Κων/νου Στυλιάδη

Έτοιμες Ασκήσεις σε Pascal

Φλώρινα, Οκτώβριος 2000

```
Program prostesi;  { όνομα ή επικεφαλίδα ή τίτλος του προγράμματος }
(* αυτό το πρόγραμμα προσθέτει δύο αριθμούς *)
Uses
        Crt;    { δήλωση της μονάδας Crt, που περιέχει συναρτήσεις οθόνης }
Var     { τμήμα δηλώσεων του προγράμματος }
        a, b, c : Real; { δήλωση τριών αριθμητικών μεταβλητών }
Begin { αρχή του κυρίου προγράμματος, δηλ. των εντολών }
        Read(a, b);   { διάβασμα τιμών για δύο μεταβλητές }
        c := a+b;     { υπολογισμός του αθροίσματος σε μια τρίτη μεταβλητή }
        Write(c)      { εμφάνιση της τιμής της μεταβλητής στην οθόνη }
End. { τέλος του προγράμματος }
(* ο χαρακτήρας ; χρησιμοποιείται για να δηλώσει το τέλος μιας εντολής και
έχει λίγες μόνο εξαιρέσεις και το σύμβολο := χρησιμοποιείται για να καταχω-
ρούμε τιμές σε μεταβλητές *)
```

```
Program misthos;
(* αυτό το πρόγραμμα κάνει υπολογισμό μισθοδοσίας *)
Uses
        Crt;
Var
        HM, ME, MA, KP, KA : Real;
{ HM είναι το ημερομίσθιο, ΜΕ είναι οι μέρες εργασίας, ΜΑ είναι οι μικτές
αποδοχές, ΚΡ είναι οι κρατήσεις και ΚΑ είναι οι καθαρές αποδοχές }
Begin
        Read(HM, ME);
        MA := ME*HM;
        KP := 0.20*MA;
        KA := MA-KP;
        Write(MA, KP, KA)
End.
```

```
Program fahr_cel;
(* πρόγραμμα μετατροπής θερμοκρασίας Φαρενάιτ σε Κελσίου *)
(* ο τύπος μετατροπής είναι ο εξής : CEL = ((FAHR-32)*5)/9 *)
Uses
        Crt;
Const
        factor = 32.0;          { δήλωση μιας σταθεράς }
Var
        bath_fahr, bath_cel : Real;
{ bath_fahr είναι οι βαθμοί Φαρενάιτ και bath_cel είναι οι βαθμοί Κελσίου }
Begin
        Writeln('Μετατροπή θερμοκρασίας από βαθμούς Φαρενάιτ ');
        Writeln('σε βαθμούς Κελσίου');
        Writeln;        { αφήνει μια κενή γραμμή στην οθόνη }
        Write('Δώσε τη θερμοκρασία σε βαθμούς Φαρενάιτ : )';
        Readln(bath_fahr); { διάβασμα της θερμοκρασίας σε Φαρενάιτ }
        { υπολογισμός της θερμοκρασίας σε βαθμούς Κελσίου }
        bath_cel := ((bath_fahr-factor)*5.0)/9.0;
        Writeln('Αντιστοιχεί σε ', bath_cel, 'βαθμούς Κελσίου');
        Writeln;        { αφήνει μια κενή γραμμή στην οθόνη }
End.
```

Το παρακάτω πρόγραμμα παρουσιάζει μια επεξεργασία συμβολοσειρών.

```
Program Strings;
Uses
        Crt;

Const
        stathera = 'Γράψτε κάτι και πατήστε Enter : ';

Var
        prompt_1 : String;
        prompt_2 : String;
        input_str : String;

Begin
        Clrscr;         { κλήση συνάρτησης που καθαρίζει την οθόνη }
        prompt_1 := stathera;
        prompt_2 := 'Γράψατε : ';
        Write(prompt_1);
        Readln(input_str);
        Write(prompt_2);
        Writeln(input_str);
End.
```

Το αποτέλεσμα από το τρέξιμο του προγράμματος θα είναι το εξής :
Γράψτε κάτι και πατήστε Enter : Turbo Pascal 6.0!
Γράψατε : Turbo Pascal 6.0!
Μια συμβολοσειρά μπορεί να περιέχει μέχρι 255 χαρακτήρες.

Ακολουθεί ένα απλό πρόγραμμα υπολογισμού του εμβαδού ενός κύκλου.

```
Program Circle_Area;
Uses
        Crt;
Const
        Pi = 3.1415926;
Var
        aktina, area : Real;
Begin
        Clrscr;
        Writeln('Δώσε την ακτίνα του κύκλου : ');
        Readln(aktina);
        area := (Pi * (aktina*aktina));
        Writeln('Εμβαδόν κύκλου = ',  area:8:2);
        Writeln;
End.
```

{ η έκφραση area:8:2 χρησιμοποιεί συνολικά 8 θέσεις για να εμφανίσει την τιμή της μεταβλητής area και 2 θέσεις για τα δεκαδικά }

```
Program Concatenation;
Uses
        Crt;
Var
        Str1, Str2, Str3 : String[20];
        { ορίζει τρεις συμβολοσειρές μεγέθους 20 χαρακτήρων }
Begin
        Clrscr;
        Str1 := 'Αντωνιάδης';
        Str2 := 'Αντώνιος';
        Str3 := Str1+Str2;          { ένωση δύο συμβολοσειρών }
        Writeln('Str1 + Str2 = ', Str3);
End.
```

```
Program Ypoloipo;
{ Το πρόγραμμα αυτό υπολογίζει και τυπώνει το υπόλοιπο της διαίρεσης ενός
ακεραίου αριθμού με το τρία }
Uses
        Crt;
Var
        x, y : Longint;
Begin
        Clrscr;
        Writeln('Δώσε έναν ακέραιο αριθμό : ');
        Readln(x);
        y := x mod 3;          { υπολογισμός του ακεραίου υπολοίπου x/3 }
        { η παρακάτω εντολή διακλαδίζει τη ροή του προγράμματος σε τρεις
περιπτώσεις, ανάλογα με την τιμή της μεταβλητής y }
        Case y of
                0 : Writeln('μηδέν');          { y = 0 }
                1 : Writeln('ένα');            { y = 1 }
                2 : Writeln('δύο')             { y = 2 }
        end; { case }
        { η παρακάτω εντολή περιμένει μέχρι να πατήσουμε ένα πλήκτρο και
είναι χρήσιμο να τοποθετείται στο τέλος του προγράμματος, ώστε να μην φεύ-
γουμε απότομα από το πρόγραμμα, αλλά να περιμένει μέχρι να δούμε τα α-
ποτελέσματα και όταν είμαστε έτοιμοι, πατάμε ένα πλήκτρο και το πρόγραμ-
μα τελειώνει }
        Repeat until keypressed;
End.
```

Program Athroisma;

{ Το πρόγραμμα αυτό υπολογίζει και εκτυπώνει το άθροισμα κάποιων αριθμών, τελειώνει δε όταν το άθροισμά τους ξεπεράσει το 1.000 }

```
Uses
        Crt;
Var
        a, s : Real;
Begin
        Clrscr;
        s := 0;          { απόδοση αρχικής τιμής στη μεταβλητή }
        Repeat           { οι παρακάτω εντολές επαναλαμβάνονται μέχρις ότου }
                Readln(a);    { η μεταβλητή s γίνει μεγαλύτερη από 1000 }
                s := s+a
        Until s>1000;
        Writeln(s);
        Repeat until keypressed;
End.
```

```
Program Foros;
{ Το πρόγραμμα αυτό υπολογίζει τον φόρο ενός μισθωτού για ένα χρόνο, δί-
νοντας σαν δεδομένα το ετήσιο εισόδημά του και τον αριθμό των παιδιών του.
Ο φόρος υπολογίζεται σαν το ένα τρίτο του εισοδήματος αφού αφαιρέσουμε
τις απαλλαγές που είναι η ατομική έκπτωση καθώς και η έκπτωση για το κάθε
παιδί }
Uses
        Crt;

Const
        ekptosh = 500,000;          { γενική έκπτωση }
        ekptosh_boys = 100,000;     { έκπτωση για το κάθε παιδί }

Var
        boys, eisodima, forologiteo, foros : Longint;
        { χρησιμοποιούμε τύπο δεδομένων Longint γιατί οι τιμές που θα επε-
ξεργαστούμε είναι μεγαλύτερες από την τιμή 32767 που υποστηρίζει ο τύπος
δεδομένων Integer }

Begin
        Clrscr;
        Writeln('Δώσε το ετήσιο εισόδημα : ');
        Readln(eisodima);
        Writeln('Δώσε τον αριθμό των παιδιών : ');
        Readln(boys);

        { υπολογισμός του φορολογητέου }
        forologiteo := eisodima - ekptosh - boys*ekptosh_boys;
        foros := forologiteo div 3;  { υπολογισμός του 1/3 του φορολογητέου }

        { στην παρακάτω εντολή Writeln συνδυάζουμε εκτύπωση σταθερών τι-
μών και μεταβλητών, όπου οι σταθερές τιμές, δηλ. τα μηνύματα, είναι πάντα
μέσα σε εισαγωγικά και χωρίζονται από τις μεταβλητές με κόμματα }
        Writeln('Για εισόδημα : ', eisodima, ' το φορολογητέο είναι : ',
                forologiteo, ' και ο φόρος είναι : ', foros);
        Repeat until keypressed;
End.
```

```
Program CharacterCount;
{ Το πρόγραμμα αυτό μετράει τα γράμματα και τα ψηφία που υπάρχουν σε
κάποια δεδομένα που καταχωρούνται από το πληκτρολόγιο και τελειώνει ό-
ταν συναντήσει τον χαρακτήρα # }
Uses
        Crt;

Var
        LetterCount, DigitCount : Integer;
        Character : Char;

Begin
        Clrscr;
        { όταν χρησιμοποιούμε μεταβλητές σαν μετρητές, πάντα τις μηδενίζου-
με στην αρχή του προγράμματος }
        LetterCount := 0;   { μετρητής γραμμάτων }
        DigitCount := 0;    { μετρητής ψηφίων }

        { οι παρακάτω εντολές επαναλαμβάνονται όσο ο χαρακτήρας που δί-
νουμε από το πληκτρολόγιο είναι διάφορος του # και όταν δώσουμε τον χα-
ρακτήρα #, βγαίνουμε από τον βρόχο While }
        While Character <> '#' do
        Begin
                Readln(Character); { διάβασμα ενός χαρακτήρα }
                { αν ο χαρακτήρας είναι μεταξύ των λατινικών A και Z }
                If ('A' <= Character) and (Character <= 'Z') then
                        LetterCount := LetterCount + 1
                        { αύξηση του LetterCount }
                { αλλιώς, αν ο χαρακτήρας είναι ένα από τα ψηφία 0 έως 9 }
                Else If ('0' <= Character) and (Character <= '9') then
                        DigitCount := DigitCount + 1
                        { αύξηση του DigitCount }
        End;  { end of while }

        Writeln('Βρέθηκαν ', LetterCount, ' γράμματα και ', DigitCount,
                ' ψηφία');
        Writeln;
        Repeat until keypressed;
End.
```

