóßá óôçí ðüëç " & i, "Åéóáãùãþ")
 Next i
'Õðïëïãéóìüò ìÝóïõ üñïõ
 TotalTemp = 0                          'Áñ÷éêü Üèñïéóìá 0
 For i = 1 To No
    TotalTemp  = TotalTemp + Temp(i) 'Ìåãéêü Üèñïéóìá
 Next i
 AvergTemp = TotalTemp / Íi           'ÌÝóïò üñïò
'ÀìöÜíéóç áðïôåëÝóìáôïò
 MsgBox "Ç ìÝóç èåñìïêñáóßá åßíáé " & AvergTemp
```

Στο τμήμα που υπολογίζεται ο μέσος όρος, ξεκινάμε δίνοντας στη μεταβλητή `TotalTemp`, που θα περιέχει το άθροισμα των θερμοκρασιών, την τιμή 0. Σε κάθε ανακύκλωση του βρόχου το άθροισμα αυξάνεται κατά τη θερμοκρασία που σημειώθηκε στην `i` πόλη.

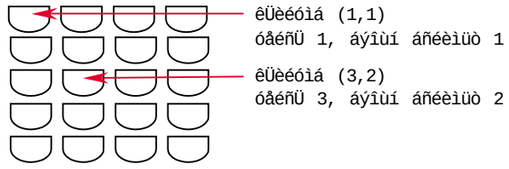
Αφού αθροιστούν οι θερμοκρασίες όλων των πόλεων γίνεται διαίρεση με το πλήθος των πόλεων. Προσέξτε, ότι για να είναι σωστός ο αλγόριθμος και να μην προκληθεί διακοπή του προγράμματος στην περίπτωση που ο χρήστης δώσει πλήθος στοιχείων ίσο με 0 (διαίρεση με το 0 κατά τον υπολογισμό του μέσου όρου), τοποθετήθηκε συνθήκη που τερματίζει το πρόγραμμα στην περίπτωση που δοθεί Νο ίσο με 0.

## Δισδιάστατοι και πολυδιάστατοι πίνακες

Έως τώρα μελετήσαμε πίνακες των οποίων τα στοιχεία μπορούμε να τα φανταστούμε ανεπτυγμένα κατά μια διάσταση (***μονοδιάστατοι πίνακες***) και των οποίων οι μεταβλητές έχουν έναν μόνο δείκτη (Δείτε σχήμα 18-1). Πολλές φορές όμως προκύπτει η ανάγκη να δημιουργησουμε πίνακες που τα στοιχεία τους να είναι αναπτυγμένα σε δύο διαστάσεις. Ένας τέτοιος πίνακας ονομάζεται ***δισδιάστατος*** και οι μεταβλητές του έχουν δύο δείκτες.

**Παραδείγματα 18-7.**

α) Τα καθίσματα ενός θεάτρου είναι διατεταγμένα σε γραμμές και σε στήλες. Σε ένα εισιτήριο θεάτρου η αναφορά στο κάθισμα γίνεται με αναγραφή της σειράς που βρίσκεται το κάθισμα και του αύξοντα αριθμού του καθίσματος μέσα στη σειρά.




***Σχήμα 18-3.*** *Η αρίθμηση αρχίζει από το πάνω αριστερό κάθισμα*

β) Έστω ότι θέλουμε να καταχωρίσουμε τη βαθμολογία των μαθητών μιας τάξης, ώστε σε κάθε γραμμή να υπάρχουν οι βαθμοί ενός μαθητή στα μαθήματα της Φυσικής, της Χημείας, των Νέων ελληνικών και της Ιστορίας.

Στο παράδειγμα 18-6 δεν έχουμε έναν δισδιάστατο πίνακα αλλά τρεις μονοδιάστατους.

| | Φυσική | Χημεία | Νέα | Ιστορία | |
|---|---|---|---|---|---|
| Παύλου | 17 | 18 | 15 | 118 | ← βαθμοί 1ου ατόμου |
| Γκίκας | 12 | 15 | 16 | 13 | ← βαθμοί 2ου ατόμου |
| Κοράλλη | 15 | 16 | 17 | 17 | |
| Οικονόμου | 20 | 18 | 19 | 17 | |
| Γαλιώτος | 18 | 19 | 19 | 20 | ← βαθμοί Νου ατόμου |
| | ↑ 1ο μάθημα | ↑ 2ο μάθημα | | ↑ 5ο μάθημα | |

***Σχήμα 18-4.*** *Η διάταξη ενός καταλόγου μαθητών και βαθμών σε μορφή πίνακα. Στον πίνακα της VB θα εκχωρηθούν μόνον οι βαθμοί. Τα ονόματα των μαθητών και των μαθημάτων δεν ανήκουν στον πίνακα.*

Στον παραπάνω πίνακα φαίνονται πέντε μαθητές και ο βαθμός τους σε κάθε ένα από τα 4 μαθήματα. Η διάταξη που προκύπτει είναι ένας δισδιάστατος πίνακας. Έχει γραμμές, που αντιστοιχούν στην πρώτη διάσταση, και στήλες που αντιστοιχούν στη δεύτερη διάσταση. Τα δεδομένα του διδιάστατου πίνακα πρέπει να είναι του ίδιου τύπου. Στην περίπτωση του παραδείγματος μας είναι ακέραιοι.

Στην VB μπορούμε να ορίσουμε πίνακες όχι μόνο μιας και δύο διαστάσεων αλλά πολλών ***διαστάσεων (multidimensional)***. Το μέγιστο πλήθος των διαστάσεων είναι περίπου 60, αριθμός κάπως υπερβολικός, αφού σπάνια χρειάζεται να χρησιμοποιήσουμε περισσότερες από 2 ή 3 διαστάσεις. Η δήλωση ενός πίνακα πολλών διαστάσεων γίνεται όπως και στην περίπτωση του μονοδιάστατου πίνακα με τις δηλωτικές εντολές `Dim`, `Public`, `Private`, `Static`. Μεταξύ των παρενθέσεων τοποθετούμε μία τιμή για κάθε διάσταση. Ο γενικός τρόπος σύνταξης είναι ο εξής:

`Dim` *όνομα_μεταβλητής(μέγιστος_δείκτης1, μέγιστος_δείκτης2, ...)* `As` *τύπος*



**Παραδείγματα 18-8.**

Ως επέκταση των παραδειγμάτων 18-7:

α) Η δηλωτική εντολή:

```
Dim Seat_Name(30, 15)  As String
```

δημιουργεί πίνακα κατάλληλο για την αποθήκευση των ονομάτων των θεατών που έχουν κλείσει θέσεις σε ένα θέατρο, που έχει 30 σειρές καθισμάτων με 15 καθίσματα σε κάθε σειρά.

β) Η δηλωτική εντολή:

```
Dim Grades(30, 12)      As Integer
```

δημιουργεί πίνακα κατάλληλο για την αποθήκευση των βαθμών 30 μαθητών σε 12 μαθήματα.

Η διαχείριση των πινάκων γίνεται συνήθως με ένθετους βρόχους. Στην περίπτωση που πρόκειται να σαρωθούν όλα τα στοιχεία ενός πολυδιάστατου πίνακα, χρησιμοποιούνται τόσοι βρόχοι όσες είναι και οι διαστάσεις του πίνακα. Σε κάθε μετρητή του βρόχου αντιστοιχίζουμε και ένα δείκτη του πίνακα.

**Παράδειγμα 18-9.**

Για να καταχωρίσουμε τα ονόματα των θεατών αντιστοιχίζοντάς τα με τα στοιχεία ενός δισδιάστατου πίνακα γράφουμε:

```
Dim Seat_Name(30, 15)   As String
Dim Message              As String
For i = 1 To 30
    For j = 1 To 15
       Message = "Óíïìá èåáôÞ: " & i & "ç óåéñÜ " & j & "ç èÝóç."
       Seat_Name(i, j) = InputBox(Message, "Åéóáãùãþ")
    Next i
Next j
```

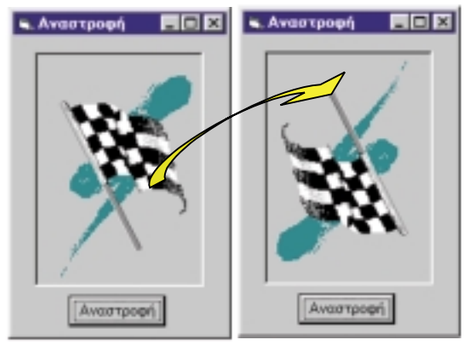
## Το κάτω όριο των δεικτών

Μέχρι αυτό το σημείο είδαμε παραδείγματα στα οποία φαίνεται πώς μπορούμε να δηλώσουμε τη μεγαλύτερη τιμή που μπορεί να πάρει ο δείκτης ενός πίνακα. Όμως, δεν αναφέραμε τίποτε για τη μικρότερη τιμή που μπορεί να πάρει ο δείκτης. Ίσως μάλιστα να σας δημιουργήθηκε η εντύπωση ότι η μικρότερη τιμή μπορεί να είναι πάντα 1. Στην πραγματικότητα η μικρότερη τιμή που μπορεί να πάρει ο δείκτης, εκτός και αν έχει δηλωθεί διαφορετικά, είναι η τιμή 0.

**Άσκηση 18-2.**

Έστω, ότι το όνομα του αρχείου `C:\...\VB\Wizards\Template\Finish.bmp` έχει δοθεί ως τιμή στην ιδιότητα **`Picture`** ενός πλαισίου εικόνας με όνομα `Canvas`. Στην περιοχή του πλαισίου εικόνας θα είναι σχεδιασμένη η εικόνα που βρίσκεται αποθηκευμένη στο αρχείο. Μας ζητούν να γράψουμε πρόγραμμα, το οποίο να αναστρέφει την εικόνα κάθε φοράπου πατείται πλήκτρο με όνομα `Invert`.

Τα εικονοστοιχεία (pixels) μιας εικόνας είναι διατεταγμένα σε στήλες x και σε γραμμές y. Την εικόνα μπορούμε να την αποθηκεύσουμε σε έναν δισδιάστατο πίνακα, αν σε κάθε στοιχείο του πίνακα αντιγράψουμε το χρώμα κάθε εικονοστοιχείου. Η αντιγραφή μπορεί να γίνει αντιγράφοντας το χρώμα του πάνω αριστερά εικονοστοιχείου της εικόνας στο στοιχείο (0,0) του πίνακα, το χρώμα του αμέσως επόμενου προς τα δεξιά εικονοστοιχείου.




***Εικόνα 18-3.*** *Το αποτέλεσμα της αναστροφής*

στο στοιχείο (0,1) του πίνακα κ.ο.κ. μέχρι να εξαντληθούν όλα τα εικονοστοιχεία της γραμμής. Με τον ίδιο τρόπο σαρώνουμε και τα εικονοστοιχεία της δεύτερης γραμμής της εικόνας μεταφέροντας τα: στο (1,0) στοιχείο του πίνακα, στο (1,1) στοιχείο του πίνακα κ.ο.κ.

Αν τα στοιχεία του πίνακα τα τοποθετήσουμε πάλι μέσα στο πλαίσιο της εικόνας, αρχίζοντας την τοποθέτηση από την κάτω δεξιά πλευρά και σαρώνοντας την περιοχή προς τα αριστερά και πάνω, θα γίνει η αναστροφή της εικόνας. Σε μορφή κώδικα θα γράφαμε:

```
Private Sub Invert_Click()

    Dim Pixel(800, 600) As Long      'ÅíäéÜìåóéò ðßíáêáò
    Dim MaxX, MaxY      As Long      'Ïñéæüíôéï ðëÜôïò
    Dim x, y            As Long      'Ääßêôåò óÜñùóçò

    Canvas.ScaleMode = vbPixels      'Ïé äéáóôÜóåéò óå åéêïíïóôïé÷åßá
    MaxX = Canvas.ScaleWidth - 1
    MaxY = Canvas.ScaleHeight - 1

   'ÁíôéãñáöÞ óôïí ðßíáêá
    For y = 0 To MaxY                'Ãéá êÜèå ãñáììÞ
        For x = 0 To MaxX            ' "    "  óôÞëç
            Pixel(x, y) = Canvas.Point(x, y)
        Next x
    Next y
   'Ìåôáöïñ¬ áðü ôïí ðßíáêá êáé áíáóôñïöÞ
     For y = 0 To MaxY
        For x = 0 To MaxX
            Canvas.PSet (x, y), Pixel(MaxX - x, MaxY - y)
        Next x
    Next y

End Sub
```

Στην πιο πάνω άσκηση η τιμή 0 έχει υπόσταση ως ελάχιστη τιμή δείκτη. Μερικές φορές όμως η τιμή 0 δεν έχει υπόσταση. Για παράδειγμα, δεν έχει νόημα ο μηδενικός μαθητής, η μηδενική πόλη κτλ. Έτσι, παρ' όλο που υπάρχει στοιχείο με δείκτη 0 και δεσμεύεται ο αντίστοιχος χώρος μνήμης, το στοιχείο δεν χρησιμοποιείται μέσα στον κώδικα. Για να αποφύγουμε αυτήν, την περιττή για πολλούς, δέσμευση, μπορούμε να χρησιμοποιήσουμε πριν από τις δηλωτικές εντολές, που ορίζουν πίνακες, τη δηλωτική εντολή:

`Option Base` *ελάχιστος_δείκτης*

όπου *ελάχιστος_δείκτης* αριθμός 0 ή 1.

**Παράδειγμα 18-10.**

Με τις εντολές:

```
Option Base 1
Dim Students(30)      As String
Dim Lessons(12)       As String
Dim Grades(30, 12)    As Integer
```

Δηλώνεται ότι ο πίνακας `Students` θα έχει 30 ακριβώς στοιχεία, ο πίνακας `Lessons` θα έχει 12 ακριβώς στοιχεία και ο πίνακας `Grades` θα έχει 30x12=360 στοιχεία. Αν δεν είχε προηγηθεί η δηλωτική εντολή `Option Base`, ο πίνακας `Students` θα δηλωνόταν για 31 στοιχεία, ο πίνακας `Lessons` για 13 στοιχεία και ο πίνακας `Grades` για 31x13=403 στοιχεία.

Η εντολή `Option Base` θεωρείται σήμερα μάλλον ξεπερασμένη, αφού μπορούμε να χρησιμοποιήσουμε εναλλακτικά έναν άλλο τρόπο δήλωσης, ο οποίος ορίζει τόσο το κάτω όριο ενός δείκτη, όσο και το πάνω όριο του. Η δήλωση γίνεται μέσα στην αντίστοιχη δηλωτική εντολή, γράφοντας μέσα στην παρένθεση την ελάχιστη τιμή του δείκτη, την κωδική λέξη `To` και τη μέγιστη τιμή του δείκτη.

Ο καθορισμός χώρου στους πολυδιάστατους πίνακες χρειάζεται προσοχή. Το μέγεθος του πίνακα προκύπτει από το γινόμενο του εύρους τιμών κάθε δείκτη. Π.χ. σε τρισδιάστατο πίνακα με κάθε δείκτη 100, χρειάζονται 100x100x100=1.000.000 στοιχεία. Πολύ εύκολα μπορούμε να βγούμε εκτός ορίων μνήμης χωρίς να το καταλάβουμε ...



**Παραδείγματα 18-11.**

α) Η δηλωτική εντολή:

```
Dim Lessons(1 Ôï 12)  As String
```

ορίζει ελάχιστη τιμή δείκτη το 1 και μέγιστη το 12. Επίσης η δηλωτική εντολή:

```
Dim YearPeriod(1990 Ôï 2100)  As Integer
```

ορίζει ελάχιστη τιμή δείκτη το 1990 και μέγιστη το 2100.

β) Η δηλωτική εντολή:

```
Dim IC_Description(1000 Ôï 9999)  As String
```

ορίζει ελάχιστη τιμή δείκτη το 1000 και μέγιστη το 9999. Στην συγκεκριμένη περίπτωση ο δείκτης μπορεί να ταυτιστεί με τον τετραψήφιο κωδικό ενός ολοκληρωμένου κυκλώματος (IC) για να απλοποιηθεί ο τρόπος αναφοράς. Π.χ.:

```
IC_Description(7400) = "4 ðýëåò NAND, 2 åéóüäùí"
IC_Description(7402) = "4 ðýëåò NOR, 2 åéóüäùí "
IC_Description(7404) = "6 ðýëåò NOR"
        :                    :
IC_Description(7474) = "Delay Flip-Flop"
```

γ) Τα όρια μέσα στα οποία κυμαίνεται ένας δείκτης μπορεί να είναι και αρνητικά, π.χ. στη δηλωτική εντολή:

```
Dim Temperature(-45 Ôï 120) As Long
```

δ) Οι δηλώσεις και των δύο ορίων είναι δυνατόν να γίνουν και σε δισδιάστατους και σε πολυδιάστατους πίνακες.

```
Dim Chess(1 Ôï 8, 1 To 8)       As Long
Dim Cube(1 To 20, 1 To 10, 10)  As Long
```

## Στατικοί και δυναμικοί πίνακες

Ο τρόπος δήλωσης του μεγέθους των πινάκων, που περιγράψαμε στις προηγούμενες παραγράφους, θέτει τον περιορισμό ότι πρέπει να γνωρίζουμε εκ των προτέρων τη μέγιστη και την ελάχιστη τιμή που μπορεί να πάρει ένας δείκτης κατά την εκτέλεση της εφαρμογής.

Αυτός ο τρόπος δήλωσης δημιουργεί ***στατικούς πίνακες***. Στην VB υπάρχει η δυνατότητα ορισμού και ***δυναμικών πινάκων***, δηλαδή πινάκων των οποίων τα όρια, και κατά συνέπεια και το μέγεθος, μπορεί να αλλάξει κατά τη διάρκεια της εκτέλεσης του προγράμματος και δεν είναι στατικά και προκαθορισμένα τη στιγμή της κωδικοποίησης από τον προγραμματιστή. Αυτό είναι αρκετά χρήσιμο, αφού πολύ σπάνια γνωρίζουμε τους όγκους των δεδομένων που θα χρησιμοποιήσει κάθε χρήστης. Για παράδειγμα, στην άσκηση 18-2 χρησιμοποιήσαμε έναν πίνακα με 800x600 = 480.000 στοιχεία ή 1.920.000 για να καλύψουμε την περίπτωση που το μέγεθος του πλαισίου εικόνας θα ήταν πολύ μεγάλο. Αν όμως χρησιμοποιούσαμε έναν δυναμικό πίνακα, το μέγεθός της μνήμης θα δεσμεύονταν κατά τη στιγμή της εκτέλεσης του προγράμματος και θα μπορούσε να καθοριστεί ανάλογα με τις τρέχουσες ανάγκες.

Η δήλωση ενός δυναμικού πίνακα γίνεται με δηλωτική εντολή, όπως και των στατικών πινάκων, αλλά χωρίς την αναγραφή της μέγιστης τιμής των δεικτών μέσα στις παρενθέσεις. Για παράδειγμα:

```
Dim Students()    As String
Dim Grades()      As Integer
```

Ο καθορισμός όμως των ορίων των δεικτών γίνεται με την εντολή `ReDim`.

```
ReDim Students(20)
ReDim Grades(1 To 30, 1 To 12)
Redim MonthDays(1 To 28)
```

Η εντολή `Redim` δεν μπορεί να γραφεί στην περιοχή δηλώσεων μιας προγραμματιστικής μονάδας, αλλά μόνον μέσα στον κώδικα μιας υπορουτίνας.

**Παράδειγμα 18-12.**

Μπορούμε να διαμορφώσουμε τον κώδικα της άσκησης 18-1, ώστε να μη γίνεται διακοπή της εκτέλεσης του προγράμματος και να είναι δυνατή η εκτέλεσή του για όποιο πλήθος τιμών θερμοκρασιών θελήσει να εισάγει ο χρήστης.

```
Dim Temp()  As Single  'Ä Þ ë ù ó ç  ä õ í á ì é ê ï ý  ð ß í á ê á  ô ý ð ï õ  Single
Dim No      As Integer 'Ð ë Þ è ì ò  è å ñ ì ï ê ñ á ó é þ í  á ð ü  ô ï í  ÷ ñ Þ ó ô ç
                       'Å é ó á ã ù ã Þ  ð ë Þ è ï õ ò  è å ñ ì ï ê ñ á ó é þ í
No = InputBox("Ð ë Þ è ï ò  ð ü ë å ù í", "Å é ó á ã ù ã Þ")
ReDim Temp(No)         'É á è ï ñ é ó ì ü ò  ì å ã Ý è ï õ ò  ô ï õ  ð ß í á ê á
```

Η εντολή `Redim` μπορεί να δεχτεί μέσα στις παρανθέσεις μεταβλητές, σε αντίθεση με την εντολή `Dim` που μπορεί να δεχτεί μόνον σταθερές.

Η τρέχουσα μέγιστη τιμή του δείκτη ενός πίνακα μπορεί να ανιχνευτεί καλώντας τη συνάρτηση `Ubound`, που δέχεται ως παραμέτρους το όνομα του πίνακα και τον αύξοντα αριθμό του δείκτη, π.χ. `Ubound(Temp,1)` για τον πρώτο δείκτη του πίνακα `Temp` και `Ubound(Grades, 2)` για τον δεύτερο δείκτη του πίνακα `Grades`.

Επίσης, η ελάχιστη τιμή του δείκτη ενός πίνακα μπορεί να ανιχνευτεί καλώντας τη συνάρτηση `Lbound`, η οποία συντάσσεται όπως ακριβώς και η `Ubound`.

Κάθε φορά που εκτελείται η εντολή `Redim` οι τιμές των στοιχείων του πίνακα χάνονται. Η VB αρχικοποιεί τον πίνακα και θέτει για τιμή στο κάθε στοιχείο, την τιμή 0 για τους αριθμητικούς τύπους και την τιμή "" για τους αλφαριθμητικούς. Υπάρχουν όμως περιπτώσεις στις οποίες δε θέλουμε να χαθούν οι υπάρχουσες τιμές. Αυτό μπορεί να γίνει χρησιμοποιώντας την κωδική λέξη `Preserve` στη δηλωτική εντολή `ReDim`.

**Παράδειγμα 18-13.**

Στο πιο κάτω τμήμα προγράμματος, όσο ο χρήστης πληκτρολογεί αριθμούς ένας δυναμικός πίνακας `Volts` συνεχώς αυξάνει το μέγεθός του και οι αριθμοί καταχωρίζονται στα στοιχεία του.

