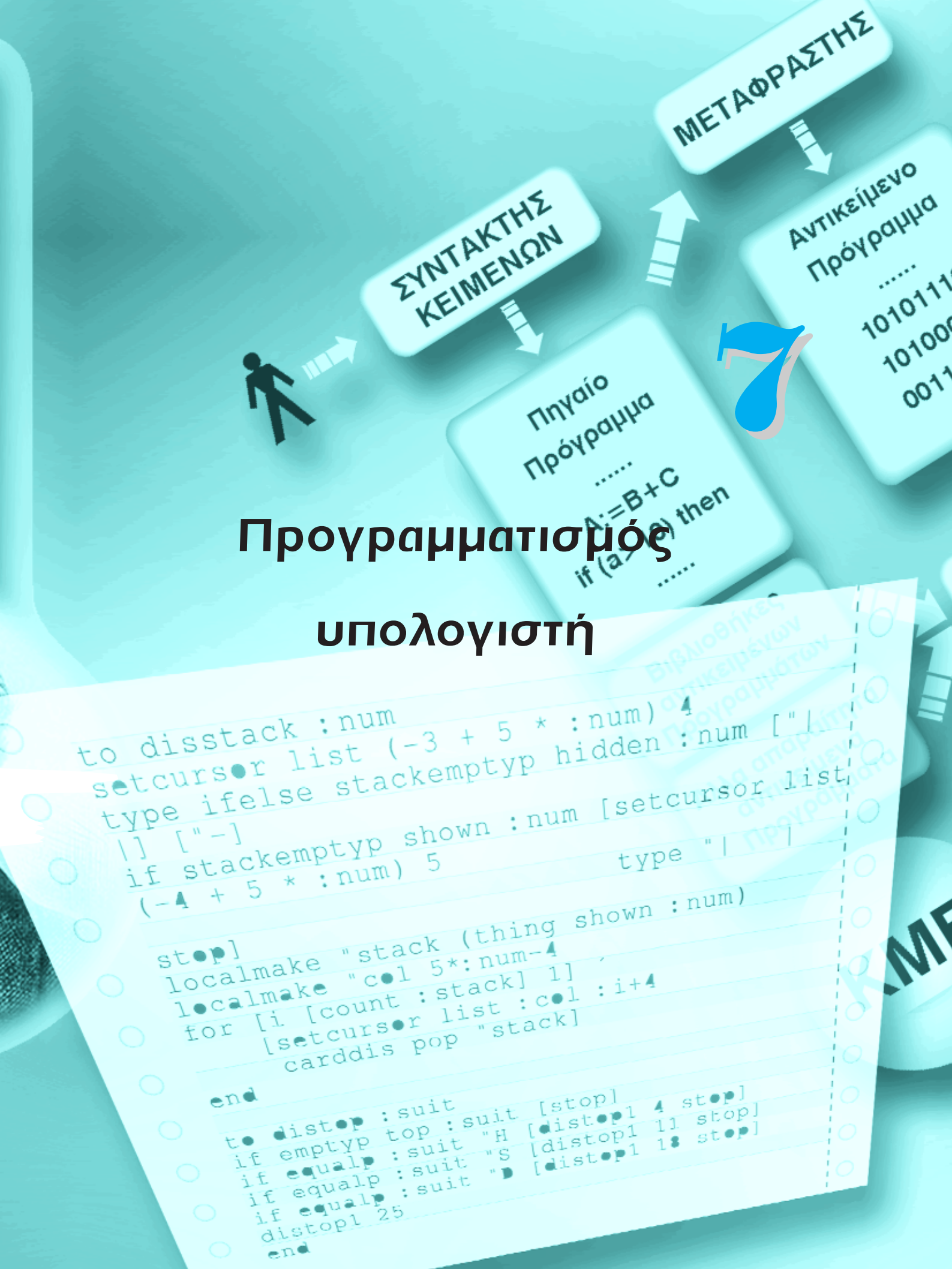

Προγραμματισμός υπολογιστή

7.1 Περίληψη κεφαλαίου

Κύριο διακριτικό γνώρισμα του υπολογιστή είναι η δυνατότητα προγραμματισμού του, με τη λειτουργία του να καθορίζεται από το πρόγραμμα που κάθε στιγμή εκτελεί. Ο προγραμματισμός, αποτελεί έναν από τους βασικούς τομείς της επιστήμης των υπολογιστών.