Program DEH;

{ *Το πρόγραμμα αυτό υπολογίζει τον λογαριασμό της ΔΕΗ ενός καταναλωτή με δεδομένο ότι για τις πρώτες 100 μονάδες η 1 KWh χρεώνεται 15 δρχ., για τις επόμενες 200 μονάδες χρεώνεται 16 δρχ. και από 201 και πάνω μονάδες χρεώνεται 17 δρχ. Ο ΦΠΑ είναι 18%, το ποσό για την ΕΡΤ είναι 5% και το πάγιο είναι 1.500 δρχ.* }

```
Uses
        Crt;
Var
        katanalosi, kostos : Real;
Begin
        Clrscr;
        Writeln('Δώσε την κατανάλωση : ');
        Readln(katanalosi);
        If katanalosi < 100 then kostos := katanalosi * 15;
        If (katanalosi > 100) and (katanalosi < 200)  then
                kostos := 100 * 15 + (katanalosi - 100) * 16;
        If katanalosi > 200 then
                kostos := 100 * 15 + 100 * 16 + (katanalosi - 200) * 17;
        kostos := kostos * 1.18;              { προστίθεται ο ΦΠΑ }
        kostos := kostos * 1.05 + 1500;    { προστίθεται η ΕΡΤ και το πάγιο }
        Writeln('Για τη ΔΕΗ θα πληρώσετε : ', kostos:9:0, ' δρχ.');
        Repeat until keypressed;
End.
```

Program MAX;

{ *Το πρόγραμμα αυτό βρίσκει τον μεγαλύτερο από τρεις αριθμούς a, b, c. Χρειάζεται να ορίσουμε μια βοηθητική μεταβλητή max, όπου εκεί θα καταχωρούμε προσωρινά τον μεγαλύτερο αριθμό* }

```
Uses
        Crt;
Var
        a, b, c, max : Real;
Begin
        Clrscr;
        Writeln('Δώσε τους τρεις αριθμούς : ');
        Readln(a, b, c);
        max := a;
        If b > max then max := b;
        If c > max then max := c;
        Writeln('Ο μεγαλύτερος αριθμός είναι ο : ', max);
        Repeat until keypressed;
End.
```

Program Monos_Zygos;
{*Το πρόγραμμα αυτό βρίσκει αν ένας ακέραιος αριθμός είναι μονός ή ζυγός*}
```
Uses
        Crt;
Var
        k, p, y : Longint;
Begin
        Clrscr;
        Writeln('Δώσε έναν ακέραιο αριθμό : ');
        Readln(k);
        p := k div 2;           { εύρεση του πηλίκου }
        y := k mod 2;           { εύρεση του υπολοίπου }
        If y = 0 then
                Writeln('Ο αριθμός είναι ζυγός')
        Else
                Writeln('Ο αριθμός είναι μονός');
        Repeat until keypressed;
End.
```

Program Misthodosia;
{ *Το πρόγραμμα αυτό υπολογίζει τον μισθό ενός εργάτη όταν δίνονται οι μέρες εργασίας του, οι υπερωρίες του σε ώρες και είναι γνωστό ότι οι κρατήσεις είναι το 20% των αποδοχών του, οι υπερωρίες του πληρώνονται με το 20% του ημερομισθίου και ο φόρος είναι 5%* }
```
Uses
        Crt;
Var
        hmerom, meres_erg, yperories, mikta, krathseis, foros, kathara : Real;
Begin
        Clrscr;
        Writlen('Δώσε το ημερομίσθιο : ');
        Readln(hmerom);
        Writeln('Δώσε τις μέρες εργασίας : ');
        Readln (meres_erg);
        Writeln('Δώσε τις υπερωρίες : ');
        Readln(yperories);

        { υπολογισμός μικτών αποδοχών }
        mikta := hmerom * meres_erg + yperories * hmerom * 0.2;
        krathseis := 0.2 * mikta;               { υπολογισμός κρατήσεων }
        foros := 0.05 * (mikta - krathseis);    { υπολογισμός φόρου }
        kathara := mikta - krathseis - foros;   { υπολογισμός καθαρών }
        Writeln('Οι μικτές αποδοχές είναι : ', mikta,
                ' οι καθαρές αποδοχές είναι : ', kathara,
                ' και οι κρατήσεις είναι : ',krathseis);
        Repeat until keypressed;
End.
```

Program Antistrofh;

{ *Το πρόγραμμα αυτό διαβάζει έναν διψήφιο ακέραιο αριθμό και υπολογίζει και εκτυπώνει τον ακέραιο που προκύπτει από την αντιστροφή των ψηφίων του* }

```
Uses
        Crt;
Var
        x, p, y, x1 : Byte;
Begin
        Clrscr;
        Writeln('Δώσε έναν διψήφιο ακέραιο αριθμό : ');
        Readln(x);
        p := x div 10;          { εύρεση του πηλίκου }
        y := x mod 10;          { εύρεση του υπολοίπου }
        x1 := y * 10 + p;       { σχηματισμός του αντίστροφου αριθμού }
        Writeln('Ο νέος αριθμός είναι :  ', x1);
        Repeat until keypressed;
End.
```

Program Talhro;

{ *Το πρόγραμμα αυτό στρογγυλοποιεί έναν μισθό στο τάληρο, δηλ. αν λήγει σε 0, 1 και 2 τον κάνει να λήγει σε 0, αν λήγει σε 3, 4, 5, 6 ή 7 τον κάνει να λήγει σε 5 και αν λήγει σε 8 ή 9 τον κάνει να λήγει σε 0 αλλά αυξημένο κατά 10* }

```
Uses
        Crt;
Var
        m, p, y : Longint;
Begin
        Clrscr;
        Writeln('Δώσε τον μισθό : ');
        Readln(m);
        p := m div 10;          { εύρεση του πηλίκου }
        y := m mod 10;          { εύρεση του υπολοίπου }
        { το y είναι στην ουσία το τελευταίο ψηφίο του μισθού και το p είναι όλα
        τα υπόλοιπα ψηφία του μισθού χωρίς το y, δηλ. αν ο μισθός είναι 156967, το y
        θα είναι ίσο με 7 και το p θα είναι ίσο με 15696 }
        If y <= 2 then m := p * 10;
                { 15696 * 10 = 156960 }
        If (y > 2) and (y < 8) then m := p * 10 + 5;
                { 15696 * 10 + 5 = 156965 }
        If y >= 8 then m := p * 10 + 10;
                { 15696 * 10 + 10 = 156970 }
        Writeln('Ο στρογγυλοποιημένος μισθός είναι : ', m);
        Repeat until keypressed;
End.
```

```
Program Praxis;
{ Το πρόγραμμα αυτό διαβάζει δύο αριθμούς και ένα από τα τέσσερα σύμβο-
λα της αριθμητικής (+ - * /) και κάνει την αντίστοιχη πράξη στους αριθμούς }
Uses
        Crt;
Var
        a, b, r : Real;
        z : Char;
Begin
        Clrscr;
        Writeln('Δώσε τους δύο αριθμούς : ');
        Readln(a, b);
        Writeln('Δώσε το σύμβολο της πράξης : ');
        Readln(z);
        Case z of
                '+'     : r := a + b;
                '-'     : r := a - b;
                '*'     : r := a * b;
                '/'     : r := a / b;
        End; { case }
        Writeln('Το αποτέλεσμα είναι : ', r);
        Repeat until keypressed;
End.
```

Το παρακάτω πρόγραμμα περιέχει ένα παράδειγμα χρήσης της εντολής While :

```
Program Twhile;
Var
        metritis : Integer;
Begin
        Clrscr;
        metritis := 0;          { μηδενισμός του μετρητή }
        While metritis < 6 do
        Begin
                Writeln('Μετρητής = ', metritis);
                metritis := metritis + 2;     { αύξηση του μετρητή κατά 2 }
        End; { end of while }
End.
```

Ο βρόχος While επαναλαμβάνεται όσο ή μεταβλητή metritis είναι μικρότερη του 6, δηλ. η επανάληψη θα συμβεί για τις τιμές της μεταβλητής 0, 2 και 4. Μόλις η μεταβλητή γίνει ίση με 6, θα βγούμε από τον βρόχο.

```
Program Trepeat;
Var
        metritis : Integer;
Begin
        Clrscr;
        metritis := 0;
        Repeat
                Writeln('Μετρητής = ', metritis);
                metritis := metritis + 2;
        Until (metritis > 4);
End.
```

Η εντολή Repeat εκτελεί το κύριο σώμα του βρόχου τουλάχιστον μία φορά, ακόμα και αν δεν ισχύει η συνθήκη επανάληψης του βρόχου, καθώς η συνθήκη αυτή ελέγχεται στο τέλος του βρόχου και αφού έχουν εκτελεστεί οι εντολές του σώματος του βρόχου.

Το παρακάτω πρόγραμμα περιέχει έναν απλό βρόχο FOR :

```
Program Tfor;
Var
        metritis : Integer;
Begin
        Clrscr;
        For metritis := 0 to 4 do
                Writeln('Μετρητής = ', metritis);
End.
```