```
Dim Volts()          'Ï ñ é ó å  ä õ í á ì é ê ü  ð ß í á ê á
Dim CurrentValue
ReDim Volts(1)

Do
   'Å é ó á ã ù ã Þ  ô é ì Þ ò
    CurrentValue = InputBox("Ô ñ Ý ÷ ï õ ó á  ô é ì Þ", "Ô Ü ó ç")
    If Not IsNumeric(CurrentValue) Then Exit Do
   'Å ð Ý ê ô á ó ç  ô ï õ  ð ß í á ê á  ê á ô Ü 1
    ReDim Preserve Volts(UBound(Volts, 1) + 1)
   'Å é ó á ã ù ã Þ  ô é ì Þ ò  ó ô ï  ô å ë å õ ô á ß ï  ó ô ï é ÷ å ß ï  ô ï õ  ð ß í á ê á
    Volts(UBound(Volts, 1)) = CurrentValue
Loop
```

Μόνον το πάνω όριο του δείκτη είναι δυνατόν να μεταβληθεί με μια εντολή `Redim`. Το κάτω όριο καθώς και το πλήθος των δεικτών μπορεί να καθοριστεί μόνον κατά την εκτέλεση της `Redim` την πρώτη φορά. Επίσης, με την εντολή `Redim` δεν είναι δυνατόν να γίνει η αλλαγή του τύπου δεδομένων του πίνακα.

## Ανακεφαλαίωση

Για ένα σύνολο μεταβλητών μπορούμε να χρησιμοποιήσουμε ένα γενικό όνομα και να αναφερόμαστε σε μια συγκεκριμένη από αυτές, γράφοντας το γενικό όνομα και έναν αριθμητικό δείκτη. Οι μεταβλητές αυτού του τύπου ονομάζονται μεταβλητές με δείκτη και συνθέτουν πίνακες. Η αναφορά σε ένα στοιχείο του πίνακα γίνεται με τη βοήθεια αριθμητικού δείκτη. Συσχετίζοντας το μετρητή ενός βρόχου με τον δείκτη ενός πίνακα μπορούμε να επεξεργαστούμε όλα τα στοιχεία του πίνακα μέσα σε ένα βρόχο.

Ανάλογα με το πλήθος των δεικτών ένας πίνακας μπορεί να είναι μονοδιάστατος, δισδιάστατο, ..., πολυδιάστατος. Η δήλωση ενός πίνακα γίνεται με εντολές `Dim`, `Private`, `Public`, `Static`. Εκτός από τους στατικούς πίνακες υπάρχουν και οι δυναμικοί. Στους δυναμικούς πίνακες το πλήθος των στοιχείων τους καθορίζεται με την εντολή `Redim`.



## Εργαστηριακές ασκήσεις

1. Συνθέστε πρόγραμμα στο οποίο να γίνεται εισαγωγή και εκτύπωση των ονομάτων μαθητών. Χρησιμοποιήστε για υπόδειγμα τον κώδικα που περιγράφεται στο παράδειγμα 18-4.
2. Να γραφεί πρόγραμμα που να δέχεται τα ονόματα των ημερών της εβδομάδας και να τα παρουσιάζει σε μια φόρμα
3. Γράψτε πρόγραμμα για την εισαγωγή του ονόματος, του ονόματος πατέρα και του επωνύμου ενός συνόλου προσώπων. Το πρόγραμμα να εκτυπώνει τα ονόματα των προσώπων όπως στο παράδειγμα 18-6.
4. Εκτελέστε την άσκηση 18-1 εισάγοντας πραγματικές τιμές θερμοκρασιών, που θα πάρετε από το νυχτερινό δελτίο ειδήσεων της τηλεόρασης.
5. Γράψτε πρόγραμμα για το παράδειγμα 18-9.
6. Εκτελέστε την άσκηση 18-2 που πραγματοποιεί την αναστροφή μιας εικόνας. Στη συνέχεια αλλάξετε την εντολή:

   `Canvas.PSet (x, y), Pixel(MaxX - x, MaxY - y)`

   του δεύτερου βρόχου σε:

   `Canvas.PSet (x, y), Pixel(MaxX - x, y)`

   Τι παρατηρείτε; Σχολιάστε το αποτέλεσμα.
   Ποιά άλλη αλλαγή μπορείτε να κάνετε, ώστε να τοποθετηθεί διαφορετικά η εικόνα μέσα στο πλαίσιο εικόνας; Τι θα συνέβαινε αν δεν χρησιμοποιούσαμε τον πίνακα για την ενδιάμεση αποθήκευση των εικονοστοιχείων αλλά εφαρμόζαμε την αναστροφή απευθείας πάνω στα εικονοστοιχεία;
7. Προσαρμόστε την άσκηση 18-2, ώστε να χρησιμοποιεί δυναμικό πίνακα.
8. Γράψτε πρόγραμμα στο οποίο να γίνονται με τη σειρά οι εξής ενέργειες:
   α) Να αποθηκεύονται τα ονόματα μαθητών σε έναν μονοδιάστατο πίνακα με όνομα `Names`.
   β) Να αποθηκεύονται τα ονόματα μαθημάτων σε έναν μονοδιάστατο πίνακα με όνομα `Lessons`.
   γ) Με συνάρτηση `InputBox` να εισάγονται οι βαθμοί κάθε μαθητή σε κάθε μάθημα. Στην περιοχή μηνύματος του διαλογικού παραθύρου να εμφανίζεται το όνομα του μαθητή και του μαθήματος και όχι οι αύξοντες αριθμοί.

## Σημειώσεις:

## Μάθημα 19

# Βασικές διαδικασίες πάνω σε πίνακες

Τελειώνοντας αυτό το μάθημα οι μαθητές θα μπορούν:

- Να περιγράφουν αλγορίθμους εύρεσης ελάχιστης και μέγιστης τιμής.
- Να αναλύουν αλγορίθμους ταξινόμησης στοιχείων πινάκων.
- Να κωδικοποιούν αλγορίθμους αναζήτησης τιμών μέσα σε πίνακες.
- Να αναλύουν αλγορίθμους συγχώνευσης στοιχείων πινάκων.

Υπάρχουν μερικές διαδικασίες που μπορούμε να τις χαρακτηρίσουμε βασικές, μιας και χρησιμοποιούνται πάρα πολύ συχνά στις διάφορες εφαρμογές που επεξεργάζονται πίνακες. Κάποιες από αυτές τις διαδικασίες, τις πιο απλές, όπως την αναφορά σε ένα στοιχείο, την εισαγωγή στοιχείων, την αντιγραφή, τις μελετήσαμε ήδη στο προηγούμενο κεφάλαιο παράλληλα με τη γενική θεωρία περί πινάκων. Σε αυτό το κεφάλαιο θα μελετήσουμε πιο σύνθετες βασικές διαδικασίες, όπως τη διαδικασία εύρεσης ελάχιστου όρου, την ταξινόμηση των δεδομένων ενός πίνακα, τη συγχώνευση των περιεχομένων δύο πινάκων και την αναζήτηση μιας τιμής μέσα σε ένα πίνακα.

## Εύρεση ελάχιστου όρου

Έστω, ότι σε έναν πίνακα Á που έχει Í στοιχεία, έχει αποθηκευτεί ένα σύνολο από αριθμούς και έστω ότι ζητάμε να βρούμε το μικρότερο από αυτούς. Ο αλγόριθμος αναλύεται στα εξής βήματα:

1. Αυθαίρετα θεωρούμε ότι ο ελάχιστος όρος βρίσκεται στο πρώτο στοιχείο του πίνακα.
2. Κάνουμε συγκρίση του υποτιθέμενου ελάχιστου όρου με το επόμενο στοιχείο του πίνακα. Αν το περιεχόμενο του επόμενου στοιχείου είναι μικρότερο του υποτιθέμενου ελάχιστου, θεωρούμε ως νέο ελάχιστο το περιεχόμενο του στοιχείου.
3. Επαναλαμβάνουμε το βήμα 2, σαρώνοντας όλα τα στοιχεία του πίνακα μέχρι να φτάσουμε στο στοιχείο Á(Í).

Στο πρόγραμμα που δίνουμε, έχουμε παρεμβάλλει και εντολές που κάνουν ανάγνωση των στοιχείων του πίνακα και εκτύπωση του ελάχιστου όρου.