Τα πρώτα προγράμματα γράφονταν στη γλώσσα μηχανής του υπολογιστή.

Στη συνέχεια, για να διευκολυνθεί η διαδικασία του προγραμματισμού, σχεδιάστηκαν σταδιακά γλώσσες προγραμματισμού οι οποίες είναι πιο κοντά στον τρόπο έκφρασης τού ανθρώπου. Ένα πρόγραμμα, λοιπόν, μπορεί να είναι γραμμένο σε:

- Γλώσσα μηχανής.
- Συμβολική γλώσσα.
- Γλώσσα υψηλού επιπέδου.

Από τον πρώτο προγραμματιζόμενο υπολογιστή έως σήμερα, έχει επινοηθεί ένας πολύ μεγάλος αριθμός γλωσσών προγραμματισμού. Μερικές από τις πιο γνωστές και επιτυχημένες εμπορικά είναι η FORTRAN, η COBOL, η C και η SQL. Ανάλογα με το σκοπό για τον οποίο σχεδιάστηκαν, οι **γλώσσες προγραμματισμού** διακρίνονται σε:

- Ειδικού σκοπού, επειδή σχεδιάστηκαν για να καλύπτουν τις απαιτήσεις ενός συγκεκριμένου φάσματος προβλημάτων.
- Γενικού σκοπού, αυτές που σχεδιάστηκαν για να καλύπτουν ένα γενικότερο φάσμα προβλημάτων.

Τα προγράμματα γράφονται είτε σε συμβολικές γλώσσες είτε σε γλώσσες υψηλού επιπέδου. Τη μετατροπή τους σε γλώσσα μηχανής αναλαμβάνουν κατά περίπτωση, οι συμβολομεταφραστές, οι μεταγλωττιστές και οι διερμηνευτές.

Η συγγραφή των προγραμμάτων έχει εξελιχθεί σε μια δομημένη διαδικασία που διέπεται από τις αρχές **δομημένου** προγραμματισμού, σύμφωνα με τις οποίες ένα πρόγραμμα κατασκευάζεται από συνεργαζόμενα δομικά στοιχεία, τα οποία υλοποιούνται με βάση καθορισμένους τύπους προγραμματιστικών δομών. Οι δομές αυτές είναι:

- Η διαδοχή.
- Η επιλογή.
- Η επανάληψη.

Ανάλογα με το πώς προσεγγίζεται ο προγραμματισμός του υπολογιστή, έχουν δημιουργηθεί διάφορα πρότυπα προγραμματισμού. Αυτά είναι:

- Ο διαδικαστικός προγραμματισμός.
- Ο αντικειμενοστρεφής προγραμματισμός.
- Ο λογικός προγραμματισμός.
- Ο συναρτησιακός προγραμματισμός.

Η κατασκευή ενός προγράμματος ακολουθεί ορισμένα βήματα. Τα πιο βασικά είναι:

α) Προσδιορισμός των απαιτήσεων του προβλήματος.
β) Ανάλυση του προβλήματος.
γ) Σχεδιασμός αλγόριθμου για την επίλυση του προβλήματος.
δ) Υλοποίηση του αλγόριθμου.

ε) Έλεγχος και επαλήθευση του τελικού προγράμματος.
στ) Συντήρηση και ενημέρωση του προγράμματος.
ζ) Τεκμηρίωση.

Για να διευκολυνθεί η διαδικασία ανάπτυξης προγραμμάτων, έχουν αναπτυχθεί προγραμματιστικά περιβάλλοντα, τα οποία στην απλή τους μορφή περιλαμβάνουν απλώς ένα συντάκτη κειμένων και τα κατάλληλα μεταφραστικά προγράμματα, ενώ την πιο σύνθετη μορφή τους αποτελούν περιβάλλοντα όπως τα εργαλεία CASE ή τα περιβάλλοντα οπτικού προγραμματισμού.