Το πρόγραμμα μετράει από το 0 μέχρι το 4 και αυξάνεται κάθε φορά κατά ένα, με αποτέλεσμα η μεταβλητή metritis να παίρνει διαδοχικά τις τιμές 0, 1, 2, 3 και 4. Αν αντικατασταθεί το To με το Downto, τότε ο βρόχος θα μετράει από το 4 μέχρι το 0, ελαττώνοντας τον μετρητή κατά ένα κάθε φορά.

```
Program Ektyposi_Akeraion;
{ Το πρόγραμμα αυτό εκτυπώνει τους ακέραιους από το 1 μέχρι το 20 }
Uses
        Crt;
Var
        i : Byte;
Begin
        Clrscr;
        For i := 1 to 20 do Writeln(i);
        { ο παραπάνω βρόχος θα εκτελεστεί 20 φορές και θα τυπώσει τους
        αριθμούς από το 1 μέχρι και το 20 }
        Repeat until keypressed;
End.
```

```
Program Sum_Product;
{ Το πρόγραμμα αυτό υπολογίζει το άθροισμα και το γινόμενο Ν αριθμών }
Uses
        Crt;
Var
        i, N : Byte;
        x, sum, product : Real;
Begin
        Clrscr;
        sum := 0;               { μηδενίζουμε τον αθροιστή }
        product := 1;           { δίνουμε την τιμή 1 στο γινόμενο }
        Writeln('Δώσε το πλήθος των αριθμών : ');
        Readln(N);
        For i := 1 to N do
        Begin
                Readln(x);              { διαβάζουμε έναν αριθμό }
                sum := sum + x;     { τον προσθέτουμε στον αθροιστή }
                product := product * x;
                { τον πολλαπλασιάζουμε με τη μεταβλητή product }
        End; { end of for }
        Writeln('Το άθροισμα των αριθμών είναι : ', sum, ' και το γινόμενό
                τους είναι : ', product);
        Repeat until keypressed;
End.
```

```
Program Max;
{ Το πρόγραμμα αυτό βρίσκει τον μεγαλύτερο από Ν αριθμούς }
Uses
        Crt;
Var
        i, N : Byte;
        x, max : Real;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των αριθμών : ');
        Readln(N);
        Readln(x);    { διαβάζουμε ανεξάρτητα τον 1ο αριθμό και }
        max := x;     { τον θεωρούμε προσωρινά σαν μέγιστο }
        For i := 1 to N-1 do          { εκτελούμε τον βρόχο Ν-1 φορές }
        Begin
                Readln(x);    { διαβάζουμε τον επόμενο αριθμό }
                { τον συγκρίνουμε με τον max και αν είναι μεγαλύτερος,
                max γίνεται ο αριθμός αυτός }
                If x > max then max := x;
        End; { end of for }
        Writeln('Ο μεγαλύτερος αριθμός είναι ο : ', max);
        Repeat until keypressed;
End.
```

```
Program Pos_Zero_Neg;
{ Το πρόγραμμα αυτό βρίσκει πόσοι από Ν αριθμούς είναι θετικοί, πόσοι αρ-
νητικοί και πόσοι μηδέν }
Uses
        Crt;
Var
        i, N, pos, zer, neg : Byte;
        x : Real;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των αριθμών : ');
        Readln(N);
        pos := 0;       { μηδενίζουμε τον μετρητή των θετικών αριθμών }
        zer := 0;       { μηδενίζουμε τον μετρητή των μηδενικών αριθμών }
        neg := 0;       { μηδενίζουμε τον μετρητή των αρνητικών αριθμών }
        For i := 1 to N do
        Begin
                Readln(x);    { διαβάζουμε έναν αριθμό }
                { αν είναι θετικός, αυξάνουμε τη μεταβλητή pos κατά ένα }
                If x > 0 then pos := pos + 1;
                { αν είναι μηδενικός, αυξάνουμε τη μεταβλητή zer κατά ένα }
                If x = 0 then zer := zer + 1;
                { αν είναι αρνητικός, αυξάνουμε τη μεταβλητή neg κατά ένα }
                If x < 0 then neg := neg + 1;
        End; { end of for }
        Writeln('Οι θετικοί αριθμοί είναι : ', pos);
        Writeln('Οι μηδενικοί αριθμοί είναι : ', zer);
        Writeln('Οι αρνητικοί αριθμοί είναι : ', neg);
        Repeat until keypressed;
End.
```

```
Program  Mathites;
{ Το πρόγραμμα αυτό υπολογίζει το σύνολο των μαθητών και τον μέσο όρο
βαθμολογίας για κάθε τάξη, για τρεις τάξεις A, B, C και για N μαθητές }
Uses
        Crt;
Var
        i, N, b, sa, sb, sc, mo_a, mo_b, mo_c : Byte;
        sa_b, sb_b, sc_b : Integer;
        onoma : String;
        taxi : Char;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των μαθητών : ');
        Readln(N);
        sa := 0;        { μετρητής μαθητών A' τάξης }
        sb := 0;        { μετρητής μαθητών B' τάξης }
        sc := 0;        { μετρητής μαθητών C' τάξης }
        sa_b : = 0;     { αθροιστής βαθμών μαθητών A' τάξης }
        sb_b := 0;      { αθροιστής βαθμών μαθητών B' τάξης }
        sc_b := 0;      { αθροιστής βαθμών μαθητών C' τάξης }
        For i := 1 to N do
        Begin
                Writeln('Δώσε όνομα, βαθμό και τάξη μαθητή : ');
                Readln(onoma, b, taxi);
                If taxi = 'A' then      { ο μαθητής είναι στην A' τάξη }
                        Begin
                                sa := sa + 1;
                                sa_b := sa_b +b;
                        End;
                If taxi = 'B' then      { ο μαθητής είναι στην B' τάξη }
                        Begin
                                sb := sb + 1;
                                sb_b := sb_b +b;
                        End;
                If taxi = 'C' then      { ο μαθητής είναι στην C' τάξη }
                        Begin
                                sc := sc + 1;
                                sc_b := sc_b +b;
                        End;
        End; { end of for }
        mo_a := sa_b/sa;      { μέσος όρος βαθμών μαθητών A' τάξης }
        mo_b := sb_b/sb;      { μέσος όρος βαθμών μαθητών B' τάξης }
        mo_c := sc_b/sc;      { μέσος όρος βαθμών μαθητών C' τάξης }
        Writeln('Το πλήθος των μαθητών και οι μέσοι όροι βαθμολογίας τους
                ανά τάξη είναι οι εξής : ', sa, mo_a, sb, mo_b, sc, mo_c);
        Repeat until keypressed;
End.
```

```
Program MultiAdd;
{ Το πρόγραμμα αυτό μάς δείχνει πώς μπορούμε να καταχωρήσουμε Ν ονόματα με τη χρήση της εντολής Readkey }
Uses crt;
Var
        Onoma, Eponymo, Address : String;
        Apanthsh : Char;
Begin
        Repeat
                Clrscr;         { καθαρίζει η οθόνη }
                GotoXY(01,01);Write('Όνομα    :');
                GotoXY(01,02);Write('Επώνυμο :');
                GotoXY(01,03);Write('Διεύθυνση:');
                GotoXY(12,01);ReadLn(Onoma);
                GotoXY(12,02);ReadLn(Eponymo);
                GotoXY(12,03);ReadLn(Address);
                Apanthsh := Readkey;      { διαβάζει έναν χαρακτήρα }
                { αν η απάντηση είναι [Ο]χι τότε τελειώνει το πρόγραμμα }
                { πρέπει να ελέγξουμε και τα αγγλικά και τα ελληνικά Ο, }
                { μικρά και κεφαλαία }
        Until (Apanthsh='Ο') or (Apanthsh='ο') or (Apanthsh='o') or
                (Apanthsh='O');
        Repeat Until KeyPressed; { με την εντολή αυτή, το πρόγραμμα περιμένει μέχρι να πατήσουμε ένα οποιοδήποτε πλήκτρο για να τερματίσει }
End.
```

```
Program ReadKey_Test;
{ Το πρόγραμμα αυτό χρησιμοποιεί την εντολή ReadKey και μας δίνει ένα αποτέλεσμα τύπου char και ειδικότερα μάς επιστρέφει το πλήκτρο που πατήσαμε και τη μορφή του σε κεφαλαίο γράμμα }
Uses crt;
Var
        Key : Char;
Begin
        While Key <> 'T' Do Begin
                Key := ReadKey;
                Writeln('Πάτησες το πλήκτρο ', Key, ' ', UpCase(Key));
        End    { While }
        Repeat Until KeyPressed;
End.
```

Program ReadKey_Test2;
{ *Το πρόγραμμα αυτό χρησιμοποιεί την εντολή ReadKey και είναι παρόμοιο με το προηγούμενο, μόνο που μας δίνει σαν αποτέλεσμα και τον ascii κωδικό του πλήκτρου και τελειώνει όταν πατηθεί το πλήκτρο Esc, που έχει ascii κωδικό 27* }

```
Uses crt;
Var
        Key : Char;
Begin
        Repeat
                Key := ReadKey;
                Writeln('Πάτησες το πλήκτρο ', Key, ' με κωδικό : ', Ord(Key));
        Until Ord(Key)=27;
        Repeat Until KeyPressed;
End.
```

Program Strs;
{ *Το πρόγραμμα αυτό επεξεργάζεται με διάφορες συναρτήσεις της Pascal ένα string χαρακτήρων και μας δείχνει με ποιον τρόπο μπορούμε να βρούμε το μήκος του string, να απομονώσουμε ένα-ένα τα γράμματα του string και να κάνουμε και άλλες ενδιαφέρουσες εργασίες πάνω σ' αυτό* }

```
Uses crt;
Var
        Symbola : String;
Begin
        ClrScr;
        Write('Δώσε Μερικά Σύμβολα : ');Read(Symbola);
        WriteLn;        { αφήνει μια κενή γραμμή }
        WriteLn('Εντολή             Επεξήγηση                   Αποτέλεσμα');
        WriteLn( '==============================');
        WriteLn('Symbola - Τα σύμβολα που έδωσες είναι : ', Symbola);
        WriteLn('Length - Το μήκος των συμβόλων είναι :', Length(Symbola);
        WriteLn('Symbola[1] - Το 1ο σύμβολο είναι το : ', Symbola[1]);
        WriteLn('Ord - Ο κωδικός ascii του 1ου συμβόλου : ', Ord(Symbola[1]);
        WriteLn('Succ - Μετά το 1ο σύμβολο είναι : ', Symbola[succ(1)]);
        WriteLn('Pred - Πριν από το 4ο σύμβολο είναι : ', Symbola[pred(4)]);
        WriteLn;        { αφήνει μια κενή γραμμή }
        Repeat Until KeyPressed;
End.
```

Program Antistrofh;

```
{ Το πρόγραμμα αυτό διαβάζει έναν τριψήφιο ακέραιο αριθμό και υπολογίζει
και εκτυπώνει τον ακέραιο που προκύπτει από την αντιστροφή των ψηφίων
του }
Uses crt;
Var
        x, x1 : Integer;
        p1, p2, y1, y2 : Byte;
Begin
        Clrscr;
        Repeat
                Writeln('Δώσε έναν τριψήφιο ακέραιο αριθμό : ');
                Readln (x);
        Until (x>=100) and (x<=999);
        { θα βγούμε από τον βρόχο μόνο όταν ο αριθμός είναι τριψήφιος }

        p1 := x div 100;        { το p1 περιέχει το πρώτο ψηφίο του αριθμού }
        y1 := x mod 100;        { το y1 είναι τα δύο τελευταία ψηφία }
        p2 := y1 div 10;        { το p2 είναι το μεσαίο ψηφίο του αριθμού }
        y2 := y1 mod 10;        { το y2 είναι το τελευταίο ψηφίο του αριθμού }

        { η επόμενη εντολή σχηματίζει τον αριθμό που αποτελείται από τα }
        { αντίστροφα ψηφία του δοθέντος αριθμού }
        x1 := y2*100 + p2*10 + p1;
        Writeln('Ο αριθμός από την αντιστροφή είναι ο :  ', x1);
        Repeat Until KeyPressed;
End.
```

Program mhden_3;

```
{ Σ' αυτό το πρόγραμμα δίνεται ένας δεκαδικός αριθμός K και κρατάμε μόνο
τα 2 πρώτα δεκαδικά του ψηφία - για να λύσουμε αυτό το πρόβλημα, πρώτα
πολλαπλασιάζουμε αυτόν τον αριθμό με το 100 και στον προκύπτοντα αριθμό
προσθέτουμε το 0.5 για να στρογγυλοποιηθεί, αποκόπτουμε τα δεκαδικά του
ψηφία για να τον κάνουμε ακέραιο και μετά τον διαιρούμε με το 100 }
Uses crt;
Var
        k, n : Real;
        p : Longint;
Begin
        Clrscr;
        Writeln('Δώσε τον αριθμό : ');
        Readln(k);
        n := k*100;             { πολλαπλασιάζουμε με 100 }
        p := int(n+0.5);        { προσθέτουμε το 0.5 και τον κάνουμε ακέραιο }
        n := p/100;             {διαιρούμε με 100 για να αποκτήσει 2 δεκαδικά }
        Writeln('Ο νέος αριθμός είναι τώρα ο : ', n);
        Repeat until keypressed;
End.
```