```
Option Explicit
Option Base 1              'ÅëÜ÷éóôïò äåßêôçò ôï 1
:
Dim A()                    'ÄÞëùóç ôïõ ðßíáêá
Dim N    As Integer        'Ôï ðëÞèïò ôùí óôïé÷åßùí
Dim i    As Integer        'Ìåôñçôþò âñü÷ïõ
Dim Min                    'Ðñïóùñéíüò åëÜ÷éóôïò
Dim k    As Integer        'ÈÝóç ðñïóùñéíïý åëÜ÷éóôïõ
'Ðñïóäéüñéóå ôï ðëÞèïò óôïé÷åßùí ôïõ ðßíáêá
N = InputBox("Ðëþèïò óôïé÷åßùí", "Åéóáãùãþ")
ReDim A(N)                 'Êáèïñéóìüò ìåãÝèïõò ôïõ ðßíáêá
'Åéóáãùãþ óôïé÷åßùí
For i = 1 To N
   A(i) = InputBox("Óôïé÷åßï " & i, "Åéóáãùãþ")
Next i
'Áýñåóç åëÜ÷éóôïõ
Min = A(1):       k = 1    'Áóôù üôé ï åëÜ÷éóôïò åßíáé ï ðñþôïò
For i = 2 To N
   If Min > A(i) Then      'Áí äåí éó÷ýåé ç ðñüèåóç
     Min = A(i)            '   íÝïò åëÜ÷éóôïò ï Á(i)
     k = i
   End If
Next i
'ÅìöÜíéóç áðïôåëÝóìáôïò
MsgBox "ÅëÜ÷éóôïò üñïò ï " & Min & " óôç èÝóç " & k
```

## Ταξινόμηση

Έστω ένα σύνολο αριθμητικών ή αλφαριθμητικών δεδομένων. Στη διαδικασία της ***ταξινόμησης (sorting)*** πίνακα μας ζητείται να διατάξουμε τα δεδομένα που υπάρχουν στα στοιχεία του σε αύξουσα ή σε φθίνουσα σειρά με βάση την τιμή τους. Για παράδειγμα, αν τα δεδομένα αντιστοιχούν στα ονόματα ενός συνόλου ατόμων, το να τα διατάξουμε σε αύξουσα σειρά, σημαίνει να τα τοποθετήσουμε σε αλφαβητική σειρά.

Ο όρος ταξινόμηση αποτελεί πιστή μετάφραση του όρου sort που θεωρείται αδόκιμος για τη λειτουργία που τον χρησιμοποιούμε. Περισσότερο δόκιμος θα ήταν ο όρος order που σημαίνει διατάσσω

Υπάρχουν πολλοί αλγόριθμοι ταξινόμησης που ποικίλουν ως προς τη δομή, το πλήθος των βημάτων, το είδος και το πλήθος των συγκρίσεων και τη διάρκεια εκτέλεσης. Στη συνέχεια θα περιγράψουμε ένα πολύ απλό αλγόριθμο ταξινόμησης που είναι γνωστός ως ταξινόμηση με ***απευθείας επιλογή (straight selection)***. Ο αλγόριθμος αυτός είναι ο πιο απλός αλγόριθμος ταξινόμησης. Ονομάζεται έτσι, διότι κάθε φορά επιλέγει τον μικρότερο αριθμό από ένα σύνολο αριθμών.

Έστω, ότι σε έναν πίνακα με όνομα A έχουν αποθηκευτεί N το πλήθος δεδομένα. Σε κάθε πέρασμα, π.χ. στο πέρασμα i, τα στοιχεία του πίνακα χωρίζονται νοερά σε δύο τμήματα :

- Το ***τμήμα υποδοχής*** με στοιχεία τα: `A(1)`, `A(2)`,..., `A(i-1)`, στο οποίο βρίσκονται τα μέχρι εκείνη τη στιγμή ταξινομημένα δεδομένα.
- Το ***πηγαίο τμήμα*** με στοιχεία τα: `A(i)`, `A(i+1)`,..., `A(N)`, στο οποίο περιέχονται τα υπόλοιπα μη ταξινομημένα δεδομένα του πίνακα.

Όταν ο αλγόριθμος ξεκινά, στο πέρασμα `i=1`, το τμήμα υποδοχής δεν έχει καθόλου στοιχεία, αφού όπως φαίνεται από τον ορισμό του τμήματος υποδοχής το πάνω του όριο είναι `i-1=1-1=0` και το `A(0)` στοιχείο δεν έχει οριστεί στον πίνακα (δείτε πίνακα 19-1). Από το πηγαίο τμήμα που περιέχει όλα τα στοιχεία του πίνακα βρίσκεται η μικρότερη τιμή, με τρόπο που περιγράψαμε στην προηγούμενη παράγραφο. Η μικρότερη τιμή τοποθετείται στο πρώτο στοιχείο του πηγαίου τμήματος, δηλαδή στο στοιχείο `A(1)`, ενώ η τιμή που βρισκόταν σε αυτό το στοιχείο τοποθεστείται στο στοιχείο που περιείχε τη μικρότερη τιμή.

Στο επόμενο πέρασμα, το πέρασμα `i=2`, το τμήμα υποδοχής περιέχει ένα στοιχείο, με τη μικρότερη τιμή και το εύρος του πηγαίου τμήματος έχει ελαττωθεί κατά 1. Από το πηγαίο τμήμα επιλέγεται πάλι ο μικρότερος όρος και τοποθετείται στη θέση του πρώτου στοιχείου του πηγαίου τμήματος, τη θέση `A(2)` αυτή τη φορά, ενώ ο όρος που βρισκόταν σε αυτήν τη θέση μεταφέρεται στη θέση, στην οποία είχε βρεθεί ο μικρότερος όρος. Είναι ευνόητο ότι το νέο τμήμα υποδοχής περιέχει δύο όρους ταξινομημένους.

Αυτή η διαδικασία επαναλαμβάνεται αυξάνοντας κάθε φορά το `i` κατά 1, μέχρις ότου να μην απομείνει πηγαίο τμήμα. Σε κάθε πέρασμα βρίσκουμε τη μικρότερη τιμή του πηγαίου τμήματος και τη μεταφέρουμε από το στοιχείο, έστω `A(j)`, στο στοιχείο `A(i)`, δηλαδή στην πρώτη θέση του πηγαίου τμήματος. Επίσης, παράλληλα για να μη χαθεί η παλιά τιμή του στοιχείου `A(i)`, γίνεται μεταφορά της στη θέση `A(j)`. Ο αλγόριθμος συμπληρώνεται σε N-1 περάσματα (σαρώσεις) των στοιχείων του πίνακα.

| | Α(1) | Α(2) | Α(3) | Α(4) | Α(5) | Α(6) | Α(7) | Α(8) |
|---|---|---|---|---|---|---|---|---|
| Αρχική κατάσταση: | 67 | 90 | 32 | 79 | 72 | 62 | 90 | 10 |
| i=1 | 67 | 90 | 32 | 79 | 72 | 62 | 90 | 10 |
| i=2 | 10 | 90 | 32 | 79 | 72 | 62 | 90 | 67 |
| i=3 | 10 | 32 | 90 | 79 | 72 | 62 | 90 | 67 |
| i=4 | 10 | 32 | 62 | 79 | 72 | 90 | 90 | 67 |
| i=5 | 10 | 32 | 62 | 67 | 72 | 90 | 90 | 79 |
| i=6 | 10 | 32 | 62 | 67 | 72 | 90 | 90 | 79 |
| i=7 | 10 | 32 | 62 | 67 | 72 | 79 | 90 | 90 |
| Τελική κατάσταση: | 10 | 32 | 62 | 67 | 72 | 79 | 90 | 90 |

8ο πέρασμα δεν χρειάζεται. Δεν υπάρχει λόγος να βρεθεί ελάχιστος για ένα στοιχείο.



Στον πίνακα 19-1 φαίνεται, στις διαδοχικές γραμμές, η πρόοδος της ταξινόμησης κατά τα διαδοχικά περάσματα. Σε κάθε γραμμή, το τμήμα που έχει τονιστεί, αποτελεί το πηγαίο τμήμα. Επίσης, φαίνεται πως το πρώτο στοιχείο του πηγαίου τμήματος αντιμετατίθεται με τον μικρότερο όρο του πηγαίου τμήματος και ότι σε κάθε πέρασμα μια τιμή μεταφέρεται και καταλαμβάνει την τελική της θέση.

Το τμήμα προγράμματος που υλοποιεί τον παραπάνω αλγόριθμο είναι το εξής:

```
Option Explicit
Option Base 1            'ÅëÜ÷éóôïò äåßêôçò ôï 1
 :
Dim A()                  'ÄÞëùóç ôïõ ðßíáêá
Dim N       As Integer   'Ôï ðëÞèïò ôùí óôïé÷åßùí
Dim i, j    As Integer   'Ìåôñçôýò âñü÷ùí
Dim Min                  'Ðñïóùñéíüò åëÜ÷éóôïò
Dim k       As Integer   'ÈÝóç ðñïóùñéíïý åëÜ÷éóôïõ
Dim MsgStr  As String    'ÁíáêïßíùóÞ äéÜôáîçò
'1. Åßóïäïò äåäïìÝíùí
'ÄéÜâáóå ôï ðëÞèïò ôùí óôïé÷åßùí ôïõ ðßíáêá
N = InputBox("ÐëÞèïò óôïé÷åßùí", "Åéóáãùãþ")
ReDim A(N)                'Êáèïñéóìüò ìåãÝèïõò ôïõ ðßíáêá
'Áéóáãùãþ óôïé÷åßùí
For j = 1 To N
    A(j) = InputBox("Óôïé÷åßï " & j, "Åéóáãùãþ")
Next j
'2. Åðåîåñãáóßá äåäïìÝíùí
'Ðñáãìáôïðïßçóç Í-1 ðåñÜóìáôá
For j = 1 To N - 1
   'Åýñåóç åëÜ÷éóôïõ óôïé ðçãáßïõ ôìÞìá
    Min = A(j):   k = j      'Áóôù üôé ï åëÜ÷éóôïò åßíáé ï ðñþôïò
    For i = j + 1 To N
        If A(i) < Min Then   'Áí äåí éó÷ýåé ç õðüèåóç áíôéêáôÜóôáóç
            Min = A(i)
            k = i
        End If
    Next i
     'ÁíôéìåôÜèåóç á' óôïé÷åßïõ ôïõ ðçãáßïõ ìå åëÜ÷éóôï
    A(k) = A(j)          'Ôï á' ôïõ ðçãáßïõ óôç èÝóç ôïõ åëÜ÷éóôïõ
    A(j) = Min           'Ôï åëÜ÷éóôï óôç èÝóç ôïõ á' ôïõ ðçãáßïõ
Next j
'3. Åìöüíéóç ôá óôïé÷åßá äéáôåôáãìÝíá
MsgStr = ""
For j = 1 To N
    MsgStr = MsgStr & A(j) & ", "
Next j
MsgBox MsgStr, , "ÄéáôåôáãìÝíá äåäïìÝíá"
```

Το τμήμα 2 (επεξεργασία δεδομένων) περιέχει έναν βρόχο, με μετρητή τη μεταβλητή j. Ο βρόχος πραγματοποιεί τα διαδοχικά περάσματα. Στο εσωτερικό αυτού του βρόχου εκτελούνται οι διαδικασίες εύρεσης του ελαχίστου στο πηγαίο τμήμα (έχει διαμορφωθεί κάπως ο κώδικας της προηγούμενης παραγράφου) και η αντιμετάθεση του ελάχιστου με το πρώτο στοιχείο του πηγαίου τμήματος.

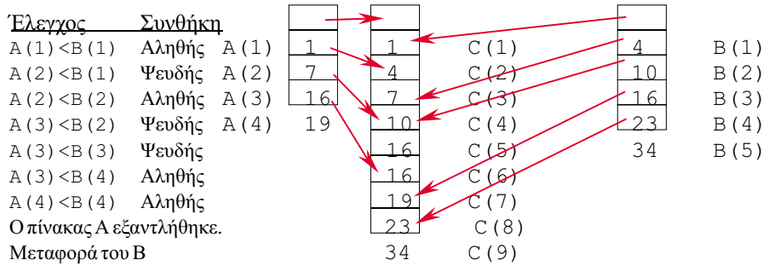
## Συγχώνευση

Στη διαδικασία της ***συγχώνευσης (merging)*** δύο πινάκων, A και B, μας ζητείται να συνενώσουμε τα δεδομένα που περιέχουν τα στοιχεία τους, σε έναν νέο πίνακα C. Ιδιαίτερο ενδιαφέρον παρουσιάζει η συγχώνευση στην περίπτωση που τα στοιχεία των πινάκων είναι ήδη διατεταγμένα κατά την ίδια φορά και θέλουμε η διάταξη να διατηρηθεί και στον νέο πίνακα.

Ας υποθέσουμε, ότι οι αρχικοί πίνακες έχουν τις τιμές τους διατεταγμένες κατά αύξουσα σειρά και ότι με την ίδια σειρά θέλουμε να τοποθετηθούν και οι τιμές στον τρίτο πίνακα. Η μέθοδος που ακολουθούμε είναι απλή. Συγκρίνουμε πάντα ένα στοιχείο του πίνακα A με ένα στοιχείο του πίνακα B. Το μικρότερο από αυτά το τοποθετούμε στον πίνακα C και από τον

πίνακα που μεταφέρθηκε το στοιχείο (τον Á ή τον Â) παίρνουμε το επόμενο στοιχείο. Για παράδειγμα, ξεκινάμε συγκρίνοντας το πρώτο στοιχείο του πίνακα A με το πρώτο στοιχείο του πίνακα Â. Αν `A(1)<B(1)`, τοποθετούμε την τιμή του `Á(1)` στη θέση `C(1)` και συνεχίζουμε για να συγκρίνουμε το στοιχείο `A(2)` με το `B(1)`. Αντίθετα, αν `A(1)>=B(1)`, τοποθετούμε την τιμή του `B(1)` στη θέση `C(1)` και συνεχίζουμε για να συγκρίνουμε το στοιχείο `A(1)` με το `B(2)`.

Επαναλαμβάνουμε μέχρι να εξαντληθούν όλα τα στοιχεία ενός εκ των δύο πινάκων. Όταν συμβεί αυτό, μεταφέρουμε στον πίνακα C τα υπόλοιπα στοιχεία του πίνακα που δεν έχουμε σαρώσει μέχρι το τέλος.




***Σχήμα 19-1.*** *Δείγμα διαδικασίας συγχώνευσης*

Στο σχήμα 19-1 δίνεται σχηματικά η διαδικασία συγχώνευσης για δύο πίνακες με 4 και 5 στοιχεία. Στην αριστερή στήλη αναφέρονται οι έλεγχοι που πραγματοποιούνται σε κάθε βήμα.
Το τμήμα προγράμματος που υλοποιεί τον παραπάνω αλγόριθμο είναι το εξής:

```
Option Base 1                'ÂëÜ÷éóôïìò äåßêôçò ôï 1
   :
Dim A(), B(), C()            'ÄÞëùóç ðéíÜêùí
Dim N, Ì     As Integer      'Ôï ðëÞèïò ôùí óôïé÷åßùí ôùí A, B
Dim i, j, k  As Integer      'Äåßêôåò ðéíÜêùí A, B, C
Dim MsgStr   As String       'Áíáêïßíùóç
'1. Åßóïäïò äåäïìÝíùí
'Óôïé÷åßá ðßíáêá A
N = InputBox("ÐëÞèïò óôïé÷åßùí ðßíáêá Á", "Åéóáãùãþ")
ReDim A(N)                   'Êáèïñéóìüò ìåãÝèïõò ôïõ ðßíáêá Á
For i = 1 To N
   A(i) = InputBox("Ðßíáêáò Á óôïé÷åßï " & i, "Åéóáãùãþ")
Next i
'Óôïé÷åßá ðßíáêá Â
Ì = InputBox("ÐëÞèïò óôïé÷åßùí ðßíáêá Â", "Åéóáãùãþ")
ReDim B(Ì)                   'Êáèïñéóìüò ìåãÝèïõò ôïõ ðßíáêá Â
For j = 1 To Ì
   B(j) = InputBox("Ðßíáêáò B óôïé÷åßï " & j, "Åéóáãùãþ")
Next j
ReDim C(N + Ì)               'Êáèïñéóìüò ìåãÝèïõò ôïõ ðßíáêá C
'2. Äéáäéêáóßá óõã÷þíåõóçò
'Áñ÷éêïðïßçóç ôùí äåéêôþí
 i = 1:    j = 1:     k = 1
'ÌÝ÷ñé íá åÎáíôëçèïýí ôá óôïé÷åßá åíüò ôùí ðéíÜêùí
 Do While (i <= N) And (j <= Ì)
    If A(i) < B(j) Then
       C(k) = A(i)           'Óôïé÷åßï áðü ôïí ðßíáêá Á
       i = i + 1             'Ôï åðüìåíï ôïõ Á
    Else
       C(k) = B(j)           'Óôïé÷åßï áðü ôïí ðßíáêá Â
       j = j + 1             'Ôï åðüìåíï ôïõ Â
    End If
    k = k + 1
 Loop
'ÌåôÜöåñå ôá õðüëïéðá óôïé÷åßá ôïõ A Þ ôïõ B óôïí C
 For j = j To Ì     'Ï âñü÷ïò äåí åêôåëåßôáé áí j>M
    C(k) = B(j)
    k = k + 1
 Next j
```



```
 For i = i To N     'Ï âñü÷ïò äåí åêôåëåßôáé áí i>N
     C(k) = A(i)
     k = k + 1
 Next i
'3. ¸ÏïäïòÞåëåóìÜôùí
 MsgStr = ""
 For k = 1 To N + Ì
     MsgStr = MsgStr & C(k) & ", "
 Next k
 MsgBox MsgStr, , "ÄéáôåôáãìÝíá äåäïìÝíá"
```

## Σειριακή αναζήτηση

Η απλούστερη μέθοδος αναζήτησης μιας τιμής μέσα σε έναν πίνακα, του οποίου τα στοιχεία είναι (ή δεν είναι) διατεταγμένα, είναι η μέθοδος της ***σειριακής αναζήτησης (sequential search)***. Η μέθοδος βασίζεται στην εξής απλή αρχή. Ξεκινώντας από το πρώτο στοιχείο του πίνακα, συγκρίνουμε το περιεχόμενό του με την τιμή που μας ενδιαφέρει. Αν οι δύο τιμές είναι ίσες, η μέθοδος σταματάει και ανακοινώνει ότι η τιμή βρέθηκε στο πρώτο στοιχείο. Αν όμως οι δύο τιμές είναι διαφορετικές, η διαδικασία σύγκρισης επαναλαμβάνεται με το δεύτερο στοιχείο, ύστερα με το τρίτο κ.ο.κ. Με αυτόν τον τρόπο είτε εντοπίζεται σε κάποιο από τα στοιχεία η τιμή που μας ενδιαφέρει, είτε διαπιστώνεται, αφού ελεγχθούν όλα τα στοιχεία του πίνακα, ότι η τιμή δεν περιέχεται στον πίνακα.

Στο παράδειγμα 2-5 έχει αναλυθεί μέ βήματα ο αλγόριθμος της σειριακής αναζήτησης.

Το τμήμα προγράμματος, που υλοποιεί τον παραπάνω αλγόριθμο, είναι το εξής:

```
Option Base 1              'ÅëÜ÷éóôïìò äåßêôçò ôï 1
 :
Dim A()                    'ÄÞëùóç ôïõ ðßíáêá
Dim N         As Integer   'Ôï ðëÞèïò ôùí óôïé÷åßùí
Dim i         As Integer   'Ìåôñçôþò âñü÷ïõ
Dim Key                    'Ôéìþ ðñïò áíáæÞôçóç
Dim MsgStr    As String    'Áíáêïßíùóç

'1. Åßóïäïò äåäïìÝíùí
'ÄéÜâáóå ôï ðëÞèïò ôùí óôïé÷åßùí ôïõ ðßíáêá
N = InputBox("ÐëÞèïò óôïé÷åßùí", "Åéóáãùãþ")
ReDim A(N)                 'Êáèïñéóìüò ìåãÝèïõò ôïõ ðßíáêá

'Åéóáãùãþ óôïé÷åßùí
For j = 1 To N
    A(j) = InputBox("Óôïé÷åßï " & j, "Åéóáãùãþ")
Next j
'Ôéìþ ðñïò áíáæÞôçóç
Key = InputBox("ÐïéÜ åßíáé ç ðñïò áíáæÞôçóç ôéìÞ ?", "ÁíáæÞôçóç")
'2. Äéáäéêáóßá áíáæÞôçóçò
For i = 1 To N
    If A(i) = Key Then Exit For
Next i
'3. Áíáêïßíùóç áðïôåëÝóìáôïò
If i <= N Then
   MsgStr = "ÂñÝèçêå óôç èÝóç " & i
Else    'Ìåôñçôþò i>N óçìáßíåé üôé äå äéáêüðçêå ï âñü÷ïò óôç ìÝóç
   MsgStr = "Äå ÂñÝèçêå"
End If
MsgBox MsgStr, , "ÁíáæÞôçóç"
```

Το πλήθος των κύκλων που γίνονται στο βρόχο του τμήματος 2 του προγράμματος κυμαίνεται από 1, αν η τιμή βρίσκεται στο πρώτο στοιχείο, μέχρι N, αν η τιμή βρίσκεται στο τελευταίο στοιχείο. Σε περίπτωση που τα στοιχεία είναι ήδη ταξινομημένα, το πρόγραμμα μπορεί να βελτιωθεί. Αν κατά τη σάρωση των στοιχείων του πίνακα βρεθεί στοιχείο με τιμή μεγαλύτερη από την αναζητούμενη πρέπει να διακόπτεται η διαδικασία, ώστε να μη γίνεται περιττή αναζήτηση στα επόμενα στοιχεία. Όπως θα δούμε στην επόμενη παράγραφο στην περίπτωση των ταξινομημένων στοιχείων εφαρμόζονται ταχύτεροι αλγόριθμοι.

Τέλος, πρέπει να σημειωθεί ότι ο αλγόριθμος αναζήτησης χρειάζεται τροποποίηση, στην περίπτωση που αναζητούνται όλες οι εμφανίσεις μιας τιμής μέσα στα στοιχεία του πίνακα.

## Δυαδική αναζήτηση

Η μέθοδος της δυαδικής αναζήτησης (binary search) εφαρμόζεται μόνο σε διατεταγμένους πίνακες. Πρόκειται για μια από τις πιο γρήγορες μεθόδους αναζήτησης. Η ιδέα στην οποία βασίζεται η μέθοδος είναι η εξής:

Στο παράδειγμα 2-5 έχει αναλυθεί με βήματα ο αλγόριθμος της δυαδικής αναζήτησης.

Σε κάθε βήμα συγκρίνουμε την τιμή που αναζητάμε, έστω Key, με την τιμή του στοιχείου A(i), που βρίσκεται στο μέσο ενός τμήματος του πίνακα. Αν Key>A(i) τότε η προς αναζήτηση τιμή πρέπει να βρίσκεται μετά από το στοιχείο που μόλις ελέγξαμε. Γι' αυτό και η αναζήτηση συνεχίζεται στο κάτω μισό του τμήματος που εξετάζεται. Αντίθετα, αν Key<A(i), τότε η αναζήτηση συνεχίζεται στο πάνω μισό του τμήματος που εξετάζουμε. Δηλαδή σε κάθε βήμα αποκλείεται το μισό του τμήματος που έχει απομείνει από το προηγούμενο βήμα. Η διαδικασία ελέγχου-απόρριψης αρχίζει με περιοχή ανίχνευσης ολόκληρο τον πίνακα και συνεχίζεται μέχρις ότου είτε να ανιχνευτεί η αναζητούμενη τιμή είτε να διαπιστωθεί ότι δεν υπάρχει η τιμή σε κανένα στοιχείο του πίνακα.

Στον πίνακα 19-2, στις διαδοχικές γραμμές φαίνεται η πρόοδος της αναζήτησης κατά τα βήματα εκτέλεσης του αλγορίθμου. Σε κάθε γραμμή, το τμήμα που έχει τονιστεί, αποτελεί την περιοχή που απομένει για αναζήτηση. Επίσης, στον πίνακα φαίνονται και οι τιμές των μεταβλητών Top, Bottom και Middle, που σημειώνουν το μικρότερο στοιχείο, το μεγαλύτερο στοιχείο και το στοιχείο του μέσου αντίστοιχα, της περιοχής αναζήτησης. Η προς αναζήτηση τιμή είναι το 90, που εντοπίζεται μετά από 4 βήματα.

| Βήμα | Top | Middle | Bottom | Á(1) | Á(2) | Á(3) | Á(4) | Á(5) | Á(6) | Á(7) | Á(8) | Á(9) | Á(10) | Á(11) | Á(12) | Á(13) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 7 | 13 | 10 | 27 | 32 | 35 | 54 | 62 | 67 | 72 | 79 | 83 | **90** | 95 | 99 |
| 2 | 8 | 10 | 13 | 10 | 27 | 32 | 35 | 54 | 62 | 67 | 72 | 79 | 83 | **90** | 95 | 99 |
| 3 | 11 | 12 | 13 | 10 | 27 | 32 | 35 | 54 | 62 | 67 | 72 | 79 | 83 | **90** | 95 | 99 |
| 4 | 11 | 11 | 11 | 10 | 27 | 32 | 35 | 54 | 62 | 67 | 72 | 79 | 83 | **90** | 95 | 99 |

*Πίνακας 19-2.*

Το τμήμα προγράμματος που υλοποιεί τον παραπάνω αλγόριθμο είναι το εξής:

```
Option Explicit
Option Base 1                'ÅëÜ÷éóôiò äåßêôçò ôï 1
 :
Dim A()                      'ÄÞëùóç ôïõ ðßíáêá
Dim N          As Integer    'Ôï ðëÞèïò ôùí óôïé÷åßùí
Dim i          As Integer    'ÌåôñçôÞò âñü÷ïõ
Dim Top        As Integer    '¢íù üñéi ðåñéi÷Þò áíáæÞôçóçò
Dim Middle     As Integer    'ÌÝóïí
Dim Bottom     As Integer
Dim Key                      'ÔéìÞ ðñïò áíáæÞôç'ÊÜôù üñéïóç
Dim MsgStr     As String     'Áíáêïßíùóç

'1. Åßóïäïò äåäïìÝíùí
'ÄéÛâáóå ôï ðëÞèïò ôùí óôïé÷åßùí ôïõ ðßíáêá
N = InputBox("ÐëÞèïò óôïé÷åßùí", "Åéóáãùãþ")
ReDim A(N)                   'Êáèïñéóìüò ìåãÝèïõò ôïõ ðßíáêá
'Åéóáãùãþ óôïé÷åßùí
For i = 1 To N
    A(i) = InputBox("Óôïé÷åßi " & i, "Åéóáãùãþ")
Next i
'ÔéìÞ ðñïò áíáæÞôçóç
Key = InputBox("ÐïéÜ åßíáé ç ðñïò áíáæÞôçóç ôéìÞ ?", "ÁíáæÞôçóç")
'2. Äéáäéêáóßá áíáæÞôçóçò
'Áñ÷éêÜ üñéá
Top = 1:     Bottom = N
```



```
'Ïóï ðÜíù ÷åé ðåñéï÷Þ áíáæÞôçóçò
Do While Top <= Bottom
    Middle = (Top + Bottom) \ 2  'Âñåò ôï ìÝóï ôïõ äéáóôÞìáôïò
     Select Case Key
         Case Is < A(Middle)
            Top = Middle + 1     'Ç ôéìÞ óôï êÜôù ìÝñïò ôïõ ðßíáêá
         Case Is > A(Middle)
             Bottom = Middle - 1 'Ç ôéìÞ óôï ðÜíù ìÝñïò ôïõ ðßíáêá
         Case Else
             Exit Do             'Ç ôéìÞ âñÝèçêå
     End Select
 Loop
'3. ¸ïäïò áðïôåëåóìÜôùí
 If Top <= Bottom Then
     MsgStr = "Ç ôéìÞ âñÝèçêå óôç èÝóç " & Middle
 Else
     MsgStr = "Ç ôéìÞ äå âñÝèçêå"
 End If
 MsgBox MsgStr, , "ÁíáæÞôçóç"
```

Παρατηρήστε το τέχνασμα διακοπής του βρόχου. Αν το κάτω όριο γίνει μικρότερο του πάνω ορίου σημαίνει ότι δεν έχει απομείνει περιοχή αναζήτησης. Επίσης, το ίδιο κριτήριο χρησιμοποιείται και για να ανιχνευτεί, αν η αναζήτηση διακόπηκε από την εύρεση της τιμής ή από εξάντληση της περιοχής αναζήτησης.

Η δυαδική αναζήτηση τερματίζεται μετά από Int(log2N)+1 ελέγχους το πολύ. Αν θεωρήσουμε ότι έχουμε έναν πίνακα με 10.000.000 στοιχεία, στα οποία είναι καταχωρισμένα ονόματα του πληθυσμού της Ελλάδας, θα χρειαστούν 24 το πολύ άλματα και αναζητήσεις στα στοιχεία του για να εντοπιστεί ένα όνομα. Αντίθετα η σειριακή αναζήτηση απαιτεί 5.000.000 ελέγχους κατά μέσο όρο.

## Εργαστηριακές ασκήσεις

1. Δημιουργήστε εφαρμογή, στην οποία να εφαρμόζεται ο αλγόριθμος της εύρεσης του μέγιστου ενός πίνακα.
2. Δημιουργήστε εφαρμογή στην οποία να χρησιμοποιείται ο αλγόριθμος της απευθείας επιλογής για την ταξινόμηση των στοιχείων ενός πίνακα.
3. Στα στοιχεία ενός πίνακα βρίσκονται τα βιβλία της σχολικής βιβλιοθήκης. Γράψτε πρόγραμμα, στο οποίο με τον αλγόριθμο της σειριακής αναζήτησης να ελέγχει αν ένα βιβλίο υπάρχει στη βιβλιοθήκη ή όχι.
4. Ένας πίνακας έχει τα στοιχεία του ταξινομημένα. Βελτιώστε το πρόγραμμα της σειριακής αναζήτησης, ώστε να διακόπτεται στην περίπτωση που κατά τη σάρωση των στοιχείων πίνακα βρεθεί στοιχείο με τιμή μεγαλύτερη από τη ζητούμενη.
5. Σε έναν πίνακα υπάρχουν επαναλαμβανόμενες τιμές. Τροποποιήστε το πρόγραμμα της σειριακής αναζήτησης, ώστε να είναι δυνατή η αναφορά όλων των θέσεων στις οποίες υπάρχει η προς αναζήτηση τιμή.
6. Γράψτε ένα πρόγραμμα που να ταξινομεί δύο πίνακες και μετά να τους συγχωνεύει.
7. Να δημιουργηθεί πρόγραμμα, το οποίο να διαγράφει μια τιμή από ένα στοιχείο ενός πίνακα.
   Υπόδειξη: Αφού εντοπιστεί το προς διαγραφή στοιχείο να γίνει μετάθεση όλων των υπολοίπων προς τα πάνω. Μετά με εντολή Redim Preserve να αλλάξει το μέγεθος του πίνακα.
8. Στα στοιχεία δύο πινάκων καταχωρίζουμε, σε αντίστοιχες θέσεις, τα ονόματα των κρατών της Ευρώπης και τις πρωτεύουσές τους. Γράψτε πρόγραμμα, το οποίο να διαβάζει το όνομα του κράτους και να μας εμφανίζει την πρωτεύουσά του.