7.2 Ειδικά θέματα

7.2.1 Εισαγωγή στη γλώσσα προγραμματισμού Pascal

Η γλώσσα προγραμματισμού PASCAL είναι γλώσσα υψηλού επιπέδου τρίτης γενιάς. Σχεδιάστηκε το 1971 στη Ζυρίχη από τον Ελβετό Niklaus Wirth , ο οποίος συμμετείχε στην ομάδα που σχεδίασε την ALGOL-68. Η αποδοχή της Pascal υπήρξε ταχύτατη και σήμερα θεωρείται ως η κατεξοχήν γλώσσα για τη συστηματική διδασκαλία των εννοιών του προγραμματισμού.

Στη συνέχεια θα δώσουμε μια **πολύ συνοπτική** και απλοποιημένη, σε ένα βαθμό, περιγραφή της γλώσσας προγραμματισμού Pascal, μια και η πλήρης διδασκαλία μιας γλώσσας προγραμματισμού δεν αποτελεί αντικείμενο αυτού του μαθήματος. Η Pascal θεωρείται μια πολύ καλή γλώσσα για την παρουσίαση των εννοιών του προγραμματισμού.

Ένα απλό πρόγραμμα σε Pascal είναι το εξής:

```
Program first(input, output);
{τυπώνει Καλημέρα}
begin
  writeln('Καλημέρα');
end {first}.
```

Το παραπάνω πρόγραμμα, αν εκτελεστεί, θα τυπώσει στην έξοδο (output) (το ρόλο της οποίας παίζει τις περισσότερες φορές η οθόνη) τη λέξη «Καλημέρα». Οι λέξεις program, begin και end είναι δεσμευμένες λέξεις της γλώσσας και βρίσκονται σε κάθε πρόγραμμα της Pascal. Ειδικότερα:

- Τα **begin** και **end** περικλείουν σειρά εντολών τις οποίες καθιστούν συντακτικά ισοδύναμες με μία εντολή. Έτσι, όπου στη συνέχεια αναφερόμαστε σε «μία εντολή», θα εννοούμε είτε μία απλή εντολή είτε σειρά εντολών που περικλείονται από τις λέξεις begin και end.
- Ό,τι περιλαμβάνεται μεταξύ των αγκυλών **{ }** θεωρείται σχόλιο και δεν εκτελείται. Τα σχόλια χρησιμοποιούνται για την εσωτερική τεκμηρίωση του προγράμματος.
- Η λέξη first είναι το όνομα του προγράμματος, δεν είναι δεσμευμένη λέξη, αλλά τη δίνει ο προγραμματιστής και συνήθως είναι τέτοια, που να θυμίζει το τι κάνει το πρόγραμμα.

Γενικά η δομή ενός προγράμματος Pascal είναι:

```
Program αναγνωριστικό (εξωτερικά αρχεία)
  Μέρος ορισμών και δηλώσεων
  Μέρος εντολών.
```

Τμήμα δηλώσεων - τύποι δεδομένων

Στο τμήμα δηλώσεων ενός προγράμματος, ορίζονται οι μεταβλητές και οι σταθερές που θα χρησιμοποιηθούν στο πρόγραμμα. Η Pascal ανήκει στην κατηγορία εκείνη των γλωσσών που απαιτούν οι μεταβλητές να δηλώνονται και ως προς το όνομα και ως προς τον τύπο, πριν να χρησιμοποιηθούν.

Το τμήμα δηλώσεων μπορεί να έχει τη μορφή:

```
Var
   Όνομα_μεταβλητής(-ών): τύπος;
   Όνομα_μεταβλητής(-ών): τύπος;
....
```

Οι κυριότεροι τύποι δεδομένων που συναντώνται σε προγράμματα Pascal είναι:

- **integer** : ακέραιος αριθμός
- **real** : πραγματικός αριθμός
- **char** : χαρακτήρας
- **string** : σειρά από χαρακτήρες ή αλλιώς συμβολοσειρά
- **boolean** : χρησιμοποιείται για ορισμό δεδομένων, τα οποία παίρνουν την τιμή true (αληθές) ή false (ψευδές)
- **text** : ακολουθιακό αρχείο χαρακτήρων