Program Kerma;

{ Στο πρόγραμμα αυτό δίνεται ο μισθός ενός υπαλλήλου Μ και του κάνουμε κερματική ανάλυση, δηλ. βρίσκουμε από πόσα χαρτονομίσματα των 5.000, 1.000, 500, 100, 50, 20, 10 και 5 δραχμών αποτελείται. Υποθέτουμε βέβαια ότι ο μισθός είναι στρογγυλοποιημένος στο τάλιρο, αν δεν είναι, τότε υπάρχει το γνωστό μας πρόγραμμα που τον στρογγυλοποιεί }

```
Uses crt;
Var
        m : Longint;
        p5000, p1000, p500, 5100, p50, p20, p10, p5 : Byte;
        y : Integer;
Begin
        Clrscr;
        Writeln('Δώσε τον μισθό : ');
        Readln(m);
        p5000 := m div 5000;        { πλήθος των πεντοχίλιαρων }
        y := m mod 5000;            { το y χρησιμοποιείται στη νέα διαίρεση }
        p1000 := y div 1000;        { πλήθος των χιλιάρικων }
        y := y mod 1000;
        p500 := y div 500;          { πλήθος των πεντακοσάρικων }
        y := y mod 500;
        p100 := y div 100;          { πλήθος των εκατοστάρικων }
        y := y mod 100;
        p50 := y div 50;            { πλήθος των πενηντάρικων }
        y := y mod 50;
        p20 := y div 20;            { πλήθος των εικοσάρικων }
        y := y mod 20;
        p10 := y div 10;            { πλήθος των δεκάρικων }
        y := y mod 10;
        p5 := y div 5;              { πλήθος των τάλιρων }
        Writeln('Ο μισθός αποτελείται από ', p5000, ' πεντοχίλιαρα ',
                p1000, ' χιλιάρικα ', p500, ' πεντακοσάρικα ',
                p100, ' εκατοστάρικα ', p50, ' πενηντάρικα ',
                p20,' εικοσάρικα ', p10,' δεκάρικα και ', p5, ' τάλιρα');
        Repeat until keypressed;
End.
```

Program mhden_1;

{ Σ' αυτό το πρόγραμμα δίνεται ένας ακέραιος αριθμός Κ και μετατρέπουμε τα 2 τελευταία ψηφία του σε μηδενικά, αφού τον στρογγυλοποιήσουμε στο πλησιέστερο μηδενικό - εδώ θα δούμε τον πρώτο τρόπο επίλυσης του προβλήματος αυτού με τη χρήση του πηλίκου και του υπολοίπου της διαίρεσης του αριθμού με το 100 }

```
Uses crt;
Var
        k, p, n : Longint;
        y : Byte;
Begin
        Clrscr;
        Writeln('Δώσε τον αριθμό : ');
        Readln(k);
        p := k div 100;         { πηλίκο της διαίρεσης με το 100 }
        y := k mod 100;         { υπόλοιπο της διαίρεσης με το 100 }
        If y < 50       { αν το υπόλοιπο είναι μικρότερο από 50 }
                Then n := p*100;
        If y >= 50      { αν το υπόλοιπο είναι μεγαλύτερο ή ίσο από 50 }
                Then n := p*100 + 100;
        Writeln('Ο νέος αριθμός είναι τώρα ο : ', n);
        Repeat until keypressed;
End.
```

Program mhden_2;

{ Ας δούμε τώρα και τον δεύτερο τρόπο επίλυσης του προβλήματος αυτού, όπου διαιρούμε πρώτα τον αριθμό με το 100, τον προκύπτοντα αριθμό τύπου real, τον στρογγυλοποιούμε με το 0.5 κάνοντάς τον ακέραιο και μετά τον πολλαπλασιάζουμε πάλι με το 100 }

```
Uses crt;
Var
        k, n : Longint;
        p : Real;
Begin
        Clrscr;
        Writeln('Δώσε τον αριθμό : ');
        Readln(k);
        p := k/100;             { δεκαδική διαίρεση με το 100 }
        n := int(p+0.5);        { στρογγυλοποίηση του αριθμού }
        n := n*100;             { πολλαπλασιασμός με το 100 }
        Writeln('Ο νέος αριθμός είναι τώρα ο : ', n);
        Repeat until keypressed;
End.
```

Program ypallhloi;

{ Σ' αυτό το πρόγραμμα, για Ν υπαλλήλους έχουμε τα εξής στοιχεία : ονοματεπώνυμο, γραμματικές γνώσεις (1=Δημοτικό, 2=Γυμνάσιο, 3=Πτυχιούχος) και ηλικία - ζητάμε να βρεθεί η μικρότερη ηλικία από κάθε κατηγορία γραμματικών γνώσεων }

```
Uses crt;
Var
        age, min1, min2, min3, N, i : Byte;
        onoma : String;
        category : Char;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των υπαλλήλων : ');
        Readln(N);
        min1 := 100; { η μικρότερη ηλικία για υπαλλήλους Δημοτικού }
        min2 := 100; { η μικρότερη ηλικία για υπαλλήλους Γυμνασίου }
        min3 := 100; { η μικρότερη ηλικία για υπαλλήλους Πτυχιούχους }
        For i := 1 To N Do Begin
                Writeln('Δώσε όνομα, κατηγορία και ηλικία : ');
                Readln(onoma); Readln(category); Readln(age);
                Case category of
                        '1' : If age < min1 then min1 := age;
                        '2' : If age < min2 then min2 := age;
                        '3' : If age < min3 then min3 := age;
                end; { Case }
        End; { For }
        Writeln('Η μικρότερη ηλικία από την 1η κατηγορία είναι : ', min1);
        Writeln('Η μικρότερη ηλικία από την 2η κατηγορία είναι : ', min2);
        Writeln('Η μικρότερη ηλικία από την 3η κατηγορία είναι : ', min3);
        Repeat until keypressed;
End.
```

Program fylo;
{ Σ' αυτό το πρόγραμμα δίνονται τα παρακάτω στοιχεία για Ν ανθρώπους : όνομα και κωδικός φύλου (1=γυναίκα, 2=άνδρας) - ζητάμε να βρεθούν πόσες είναι οι γυναίκες και πόσοι οι άνδρες }

```
Uses crt;
Var
        onoma : String;
        fylo : Char;
        sum_a, sum_g, N, i : Byte;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των ανθρώπων : '); Readln(N);
        sum_a := 0;             { μετρητής ανδρών }
        sum_g := 0;             { μετρητής γυναικών }
        For i := 1 To N Do Begin
                Writeln('Δώσε όνομα & φύλο : ');Readln(onoma);Readln(fylo);
                Case fylo of
                        '1' : sum_g := sum_g + 1;
                        '2' : sum_a := sum_a + 1;
                End; { Case }
        End; { For }
        Writeln('Οι γυναίκες είναι : ', sum_g);
        Writeln('Οι άνδρες είναι : ', sum_a);
        Repeat until keypressed;
End.
```

Program N_arithmoi;
{ Να διαβαστούν Ν αριθμοί και να βρεθεί το άθροισμά τους, το γινόμενό τους και ο μέσος όρος τους }

```
Uses crt;
Var
        N, i : Byte;
        sum, product, m_o, number : Longint;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των αριθμών : '); Readln(N);
        sum := 0;               { αθροιστής }
        product := 1;           { γινόμενο }
        For i := 1 To N Do Begin
                Writeln('Δώσε έναν αριθμό : '); Readln(number);
                sum : = sum + number;
                product := product * number;
        End; { For }
        m_o := sum/N;           { υπολογισμός μέσου όρου }
        Writeln('Το άθροισμά τους είναι : ', sum);
        Writeln('Το γινόμενό τους είναι : ', product);
        Writeln('Ο μέσος όρος τους είναι : ', m_o:10:2);
        Repeat until keypressed;
End.
```

```
Program Eidh;
{ Το πρόγραμμα αυτό διαβάζει για Ν είδη μιας αποθήκης τα στοιχεία ονομα-
σία και τιμή μονάδας - βρίσκει και εκτυπώνει το ακριβότερο είδος, αφού
πρώτα βάλει τις ονομασίες και τις τιμές των ειδών σ' έναν πίνακα }
Uses crt;
Bar
        i, N : Byte;
        max : Longint;
        onoma_max : String[30];
        onomasia : array[1..50] of String[30];
        timh : array[1..50] of Longint;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των ειδών : ');
        Readln(N);

        { καταχωρούμε την ονομασία και την τιμή του κάθε είδους }
        { σε δύο ξεχωριστούς πίνακες }
        For i:=1 to N Do Begin
                Writeln('Δώσε ονομασία είδους : ');
                Readln(onomasia[i]);
                Writeln('Δώσε τιμή είδους : ');
                Readln(timh[i]);
        End; { For }

        { θέτουμε την τιμή του πρώτου είδους σαν τη μεγαλύτερη και }
        { την συγκρίνουμε μ' όλες τις υπόλοιπες }
        max := timh[1];
        onoma_max := onomasia[1];
        For i:=2 To N Do Begin
                If timh[i] > max Then Begin
                        max := timh[i];
                        onoma_max := onomasia[i];
                End;{ If }
        End;{ For }

        Writeln('Το είδος με την ακριβότερη τιμή είναι το : ',onoma_max,'
                και κοστίζει ', max, ' δρχ.');
        Repeat until keypressed;
End.
```

Program Thleorash;
{ *Γίνεται μια έρευνα για τις ώρες που παρακολουθεί τηλεόραση το κοινό και τα αποτελέσματα καταχωρούνται στις εξής τρεις κατηγορίες :*
α. υψηλή (περισσότερες από 20 ώρες την εβδομάδα)
β. μεσαία (από 8 έως 20 ώρες)
γ. χαμηλή (λιγότερες από 8 ώρες)
ζητείται να διαβαστούν οι ώρες για 30 άτομα και να τυπωθεί το πλήθος της κάθε κατηγορίας }

```
Uses crt;
Var
        i, hours, category_1, category_2, caterogy_3 : Byte;
Begin
        Clrscr;
        category_1 := 0;      { μηδενίζουμε τους τρεις μετρητές }
        category_2 := 0;
        caterogy_3 := 0;
        For i:=1 to 30 Do Begin
                Writeln('Δώσε τις ώρες : '); Readln(hours);
                Case hours of
                        0..7    : category_1 := category_1 + 1;
                        8..20   : category_2 := category_2 + 1;
                        21..255: category_3 := category_3 + 1;
                End;{ Case }
        End; { For }
        Writeln('Οι τρεις κατηγορίες έχουν τις εξής τιμές : ', category_1,
                        category_2, category_3);
        Repeat until keypressed;
End.
```