Για παράδειγμα στο παρακάτω πρόγραμμα:

```
program booleanexample (output)
const
  a=5;
  b=3;
var
  c:boolean;
begin
  c:=(a<b);
  write(c)
end.
```

ορίζουμε δύο σταθερές a και b με τιμές 5, 3 αντίστοιχα και μία μεταβλητή τύπου boolean, την c. Στη συνέχεια εκχωρούμε στη c τη λογική τιμή της συνθήκης a<b και δίνουμε εντολή να τυπωθεί αυτή στην οθόνη. Τελικά θα τυπωθεί false, αφού η συνθήκη a<b είναι ψευδής.

Βασικές εντολές της Pascal

Εντολή εκχώρησης

Η σύνταξη μιας εντολής εκχώρησης έχει τη μορφή:

```
Μεταβλητή := παράσταση;
```

και έχει ως αποτέλεσμα την αποθήκευση της τιμής της παράστασης στη θέση που προσδιορίζει η μεταβλητή του αριστερού μέρους της εντολής. Να σημειωθεί ότι η παράσταση στο δεξί μέρος μπορεί να περιλαμβάνει μεταβλητές, στις οποίες πρέπει όμως να έχει εκχωρηθεί προηγουμένως τιμή.

Παραδείγματα:

```
platos := 45;    {όπου platos δηλώθηκε ως integer ή real}
mikos := 32.4;   {όπου mikos δηλώθηκε ως real}
embadon := platos * mikos;   {όπου embadon δηλώθηκε ως real}
arxikoGramma := 'A';   {όπου arxikoGramma δηλώθηκε ως char}
found := true;  {όπου found δηλώθηκε ως boolean}
salute := 'Kalispera';  {όπου salute δηλώθηκε ως string}
```

Εντολές εισόδου - εξόδου

WRITE

Τυπώνει στην έξοδο, η οποία είναι συνήθως η οθόνη, στοιχεία.

Παραδείγματα:

```
Write('hello');
```

Τυπώνει hello στην οθόνη.

```
Write(file1, 'Johnny');
```

Προσθέτει τη λέξη Johnny στο τέλος του αρχείου file1.

```
Write(output,c);
```

Τυπώνει στην οθόνη την τρέχουσα τιμή της μεταβλητής c.

```
Write(file1,b);
```

Προσθέτει την τρέχουσα τιμή της μεταβλητής c στο τέλος του αρχείου file1.

READ

Χρησιμοποιείται για να διαβάσουμε από την είσοδο -πληκτρολόγιο ή αρχείο- δεδομένα. Τα δεδομένα καταχωρούνται σε αντίστοιχες μεταβλητές.

Παραδείγματα:

```
Read(input,c);
```

Διαβάζει από την οθόνη την τιμή που έχει πληκτρολογηθεί και την εκχωρεί στη μεταβλητή c.

```
Read(file1,c);
```

Εφόσον δεν έχει διαβαστεί όλο το αρχείο file1, εκχωρεί το επόμενο στοιχείο του στη μεταβλητή c.

Εντολές ελέγχου

IF

Η σύνταξή της είναι της μορφής:

```
If  λογική συνθήκη then
  Εντολή
else
  Εντολή;
```

και έχει την εξής έννοια:

εάν (if) η λογική συνθήκη έχει την τιμή true, εκτέλεσε την εντολή που ακο-

λουθεί. Αλλιώς (else) εκτέλεσε την εντολή που ακολουθεί μετά την ειδική λέξη else. Το κομμάτι με το else είναι προαιρετικό και, αν απουσιάζει, προχωρεί στις επόμενες εντολές του προγράμματος.

Παραδείγματα:

```
if a>0 then
  sum := sum + a;
```

Προσθέτει στη μεταβλητή sum (που έχει τον ρόλο **αθροιστή**) την τιμή της μεταβλητής a, εάν αυτή είναι θετική.

```
read(a);
if a>0 then
  thetikoi := thetikoi + 1
else
  arnitikoi := arnitikoi + 1
end
```

Μετράει (οι μεταβλητές `thetikoi και arnitikoi` έχουν τον ρόλο **μετρητών**) το πλήθος των θετικών και αρνητικών ακεραίων χρησιμοποιώντας τις μεταβλητές thetikoi και arnitikoi αντίστοιχα.

CASE

Η σύνταξή της είναι της μορφής:

```
case δείκτης (μεταβλητή ή σταθερά ή παράσταση) of
τιμή δείκτη : εντολή/ές
      •
      •
      •
end
```

και έχει την εξής έννοια:
Εκτέλεσε την εντολή/ές που αντιστοιχούν στη σωστή τιμή δείκτη.

Παράδειγμα:

```
read(monthnum);
case monthnum of
      1: write('January');
      2: write('February');
      3: write('March');
      4: write('April');
      5: write('May');
      6: write('June');
      7: write('July');
      8: write('August');
      9: write('September');
      10: write('October');
      11: write('November');
      12: write('December')
end;
```

Διαβάζει έναν ακέραιο αριθμό (monthnum) και τυπώνει το μήνα στον οποίο αυτός αντιστοιχεί.

Εντολές επανάληψης

WHILE

Η σύνταξή της είναι της μορφής:

```
While λογική συνθήκη do
  Εντολή;
```

και έχει την εξής λειτουργία:
όσο (while) η λογική συνθήκη έχει την τιμή true, εκτέλεσε την εντολή που ακολουθεί. Όταν η λογική συνθήκη γίνει false, προχώρησε στην εντολή που ακολουθεί την while.

Παραδείγματα:

```
read(int);
while int > 0 do
  read(int);
```

Διαβάζει τα δεδομένα, μέχρι να βρει τον πρώτο αρνητικό ακέραιο.

```
sum:=0; read(input, a);
while a<>0 do
begin
  sum:=sum+1;
  read(a)
end
```

Μετράει μέσω της μεταβλητής sum το πλήθος των ακεραίων, μέχρι να βρει τον πρώτο μηδενικό ακέραιο.

Προσοχή: Πρέπει, υποχρεωτικά, η εντολή που ακολουθεί την ειδική λέξη do να τροποποιεί, σε κάποια επανάληψη, τη λογική συνθήκη σε False, αλλιώς το πρόγραμμα θα πέσει σε ατέρμονα ανακύκλωση (**endless loop**). Π.χ.

```
a:=5;
while a>0 do
  read(c);
```

Επειδή η λογική συνθήκη a>0 είναι πάντα true, θα εκτελείται συνεχώς η εντολή read(c).

REPEAT

Η σύνταξή της είναι της μορφής:

repeat Εντολή/εντολές
until λογική συνθήκη

και έχει την εξής έννοια:
Εκτέλεσε την εντολή/ές που ακολουθεί/ούν τη λέξη repeat. Στη συνέχεια έλεγξε τη λογική συνθήκη. Αν αυτή είναι true, επανάλαβε την εκτέλεση των εντολών, αλλιώς προχώρησε παρακάτω στο πρόγραμμα.

Παράδειγμα:

```
repeat
  read(int);
  writeln('Ο αριθμός που δόθηκε είναι', int);
```

```
  writeln('Θέλετε να ξαναδώσετε αριθμό; Πατήστε Y ή N');
  read(answer)
until answer = 'N'
```

Διαβάζει τον αριθμό που πληκτρολογεί ο χρήστης και τον τυπώνει μέχρι ο χρήστης να πληκτρολογήσει Ν.

Η διαφορά της repeat με την while είναι ότι πρώτα εκτελεί τις εντολές και μετά εξετάζει τη συνθήκη, με αποτέλεσμα να τις εκτελεί τουλάχιστον μία φορά. Επίσης η repeat τερματίζεται όταν η λογική συνθήκη πάρει την τιμή true, ενώ η while τερματίζεται όταν πάρει την τιμή false. Και οι δύο παρατηρούμε ότι βασίζονται σε συνθήκη.