Program Athroisma_Pinaka;
{ *Να διαβαστούν τα ακέραια στοιχεία ενός πίνακα πλήθους Ν και να βρεθεί το άθροισμά τους* }

```
Uses crt;
Var
        i, N : Byte;
        sum : Longint;
        pinakas : Array[1..100] of Integer;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των αριθμών : '); Readln(N);
        sum := 0;      { μηδενίζουμε τον αθροιστή }
        For i:=1 To N Do Begin
                Writeln('Δώσε αριθμό : '); Readln(pinakas[i]);
        End; { For }
        For i:=1 To N Do
                sum := sum + pinakas[i];
        Writeln('Το άθροισμα των αριθμών είναι : ', sum);
        Repeat until keypressed;
End.
```

```
Program Athlites;
{ Το πρόγραμμα αυτό καταγράφει τις προσπάθειες 10 αθλητών στο άλμα σε
μήκος και για κάθε αθλητή καταχωρούμε το όνομα και την επίδοσή του σε 6
προσπάθειες - το πρόγραμμα βρίσκει τον νικητή του αγωνίσματος και την ε-
πίδοσή του καθώς και την καλύτερη επίδοση του κάθε αθλητή - χρησιμο-
ποιούμε έναν πίνακα δύο διαστάσεων όπου καταχωρούμε τις επιδόσεις των
αθλητών }
Uses crt;
Var
        i, j : Byte;
        max_oliko : Real;     { καλύτερη επίδοση όλων των αθλητών }
        max : Real;           { καλύτερη επίδοση ενός αθλητή }
        onoma_max : String[30];    { όνομα αθλητή με την καλύτερη επίδοση }
        onoma : array[1..10] of String[30];      { πίνακας ονομάτων αθλητών }
        epidoseis : Array[1..10, 1..6] of Real;  { πίνακας επιδόσεων αθλητών }
Begin
        Clrscr;
        max_oliko := 0.0;
        { μηδενίζουμε την καλύτερη επίδοση όλων των αθλητών }

        For i:=1 to 10 Do Begin
                Writeln('Δώσε το όνομα του ', i, 'ου αθλητή : ');
                Readln(onoma[i]);
                max := 0.0;   { μηδενίζουμε την καλύτερη επίδοση του αθλητή }
                For j:=1 to 6 Do Begin
                        Writeln('Δώσε την ', j, 'η προσπάθεια του αθλητή : ');
                        Readln(epidoseis[i, j]);
                        If epidoseis[i, j] > max Then
                                max := epidoseis[i, j];
                        { καλύτερη επίδοση αθλητή }
                        If epidoseis[i, j] > max_oliko Then Begin
                        { καλύτερη επίδοση όλων των αθλητών }
                                max_oliko := epidoseis[i, j];
                                onoma_max := onoma[i];
                        End;{ If }
                End; { For-j }
                Writeln('Η καλύτερη επίδοση του αθλητή είναι : ',max);
        End;{ For-i }

        Writeln('Η καλύτερη επίδοση όλων των αθλητών είναι : ', max_oliko,
                ' και την έχει ο αθλητής : ', onoma_max);
        Repeat until keypressed;
End.
```

```
Program Mathites;
{ Δίνονται τα εξής στοιχεία για 20 μαθητές μιας τάξης :
        α. όνομα
        β. κωδικός φύλου (1=αγόρι, 2=κορίτσι)
        γ. βαθμός
να βρεθεί ποιο αγόρι έχει τον μικρότερο βαθμό στην τάξη και ποιο κορίτσι τον
μεγαλύτερο }
Uses crt;
Var
        i : Byte;
        max_bathmos, min_bathmos : Byte;
        onoma_max, onoma_min : String[30];
        onoma : array[1..20] of String[30];
        fylo, bathmos : Array[1..20] of Byte;
Begin
        Clrscr;
        { πρώτα διαβάζουμε όλα τα στοιχεία των μαθητών σε τρεις πίνακες }
        For i:=1 To 20 Do Begin
                Writeln('Δώσε όνομα μαθητή : ');
                Readln(onoma[i]);
                Writeln('Δώσε φύλο μαθητή : ');
                Readln(fylo[i]);
                Writeln('Δώσε βαθμό μαθητή : ');
                Readln(bathmos[i]);
        End; { For }

        max_bathmos := 0;          { αρχική τιμή για τον μεγαλύτερο βαθμό }
        min_bathmos := 21;         { αρχική τιμή για τον μικρότερο βαθμό }
        For i:=1 To 20 Do Begin
                Case fylo[i] of
                  1 : If bathmos[i] < min_bathmos Then Begin
                        min_bathmos := bathmos[i];
                        onoma_min := onomasia[i];
                      End;{ If }
                  2 : If bathmos[i] > max_bathmos Then Begin
                        max_bathmos := bathmos[i];
                        onoma_max := onomasia[i];
                      End;{ If }
                End;{ Case }

        Writeln('Η μαθήτρια με τον μεγαλύτερο βαθμό είναι η : ', max_onoma,
                        ' και έχει βαθμό : ', max_bathmos);
        Writeln('Ο μαθητής με τον μικρότερο βαθμό είναι ο : ', min_onoma,
                        ' και έχει βαθμό : ', min_bathmos);
        Repeat until keypressed;
End.
```

```
Program Max_Pinaka;
{ Να διαβαστούν τα ακέραια στοιχεία ενός πίνακα πλήθους N και να βρεθεί
ποιο είναι το μεγαλύτερο στοιχείο και σε ποια θέση }
Uses crt;
Var
        i, N, thesi : Byte;
        sum : Longint;
        max : Integer;
        pinakas : Array[1..100] of Integer;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των αριθμών : ');
        Readln(N);

        { διαβάζουμε τα στοιχεία του πίνακα }
        For i:=1 To N Do Begin
                Writeln('Δώσε έναν αριθμό : ');
                Readln(pinakas[i]);
        End; { For }

        { αρχικά θέτουμε σαν μεγαλύτερο το πρώτο στοιχείο του πίνακα }
        { και μετά συγκρίνουμε όλα τα στοιχεία του πίνακα }
        max := pinakas[1];
        thesi := 1;
        For i:=2 To N Do
                If pinakas[i] > max Then Begin
                        max := pinakas[i];
                        thesi := i;
                End;{ If }
        Writeln('το μεγαλύτερο στοιχείο του πίνακα είναι το : ', max,' και
                βρίσκεται στη θέση : ', thesi);
        Repeat until keypressed;
End.
```

```
Program Bathmoi_Mathiton;
{ Να διαβαστούν τα ονόματα και οι βαθμοί Ν μαθητών σε δύο πίνακες και να
βρεθεί ο καλύτερος μαθητής και ο βαθμός του }
Uses crt;
Var
        i, N : Byte;
        max : Byte;
        max_onoma : String[30];
        bathmos : Array[1..100] of Byte;
        onoma : Array[1..100] of String[30];
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των μαθητών : ');
        Readln(N);

        { διαβάζουμε τα ονόματα και τους βαθμούς των μαθητών σε }
        { δύο πίνακες }
        For i:=1 To N Do Begin
                Writeln('Δώσε όνομα μαθητή : ');
                Readln(onoma[i]);
                Writeln('Δώσε βαθμό μαθητή : ');
                Readln(bathmos[i]);
        End; { For }

        { αρχικά θέτουμε σαν μεγαλύτερο βαθμό τον βαθμό του }
        { πρώτου μαθητή και συγκρίνουμε όλα τα υπόλοιπα στοιχεία }
        max := bathmos[1];
        max_onoma := onoma[1];
        For i:=2 to N Do
                If bathmos[i] > max Then Begin
                        max := bathmos[i];
                        max_onoma := onoma[i];
                End;{ If }

        Writeln('Ο καλύτερος μαθητής είναι ο : ', max_onoma,' και είναι
                τον βαθμό : ', max);
        Repeat until keypressed;
End.
```

```
Program Men_Women;
{ Δίνονται τα εξής στοιχεία για Ν ανθρώπους : όνομα, κωδικός φύλου, (1 =
γυναίκα, 2 = άνδρας), ύψος, βάρος και ηλικία - ζητείται να βρεθεί ο μέσος
όρος ηλικίας των γυναικών που έχουν βάρος μεγαλύτερο των 60 κιλών και πό-
σοι άνδρες έχουν ύψος μεγαλύτερο του 1.80, βάρος μικρότερο των 85 κιλών
και ηλικία μικρότερη των 40 }
Uses crt;
Var
        i, N, g60, i180, i85, i40, mo_gyn, fylo, ypsos, baros, age : Byte;
        sum_age : Integer;
        onoma : String[30];
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των ατόμων : '); Readln(N);
        sum_age:=0; g60:=0; i180:=0; i85:=0; i40:=0;

        { εδώ γίνεται η καταχώρηση και η επεξεργασία των στοιχείων }
        For i:=1 To N Do Begin
                Writeln('Δώσε όνομα : '); Readln(onoma);
                Repeat
                        Writeln('Δώσε φύλο : '); Readln(fylo);
                Until fylo in [1..2];
                Writeln('Δώσε ύψος : '); Readln(ypsos);
                Writeln('Δώσε βάρος : '); Readln(baros);
                Writeln('Δώσε ηλικία : '); Readln(age);
                { επιλογή ανάλογα με το φύλο }
                Case fylo of
                        1 : If baros>60 Then Begin
                                sum_age := sum_age + age;
                                g60 := g60 + 1;
                            End; { If - Case-1 }
                        2 : Begin
                                If ypsos > 180 Then i180 := i180 +1;
                                If baros < 85 Then i85 := i85 + 1;
                                If age < 40 Then i40 := i40 + 1;
                            End; { Case-2 }
                End; { Case }
        End; { For }

        { εδώ γίνονται οι εκτυπώσεις }
        Writeln('Ο μέσος όρος ηλικίας των γυναικών με βάρος μεγαλύτερο
                των 60 κιλών είναι : ', Round(sum_age/g60));
        Writeln(i180, ' άνδρες έχουν ύψος μεγαλύτερο του 1.80 ', i85,' έχουν
        βάρος κάτω από 85 κιλά και ',i40,' έχουν ηλικία μικρότερη των 40);
        Repeat until keypressed;
End.
```

```
Program Taxeis;
{ Να διαβαστούν τα στοιχεία Ν μαθητών (όνομα, βαθμός και τάξη) από τρεις
τάξεις (Α, Β και C) και να εκτυπωθούν ανά τάξη ξεχωριστά }
Uses crt;
Var
        a_onoma : array[1..100] of String; { πίνακας ονομ.μαθητών Α' τάξης }
        b_onoma : array[1..100] of String; { πίνακας ονομ.μαθητών Β' τάξης }
        c_onoma : array[1..100] of String;          { πίνακας ονομ.μαθητών C'
τάξης }
        a_bathmos : array[1..100] of Byte; { πίνακας βαθμ.μαθητών Α' τάξης }
        b_bathmos : array[1..100] of Byte; { πίνακας βαθμ.μαθητών Β' τάξης }
        c_bathmos : array[1..100] of Byte; { πίνακας βαθμ.μαθητών C' τάξης }
        onoma : String;
        i, N, ia, ib, ic, bathmos : Byte;
        taxi : Char;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος Ν των μαθητών : ');
        Readln(N);
        ia := 0;        { μηδενισμός των μετρητών των μαθητών }
        ib := 0;        { για τις τρεις τάξεις }
        ic := 0;

        { εδώ γίνεται η καταχώρηση και η επεξεργασία των στοιχείων }
        For i:=1 To N Do Begin
                Writeln('Δώσε όνομα μαθητή : ');
                Readln(onoma);
                Writeln('Δώσε βαθμό μαθητή : ');
                Readln(bathmos);
                Repeat
                        Writeln('Δώσε τάξη μαθητή : ');
                        Readln(taxi)
                Until taxi In ['a', 'b', 'c'];    { δέχεται μόνο τιμές 'a', 'b' ή 'c' }

                { επιλογή ανάλογα με την τάξη }
                Case taxi of
                        'a' : Begin
                                ia := ia+1;    { αύξηση μετρητή τάξης }
                                a_onoma[ia] := onoma;
                                a_bathmos[ia] := bathmos;
                        End; { Case-a }

                        'b' : Begin
                                ib := ib+1;
                                b_onoma[ib] := onoma;
                                b_bathmos[ib] := bathmos;
                        End; { Case-b }
```

```
                    'c' : begin
                            ic := ic+1;
                            c_onoma[ic] := onoma;
                            c_bathmos[ic] := bathmos;
                          End;{ Case-c }
                End; { Case }
        End; { For }

        { εκτύπωση των τριών πινάκων των τάξεων A', B' και C' }
        Writeln('Τάξη A' :');
        For i:=1 To ia Do
                Writeln(i, '. ', a_onoma[i], ' ', a_bathmos[i]);

        Writeln('Τάξη B' :');
        For i:=1 To ib Do
                writeln(i, '. ', b_onoma[i], ' ', b_bathmos[i]);

        Writeln('Τάξη C' :');
        For i:=1 To ic Do
                Writeln(i, '. ', c_onoma[i], ' ', c_bathmos[i]);
        Repeat until keypressed;
End.
```