FOR
Η σύνταξή της είναι της μορφής:

```
For μεταβλητή ελέγχου := αρχική τιμή to/downto τελική τιμή do
  Εντολή
```

και έχει την εξής έννοια:
Εκχώρησε στη μεταβλητή ελέγχου την αρχική τιμή. Έπειτα εκτέλεσε την εντολή που ακολουθεί την ειδική λέξη do, αν η τιμή της μεταβλητής ελέγχου είναι μικρότερη ή ίση στην περίπτωση του to ή μεγαλύτερη ή ίση στην περίπτωση του downto της τελικής τιμής. Στη συνέχεια αύξησε (to) ή μείωσε (downto) κατά το επόμενο στοιχείο την τιμή της μεταβλητής ελέγχου και επανάλαβε το προηγούμενο βήμα. Βασίζεται σε μετρητή.

Παραδείγματα:

```
for I:=1 to 10 do
  write('*')
```

Τυπώνει στην οθόνη 10 αστερίσκους και είναι ισοδύναμο με το

```
for I:=10 downto 1 do
   write('*')
```

```
sum:=0;
for i:=1 to 20 do
  sum := sum + i;
```

Εκχωρεί στη μεταβλητή sum το άθροισμα των 20 πρώτων ακεραίων.

```
sum := 0;
for i:=1 to 15 do
begin
  read(num);
  sum:= sum + num
end
```

Διαβάζει 15 αριθμούς και τους αθροίζει στη μεταβλητή sum.

Οι εντολές while, repeat και for καλούνται επαναληπτικές.

Υποπρογράμματα

Η Pascal υποστηρίζει τη χρήση **υποπρογραμμάτων,** δηλαδή ενός συνόλου εντολών το οποίο γράφεται μόνο μια φορά στο τμήμα δηλώσεων και στη συνέχεια μπορεί να κληθεί μέσω του ονόματός του και των παραμέτρων του, όσες φορές είναι απαραίτητο, από το κυρίως πρόγραμμα ή από άλλα υποπρογράμματα.
Υπάρχουν δύο ειδών υποπρογράμματα: οι **συναρτήσεις** (functions) και οι **διαδικασίες** (procedures). Στη συνέχεια παρατίθενται δύο παραδείγματα χρήσης συνάρτησης και διαδικασίας για απόκτηση μιας γενικής ιδέας, δεδομένου ότι η περαιτέρω ανάλυση του θέματος ξεφεύγει από τα πλαίσια αυτού του παραρτήματος:

Πρόγραμμα με χρήση συνάρτησης

```
Program functionexample (input, output);
{Υπολογίζει και τυπώνει τον μέσο όρο τριών ακεραίων που
δίνονται από τον χρήστη}
var
  int1, int2, int3: integer;
  avnum: real;

function average(a, b, c: integer): real;
{υπολογίζει τον μέσο όρο τριών ακεραίων}
var
  sum : integer;
begin
  sum := a + b + c;
  average := sum div 3
end {τέλος συνάρτησης average};

begin { κύριο πρόγραμμα }
  writeln (output, «Δώσε τρεις ακεραίους»);
  readln(input, int1);
  readln(input, int2);
  readln(input, int3);
  avnum := average (int1, int2, int3);
  writeln(output, 'Ο μέσος όρος των ακεραίων ', int1, '
  ', int2, 'και', int3, 'είναι', avnum)
end{functionexample}.
```

Παράδειγμα διαδικασίας

```
Procedure swap(var a, b: integer);
{Ανταλλάσσει τις τιμές δύο ακέραιων μεταβλητών}
var
  temp: integer;
begin
```

```
  temp := a;
  a := b;
  b := temp
end{swap}
```

στο σημείο ενός προγράμματος που θέλουμε να καλέσουμε τη διαδικασία, γράφουμε:

```
swap(x, y);
```