Program Molinsi;
{ Μια εταιρεία μετράει τη μόλυνση της ατμόσφαιρας και καταχωρεί τις ενδείξεις της σε ακέραιες τιμές από το 0 έως το 100 με μία μέτρηση για κάθε μέρα - να γίνει πρόγραμμα που να καταχωρεί τις τιμές της ατμοσφαιρικής μόλυνσης για 20 μέρες και να βρίσκει τις κορυφές της μόλυνσης, δηλ. αυτές που είναι μεγαλύτερες από τη μόλυνση της προηγούμενης και της επόμενης μέρας - για κάθε κορυφή το πρόγραμμα θα τυπώνει τη μέρα που έγινε και τη μόλυνση που υπήρχε εκείνη τη μέρα }

```
Uses crt;
Var
        molinsi : Array[1..20] of byte;
        i : Byte;
Begin
        Clrscr;

        { εδώ γίνεται η καταχώρηση των δεδομένων της μόλυνσης }
        { και ελέγχουμε αν η τιμή της μόλυνσης είναι από 0 έως 100 }
        For i:=1 To 20 Do Begin
                Repeat
                        Writeln('Δώσε τη μόλυνση της ', i, 'ης ημέρας : ');
                        Readln(molinsi[i]);
                Until molinsi[i] in [0..100];
        End; { For }

        { εδώ συγκρίνουμε τη μόλυνση της ημέρας i με τις μολύνσεις }
        { της επόμενης και της προηγούμενης ημέρας απ' αυτήν και ανάλογα }
        { εκτυπώνουμε }
        For i:=2 To 19 Do
                If (molinsi[i]>molinsi[i+1]) and (molinsi[i]>molinsi[i-1]) Then
                  Writeln('H ',i,'η ημέρα έχει κορυφή με τιμή ',molinsi[i]);

        { κάνουμε και μια εκτύπωση όλων των δεδομένων της μόλυνσης }
        For i:=1 To 20 Do
                Write(molinsi[i], ' ');

        Repeat until keypressed;
End.
```

```
Program Ptiseis;
{ Μια εταιρεία μεταφορών χρησιμοποιεί ένα δικό της αεροπλάνο για τις μεταφορές που κάνει στο εξωτερικό - το αεροπλάνο μπορεί κάθε φορά να μεταφέρει ένα μέγιστο συνολικό φορτίο - τα εμπορεύματα είναι συσκευασμένα σε πακέτα και κάθε πακέτο έχει έναν αύξοντα αριθμό και ένα βάρος - να γίνει πρόγραμμα που να καταχωρεί τα βάρη των πακέτων σε τόνους με τη σειρά που δίνονται και να υπολογίζει και να τυπώνει πόσες αεροπορικές πτήσεις θα χρειαστούν για να μεταφερθούν όλα τα πακέτα και πόσο συνολικό φορτίο θα μεταφέρει η κάθε πτήση }
Uses crt;
Const
        max_load = 100;{ 100 τόνοι είναι το μέγιστο φορτίο του αεροπλάνου }
Var
        i, weight, ptisi, sum : Byte;
Begin
        Clrscr;
        sum := 0;
        ptisi := 0;

        { εδώ γίνεται η καταχώρηση και η επεξεργασία των δεδομένων }
        For i:=1 To 20 Do Begin { υποθέτουμε ότι υπάρχουν 20 πακέτα }
                Writeln('Δώσε το βάρος του πακέτου : '); Readln(weight);
                sum := sum + weight;  { πρόσθεση του βάρους του πακέτου }

                If sum > max_load Then Begin
                { τα πακέτα ξεπέρασαν το μέγιστο φορτίο }
                        ptisi := ptisi + 1;     { αυξάνουμε τις πτήσεις }
                        sum := sum - weight; { αφαιρούμε το τελευταίο πακέτο }
                        Writeln('Θα φύγει η ', ptisi, 'η πτήση με βάρος ', sum);
                        sum := weight;          { το sum παίρνει νέα τιμή }
                End; { If }

                If sum = max_load Then Begin
                { τα πακέτα είναι ίσα με το μέγιστο φορτίο }
                        ptisi := ptisi + 1;     { αυξάνουμε τις πτήσεις }
                        Writeln('Θα φύγει η ', ptisi,' η πτήση με βάρος ', sum);
                        sum := 0;               { το sum μηδενίζεται }
                End;{ If }
        End; { For }

        If sum>0 Then Do Begin
        { αν υπάρχουν πακέτα που έχουν απομείνει }
                ptisi := ptisi + 1;             { αυξάνουμε τις πτήσεις }
                Writeln('Θα φύγει η ', ptisi, 'η πτήση με βάρος ', sum);
        End;{ If }
        Repeat until keypressed;
End.
```

Program Taxinomisi_Pinaka;

{ Να ταξινομηθεί κατά αύξουσα σειρά πίνακας Α με Ν στοιχεία - θα εφαρμοσθεί η μέθοδος Bubble-Sort σύμφωνα με την οποία σαρώνουμε συνέχεια τον πίνακα και αν κάποια στοιχεία δεν είναι στη σωστή σειρά, τότε τα κάνουμε ανταλλαγή και βάζουμε την τιμή 1 σε μια σημαία flag - όταν σαρώσουμε τον πίνακα και η flag είναι ίση με μηδέν, τότε αυτό θα σημαίνει ότι ο πίνακας είναι ταξινομημένος }

```
Uses crt;
Var
        a : Array[1..100] of Integer;
        temp : Integer;
        i, N, flag : Byte;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των στοιχείων : '); Readln(N);

        { εδώ γίνεται η καταχώρηση των δεδομένων }
        For i:=1 To N Do Begin
                Writeln('Δώσε το ', i, 'ο στοιχείο : '); Readln(a[i]);
        End; { For }

        { εδώ γίνεται η επεξεργασία των στοιχείων του πίνακα }
        Repeat
                flag := 0;      { αρχικά μηδενίζουμε τη flag }
                For i:=1 To N-1 Do Begin
                        If a[i] > a[i+1] Then Begin
                                { εδώ γίνεται η ανταλλαγή των τιμών }
                                temp := a[i];
                                a[i] := a[i+1];
                                a[i+1] := temp;
                                flag := 1;      { η flag γίνεται ίση με 1 }
                        End; { If }
                End; { For }
        Until flag = 0;
        { όταν η flag παραμείνει ίση με 0, αυτό θα σημαίνει ότι πίνακας }
        { είναι ταξινομημένος }

        { εμφανίζουμε τα στοιχεία του ταξινομημένου πίνακα }
        For i:=1 To N Do Begin
                Writeln('Το ', i, 'ο στοιχείο του πίνακα είναι : ', a[i]);
        End; { For }

        Repeat until keypressed;
End.
```

```
Program Athroisma_Pinaka;
{Να βρεθεί το άθροισμα των στοιχείων πίνακα Α με Ν γραμμές και Μ στήλες}
Uses crt;
Var
        i, j, n, m : Byte;
        sum : Longint;
        a : Array[1..100, 1..100] of Integer;
Begin
        Clrscr;
        Writeln('Δώσε το N : ');
        Readln(n);
        Writeln('Δώσε το M : ');
        Readln(m);
        { μηδενίζουμε τον αθροιστή }
        sum := 0;

        {εδώ γίνεται η καταχώρηση και η άθροιση των στοιχείων του πίνακα}
        For i:=1 To n Do Begin              { οι γραμμές του πίνακα }
                For j:=1 To m Do Begin      { οι στήλες του πίνακα }
                        Writeln('Δώσε το στοιχείο : ', i, ', ', j); Readln(a[i, j]);
                        sum := sum + a[i, j];
                End; { For-j }
        End; { For-i }

        {εδώ γίνεται η εκτύπωση του αθροίσματος των στοιχείων του πίνακα}
        Writeln('Το άθροισμα των στοιχείων του πίνακα είναι : ', sum);

        Repeat until keypressed;
End.
```

```
Program Max_Pinaka;
{ Να βρεθεί το μεγαλύτερο στοιχείο πίνακα Α με Ν γραμμές και Μ στήλες }
Uses crt;
Var
        i, j, n, m : Byte;
        max : Integer;
        a : Array[1..100, 1..100] of Integer;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των γραμμών : ');
        Readln(n);
        Writeln('Δώσε το πλήθος των στηλών : ');
        Readln(m);

        { εδώ γίνεται η καταχώρηση των στοιχείων του πίνακα }
        For i:=1 To n Do Begin
                For j:=1 To m Do Begin
                        Writeln('Δώσε το στοιχείο : ', i ,' ,' , j);
                        Readln(a[i, j]);
                End; { For-j }
        End; { For-i }

        { θέτουμε σαν max το στοιχείο του πίνακα που είναι στη θέση 1,1 }
        max := a[1,1];

        { εδώ ψάχνουμε για το μεγαλύτερο στοιχείο του πίνακα }
        For i:=1 To n Do Begin
                For j:=1 To m Do Begin
                        If a[i, j]) > max Then
                                max := a[i, j];
                End; { for-j }
        End; { for-i }

        { εδώ εκτυπώνουμε το μεγαλύτερο στοιχείο του πίνακα }
        Writeln('Το μεγαλύτερο στοιχείο του πίνακα είναι το : ', max);
        Repeat until keypressed;
End.
```

```
Program Athroisma_Grammhs_Pinaka;
{ Να βρεθεί το άθροισμα κάθε γραμμής πίνακα A (NXM) και να καταχωρηθεί
σ' έναν πίνακα B(N) }
Uses crt;
Var
        i, j, n, m : Byte;
        sum : Longint;
        a : Array[1..100, 1..100] of Integer;
        b : Array[1..100] of Longint;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των γραμμών : ');
        Readln(n);
        Writeln('Δώσε το πλήθος των στηλών : ');
        Readln(m);

        { εδώ γίνεται η καταχώρηση των στοιχείων του πίνακα }
        For i:=1 To n Do Begin
                For j:=1 To m Do Begin
                        Writeln('Δώσε το στοιχείο : ', i, ' ,', j); Readln(a[i, j]);
                End;{ For-j }
        End; { For-i }

        { εδώ γίνεται το διάβασμα ανά γραμμή του πίνακα και η πρόσθεση }
        { των στοιχείων της κάθε γραμμής }
        For i:=1 To n Do Begin
                sum := 0;
                { μηδενίζουμε το άθροισμα πριν από την }
                { επεξεργασία της γραμμής }
                For j:=1 To m Do Begin
                        sum := sum + a[i, j];
                End; { For-j }
                b[i] := sum;
                { το άθροισμα της i γραμμής καταχωρείται στη i θέση του }
                { πίνακα B }
        End; { For-i }

        Repeat until keypressed;
End.
```

```
Program Thetikoi_Arnitikoi;
{ Να βρεθεί και να εκτυπωθεί το πλήθος των θετικών και των αρνητικών στοι-
χείων κάθε γραμμής ενός πίνακα A(NXM) }
Uses crt;
Var
        i, j, n, m, pos, neg : Byte;
        a : Array[1..100, 1..100] of Integer;
Begin
        Clrscr;
        Writeln('Δώσε το πλήθος των γραμμών : ');
        Readln(n);
        Writeln('Δώσε το πλήθος των στηλών : ');
        Readln(m);

        { εδώ γίνεται η καταχώρηση των στοιχείων του πίνακα }
        For i:=1 To n Do Begin
                For j:=1 To m Do Begin
                        Writeln('Δώσε το στοιχείο : ', i, ',', j);readln(a[i, j]);
                End; { For-j }
        End; { For-i }

        { εδώ γίνεται η επεξεργασία των στοιχείων του πίνακα }
        For i:=1 To n Do Begin
                pos := 0;
                neg := 0;
                { μηδενίζουμε τα αθροίσματα πριν από την }
                { επεξεργασία των στοιχείων της γραμμής }
                For j:=1 To m Do Begin
                        If a[i, j] > 0 Then
                                pos := pos + 1;
                        If a[i, j] < 0 Then
                                neg := neg +1 ;
                End; { For-j }
                { τυπώνουμε τα δύο αθροίσματα }
                Writeln('Η γραμμή ', i, ' έχει ', pos, ' θετικούς και ', neg,
                        'αρνητικούς αριθμούς');
        End; { For-i }

        Repeat until keypressed;
End.
```

```
Program Eteria;
{ Μια εταιρία έχει έξι υποκαταστήματα - για κάθε υποκατάστημα στο τέλος
κάθε μήνα δίνονται οι πωλήσεις κάθε εβδομάδας - να γίνει αλγόριθμος και λο-
γικό διάγραμμα που να διαβάζει τα στοιχεία τεσσάρων εβδομάδων για κάθε
υποκατάστημα, να εκτυπώνει πίνακα με τις πωλήσεις και να βρίσκει τη μεγα-
λύτερη πώληση, ποιο υποκατάστημα την έκανε και σε ποια εβδομάδα }
Uses crt;
Var
        polisis : Array[1..6, 1..4] of Longint;
        max : Longint;
        i, j, i_max, j_max : Byte;
Begin
        Clrscr;
        {εδώ γίνονται οι καταχωρήσεις των πωλήσεων των }
        { 6 υποκαταστημάτων για 4 εβδομάδες }
        For i:=1 To 6 Do Begin
                For j:=1 To 4 Do Begin
                        Writeln('Δώσε τις πωλήσεις της ', j, 'ης εβδομάδας
                          του ', i, 'ου υποκαταστήματος : '); Readln(polisis[i, j]);
                End; { For-j }
        End; { For-i }

        { εκτύπωση των πωλήσεων ανά υποκατάστημα και ανά εβδομάδα }
        For i:=1 To 6 Do Begin
                Writeln('οι πωλήσεις του ', i, 'ου υποκαταστήματος είναι : ');
                Writeln('-----------------------------------------------------------------');
                For j:=1 To 4 Do Begin
                        Write(j, 'η εβδομάδα : '); Writeln(polisis[i, j]);
                End; { For-j }
                Writeln;
        End; { For-i }

        max := polisis[1,1];
        i_max := 1; j_max := 1;
        { θέτουμε σαν αρχική τιμή του max τις πωλήσεις του 1ου }
        { υποκατ. την 1η εβδομάδα και συγκρίνουμε με τις υπόλ. πωλήσεις }
        For i:=1 To 6 Do Begin
                For j:=1 To 4 Do Begin
                        If polisis[i,j] > max Then Begin
                                max := polisis[i, j]; i_max := i; j_max := j;
                        End; { If }
                End; { For-j }
        End; { For-i }

        { εκτυπώνουμε τα στοιχεία για τις μέγιστες πωλήσεις }
        Writeln('Οι μεγαλύτερες πωλήσεις είναι ', max, ' και τις έκανε το ',
                i_max, 'ο υποκατάστημα, την ', j_max, 'η εβδομάδα');
        Repeat until keypressed;
End.
```

```
Program Oplites;
{ Δίνονται τα εξής στοιχεία για 20 νέους οπλίτες που πρόκειται να κατατα-
γούν στα διάφορα όπλα του στρατού : ονοματεπώνυμο και όπλο κατάταξης
(1=στρατός ξηράς, 2=ναυτικό, 3=αεροπορία) - να γίνει εκτύπωση για τα
τρία όπλα ξεχωριστά }
Uses crt;
Var
        i, i_stratos, i_aeroporia, i_nautiko, oplo : Byte;
        name : String[30];
        stratos, aeroporia, nautiko : Array[1..20] of String[30];
Begin
        Clrscr;
        i_stratos :=0;          { μηδενισμός των μετρητών }
        i_aeroporia := 0;
        i_nautiko :=0;

        { καταχώρηση και επεξεργασία των στοιχείων των οπλιτών }
        For i:=1 To 20 Do Begin
                Writeln('Δώσε το όνομα : '); Readln(name);
                Repeat
                        Writeln('Δώσε το όπλο : '); Readln(oplo);
                Until oplo in [1..3];
                Case oplo of
                        1 : Begin
                                i_stratos := i_stratos + 1;
                                stratos[i_stratos] := name;
                            End; { Case-1 }
                        2 : Begin
                                i_aeroporia := i_aeroporia + 1;
                                aeroporia[i_aeroporia] := name;
                            End; { Case-2 }
                        3 : Begin
                                i_nautiko := i_nautiko + 1;
                                nautiko[i_nautiko] := name;
                           End; { Case-3 }
                End; { Case }
        End; { For }
        { εδώ γίνεται η εκτύπωση των στοιχείων και των τριών πινάκων }
        Writeln('Στρατός : ');
        For i:=1 To i_stratos Do
                Writeln(stratos[i]);
        Writeln; Writeln('Αεροπορία : ');
        For i:=1 tTo i_aeroporia Do
                writeln(aeroporia[i]);
        Writeln; Writeln('Ναυτικό : ');
        For i:=1 To i_nautiko Do
                Writeln(nautiko[i]);
        Repeat until keypressed;
End.
```

Το παρακάτω πρόγραμμα περιέχει μια διαδικασία που κεντράρει μια γραμμή στην οθόνη.

```
Program Center;
Uses crt;
Var
        row : Byte;
        title : String;

Procedure center_line(message : String; line : Byte);
Begin
        GotoXY(40 - Length(message) div 2, line);
        Writeln(message);
End;

Begin { Κύριο Πρόγραμμα }
        row := 2;
        title := 'Κάθε γραμμή κειμένου κεντράρεται.';
        Clrscr;
        center_line(title, row);              { κλήση της διαδικασίας }
        center_line('--------', row+1);
        center_line('Borland Turbo Pascal !', row+2);
End.
```

Το αποτέλεσμα του προγράμματος είναι το εξής :

Κάθε γραμμή κειμένου κεντράρεται

Borland Turbo Pascal !

Το παράδειγμα που ακολουθεί περιέχει δύο τοπικές μεταβλητές και απαιτεί από τον χρήστη να δώσει έναν αριθμό, προκειμένου να βρεθεί το παραγοντικό του. Το παραγοντικό π.χ. του 6 είναι το γινόμενο 1*2*3*4*5*6.

```
Program local_variables;
Var
        num : Byte;

Procedure factor(value : Byte);
Var
        factorial : Real;
        count : Byte;
Begin
        factorial := 1.0;
        For count := 1 To value Do
                factorial := factorial * count;
        Write('Το παραγοντικό του ', value,' είναι : ' );
        Writeln(factorial);
End; { Τέλος της διαδικασίας Factor }

Begin { Κύριο πρόγραμμα }
        Write('Δώστε έναν αριθμό μικρότερο του 34 : ');
        Readln(num);
        Factor(num);
End. { Τέλος του προγράμματος }
```

Ένα παράδειγμα από τη λειτουργία του προγράμματος είναι το εξής :

Δώστε έναν αριθμό μικρότερο του 34 : 5
Το παραγοντικό του 5 είναι : 1.200000000E+02

Στο πρόγραμμα που ακολουθεί παρουσιάζεται το πέρασμα παραμέτρων με τιμή :

```
Program byvalue;
Var
        global_var : Integer;

Procedure proc(local_var : Integer);
Begin
        Writeln('Τοπική μεταβλητή = ', local_var);
        local_var := 333;
        Writeln('Τοπική μεταβλητή = ', local_var);
End;

Begin { Κύριο πρόγραμμα }
        global_var := 5;
        proc(global_var);
        Writeln('Καθολική μεταβλητή = ', global_var);
End.
```

Το αποτέλεσμα του προγράμματος θα είναι :

Τοπική μεταβλητή = 5
Τοπική μεταβλητή = 333
Καθολική μεταβλητή = 5

Βλέπουμε ότι η μεταβλητή global_var δεν επηρεάζεται από τη διαδικασία και ότι δίνει την τιμή της στην local_var.

Το παρακάτω πρόγραμμα παρουσιάζει τη μέθοδο αυτή (με αναφορά):

```
Program byref;
Var
        var, var2 : Integer;

Procedure swap_vars(Var var1 : Integer; Var var2 : Integer);
Var
        temp : Integer;
Begin
        temp := var1;
        var1 := var2;
        var2 := temp;
End;

Begin
        var1 := 55;
        var2 := 99;
        Writeln('Μεταβλητή1 = ', var1, 'Μεταβλητή2 = ', var2);
        swap_vars(var1, var2);
        Writeln('Μεταβλητή1 = ', var1, 'Μεταβλητή2 = ', var2);
End.
```

Το αποτέλεσμα του προγράμματος θα είναι :

Μεταβλητή1 = 55 Μεταβλητή2 = 99
Μεταβλητή1 = 99 Μεταβλητή2 = 55

Όταν περνάμε μια παράμετρο με αναφορά, θα πρέπει να περνάμε τη μεταβλητή και όχι την τιμή της. Συνίσταται να περνούν οι μεταβλητές με αναφορά μόνο όταν απαιτείται η διαδικασία να αλλάξει το περιεχόμενό τους.

Δηλαδή, η παράμετρος θα πρέπει να είναι όνομα μεταβλητής :

```
swap_vars(global_1, global_2);
```

και όχι όνομα σταθεράς :

```
swap_vars(55, 99);
```

Το παρακάτω πρόγραμμα που περιέχει μια συνάρτηση που υψώνει έναν αριθμό σε κάποια δύναμη.

```
Program Funct;
Var
        num, expo, powr : Real;

Function power(base, exponent : Real) : Real;
Begin
        If base >0 Then
                Power := Exp(exponent * Ln(base))
        Else
                Power := -1.0;
End; { end of power() }

Begin
        Writeln('Δώστε έναν αριθμό : ');
        Readln(num);
        Writeln('Δώστε τον εκθέτη : ');
        Readln(expo);
        powr := Power(num, expo);
        Writeln(num, '^', expo, ' = ', powr);
End.
```

Το παραπάνω πρόγραμμα Funct περιμένει την εισαγωγή δύο αριθμών, της βάσης και του εκθέτη και στη συνέχεια καλεί τη συνάρτηση power, η οποία υψώνει τον αριθμό στην αντίστοιχη δύναμη.

Το παρακάτω πρόγραμμα παρουσιάζει τον φωλιασμό των διαδικασιών :

```
Program Hideproc;
Var
        globl : Integer;

Procedure proc(p_parm : Integer);
Var
        p_locl : Integer;
        Procedure hidden(hidn_parm : Integer);
        Var
                hidn_locl : Integer;
        Begin
                Writeln('Η διαδικασία hidden μπορεί να δει τις : globl, p_parm,
                        p_locl, hidn_parm, hidn_locl');
        End; { τέλος της διαδικασίας hidden }
Begin
        Writeln('η διαδικασία proc μπορεί να δει τις : globl, p_parm, p_locl');
        hidden(44);
End; { τέλος της διαδικασίας proc }

Begin { κύριο πρόγραμμα }
        Writeln('Το κύριο πρόγραμμα μπορεί να δει την : globl');
        proc(99);
End.
```

Το αποτέλεσμα θα είναι :

το κύριο πρόγραμμα μπορεί να δει την : globl
η διαδικασία proc μπορεί να δει τις : globl, p_parm, p_locl
η διαδικασία hidden μπορεί να δει τις : globl, p_parm, p_locl,
hidn_parm, hidn_locl

Το παραπάνω πρόγραμμα περιέχει δύο διαδικασίες, την proc και την hidden. Η διαδικασία hidden είναι ορατή μόνο μέσα στην proc, στην οποία και περιέχεται.

Στο βαθύτερο σημείο του προγράμματος, στο εσωτερικό δηλ. της hidden, όλες οι μεταβλητές του προγράμματος γίνονται ορατές. Η διαδικασία hidden μπορεί να δει τις δικές της τοπικές μεταβλητές, αυτές της διαδικασίας proc καθώς και τις καθολικές μεταβλητές του προγράμματος.

Η διαδικασία proc μπορεί να δει μόνο τις δικές της τοπικές μεταβλητές καθώς και τις καθολικές. Το κύριο πρόγραμμα έχει την πιο περιορισμένη ορατότητα. Μπορεί να δει μόνο τις καθολικές μεταβλητές.

Στο παρακάτω πρόγραμμα χρησιμοποιείται η αναδρομική συνάρτηση factor για την εύρεση του παραγοντικού ενός αριθμού :

```
Program Recurse;
Uses crt;
Var
        num : Byte;
        result : Real;

Function Factor(value : Byte ) : Real;
Begin
        If value > 1 Then
                factor := value * factor(value-1)
        Else
                factor := 1.0;
End; { Factor }

Begin { κύριο πρόγραμμα }
        Write('Δώσε έναν αριθμό : ');
        Readln(num);
        result := Factor(num);
        Write('Το παραγοντικό του ', num, ' είναι : ');
        Writeln(result);
End.
```

Η διαφορά μεταξύ της απλής κλήσης μιας συνάρτησης και της αναδρομικής, είναι στο ότι η τελευταία περιέχει μια δήλωση που καλεί τον εαυτό της :

```
factor := value * factor(value-1)
```

Η δήλωση If στην αρχή της συνάρτησης θέτει μια συνθήκη η οποία εμποδίζει τη συνάρτηση να καλεί συνέχεια τον εαυτό της. Κάθε αναδρομική συνάρτηση ή διαδικασία πρέπει να έχει έναν τέτοιο μηχανισμό εξόδου.

Με λίγη προσοχή ένα πρόγραμμα μπορεί να εκτελέσει τρεις διαφορετικές λειτουργίες : εκτύπωση δεδομένων στον εκτυπωτή, στην οθόνη ή σε αρχείο. Δείτε το παρακάτω πρόγραμμα, όπου μια εντολή case εκχωρεί την κατάλληλη μεταβλητή ονόματος αρχείου, ανάλογα με την επιλογή του χρήστη :

```
Program testfile;
Uses Crt;
Var
        datafile : Text;
        FileName : String;
        i, epil : Integer;
        flag : Boolean;
Procedure Menu;
Begin
        ClrScr;
        epil := 0;
        GotoXY(30,4);Writeln('ΕΠΙΛΟΓΗ ΕΞΟΔΟΥ');
        GotoXY(31,6);Writeln('1. ΣΤΟ ΔΙΣΚΟ');
        GotoXY(31,7);Writeln('2. ΣΤΗΝ ΟΘΟΝΗ');
        GotoXY(31,8);Writeln('3. ΣΤΟΝ ΕΚΤΥΠΩΤΗ');
        GotoXY(31,9);Writeln('4. ΕΞΟΔΟΣ');
        GotoXY(28,12);Writeln('ΔΩΣΤΕ ΤΗΝ ΕΠΙΛΟΓΗ ΣΑΣ : ');
        Readln(epil);
End; { Menu }
Procedure Selection;
Begin
        Case epil of
                1 : FileName := 'RAN_DATA.DAT'
                2 : Filename := 'CON';
                3 : FileName := 'PRN';
        End; { Case }
End; { Selection }

Begin { Κύριο Πρόγραμμα }
        Repeat
                Menu;
                flag := True;
                While (epil>0) and (epil<4) and (flag=True) do
                Begin
                        Selection;
                        Assign(datafile, FileName);
                        Rewrite(datafile);
                        for i:= 1 to 10 do
                                Writeln(datafile, Random(50));
                        flag := False;
                End; { While }
        Until epil = 4;
End.
```

Οι διαδικασίες τυχαίας προσπέλασης μπορούν να χρησιμοποιηθούν με οποιοδήποτε αρχείο καθορισμένου τύπου. Τυχαία Προσπέλαση είναι η δυνατότητα ανάγνωσης ή εγγραφής στοιχείων οπουδήποτε μέσα στο αρχείο. Οι δύο βασικές διαδικασίες για την τυχαία προσπέλαση είναι η Seek και η FilePos. Το πρώτο στοιχείο του αρχείου είναι το μηδέν, το δεύτερο είναι το ένα κ.ο.κ.

Η σύνταξη της διαδικασίας Seek είναι η ακόλουθη :

Seek*(Μεταβλητή_Αρχείου, Θέση_Δείκτη_Αρχείου)*

Δείτε το παρακάτω παράδειγμα :

```
Type
        phone_rec = RECORD
                name, notes : String;
                number : LongInt;
                end; { record }
Var
        phone_list : File of phone_rec;
        rec_10, rec_11, rec_15, rec_25 : phone_rec;
Begin
        Assign(phone_list, 'phone.dat');
        Reset(phone_list);
        Seek(phone_list, 9);
        Read(phone_list, rec_10, rec_11);
        Seek(phone_list, 14);
        Read(phone_list, rec_15);
        Seek(phone_list, 24);
        Read(phone_list, rec_25);
End.
```

Βλέπουμε στο παραπάνω πρόγραμμα ότι μπορούμε να έχουμε άμεση πρόσβαση σ' όποιον αριθμό εγγραφής θέλουμε.

Δείτε και το παρακάτω πρόγραμμα όπου δίνουμε εμείς τον αριθμό της εγγραφής για να κάνουμε καταχώρηση στοιχείων :

```
Type
        phone_rec = Record
                name, notes : String;
                number : LongInt;
                end; { record }
Var
        phone_list : File of phone_rec;
        temp_rec : phone_rec;
        N : LongInt;
Begin
        Assign(phone_list, 'phone.dat');
        Reset(phone_list);
        Write('Δώστε τον αριθμό της εγγραφής : ');
        Readln(N);
        Write('Δώστε το όνομα : ');
        Readln(temp_rec.name);
        Write('Δώστε τον αριθμό τηλεφώνου : ');
        Readln(temp_rec.number);
        Seek(phone_list, N);
        Write(phone_list, temp_rec);
End.
```

Η προσάρτηση δεδομένων στο τέλος του αρχείου γίνεται ως εξής :

```
Seek(f, FilePos(f));
```

Η προσπέλαση οποιουδήποτε στοιχείου του πίνακα γίνεται ως εξής :

Μεταβλητή_Δείκτη_ΠίνακαÙ[Δείκτης]

Δείτε το παρακάτω παράδειγμα :

```
Const
        max_elements = 65520 div SizeOf(Real);
Type
        some_reals = Array[1..max_elements] of Real;
Var
        rptr : ∧some_reals;
        i : Byte;
        array_size : Word;
Begin
        array_size := 100;
        GetMem(rptr, array_size * SizeOf(Real));
        For i:=1 to array_size do
        Begin
                rptr∧[i] := i;
                Writeln(rptr∧[i]);
        End; { For }
        FreeMem(rptr, array_size * SizeOf(Real));
End.
```

Η διαδικασία FreeMem αποδεσμεύει τα τμήματα μνήμης που κατακρατήθηκαν από τη GetMem.