Πίνακες

Οι απλές μεταβλητές καταλαμβάνουν ένα κελί στη μνήμη του υπολογιστή και μπορούν να αποθηκεύσουν μία τιμή κάθε χρονική στιγμή. Υπάρχουν περιπτώσεις στις οποίες είναι προτιμότερο να αποθηκεύουμε συσχετιζόμενα δεδομένα όχι σε ξεχωριστές μεταβλητές αλλά ομαδοποιημένα. Οι πίνακες αποτελούν ένα τέτοιο μηχανισμό. Σε μια μεταβλητή τύπου πίνακα αποθηκεύουμε ομοειδή δεδομένα, π.χ. δεδομένα ακέραιου τύπου.
Υπάρχουν πίνακες μονοδιάστατοι και πίνακες πολλών διαστάσεων. Η δήλωση ενός μονοδιάστατου πίνακα είναι:

```
Αναγνωριστικό πίνακα : array[κάτω όριο πίνακα.. πάνω όριο πίνακα] of τύπος στοιχείων πίνακα;
```

Για παράδειγμα η δήλωση

```
Sales : array [1..30] of real;
```

ορίζει ένα πίνακα 30 θέσεων που περιέχει στοιχεία τύπου real (πραγματικοί αριθμοί) και ονομάζεται sales. Στις 30 θέσεις του πίνακα αναφερόμαστε μέσω δείκτη. Η μεταβλητή sales[4] αναφέρεται στο 4ο στοιχείο του πίνακα sales. Έτσι μπορούμε να γράφουμε:

```
Sales[5] := 123.4;
```

Παράδειγμα
Το παρακάτω πρόγραμμα διαβάζει ακέραιες βαθμολογίες 30 μαθητών, τις τυπώνει και υπολογίζει τον μέσο όρο τους.

```
Program vathmologia (input, output);
const
  plithos = 30; {αριθμός μαθητών}
var
  vathmoi : array [1..plithos] of integer;
  counter, sum :integer;
  average : real;
begin {κύριο πρόγραμμα}
  sum:=0;
  for counter := 1 to plithos do
```

```
  begin
    read( vathmoi[counter]);
    sum := sum + vathmoi[counter]
  end;
  For counter := 1 to plithos do
    write (output, 'Η βαθμολογία του μαθητή', counter,
    'είναι', vathmoi[counter]);
    average := sum / plithos;
    write(output, ' Ο μέσος όρος της βαθμολογίας είναι',
    average)
end.
```

7.3 Ασκήσεις - Δραστηριότητες

A ■□□ 1. Διερευνήστε το προγραμματιστικό περιβάλλον που είναι διαθέσιμο στο εργαστήριό σας.
- Πώς ονομάζεται;
- Ποια ή ποιες γλώσσες υποστηρίζει;
- Να γράψετε ένα πρόγραμμα που να τυπώνει «Καλημέρα».
- Να καταγράψετε τα βήματα και τις εντολές που απαιτήθηκαν στο προγραμματιστικό περιβάλλον του εργαστηρίου σας για τον πιο πάνω σκοπό.

A ■□□ 2. Να γραφεί ένα πρόγραμμα που να διαβάζει τους βαθμούς θερμοκρασίας σε Fahrenheit και να τυπώνει τους αντίστοιχους σε Celsius.

ΥΠΟΔΕΙΞΗ

Ο σχετικός τύπος είναι $C = \frac{5}{9}(F-32)$*, όπου F είναι οι βαθμοί Fahrenheit και C οι βαθμοί Celsius.*

A ■□□ 3. Να γραφεί ένα πρόγραμμα που να διαβάζει τρεις αριθμούς και να απαντά αν μπορούν να αποτελέσουν γωνίες τριγώνου. (Μονάδα μέτρησης των γωνιών η μοίρα).

A ■□□ 4. Να γραφεί πρόγραμμα που να υπολογίζει το εμβαδό ενός τριγώνου από τις πλευρές του.

A ■□□ 5. Να γραφεί πρόγραμμα που να διαβάζει δύο ακέραιους αριθμούς και να τυπώνει το ποιος είναι ο μεγαλύτερος και ποιος ο μικρότερος.

A ■□□ 6. Να γραφεί πρόγραμμα που να διαβάζει έναν ακέραιο αριθμό και να βρίσκει αν είναι άρτιος ή περιττός.

A ■□□ 7. Να γραφεί πρόγραμμα που να διαβάζει έναν ακέραιο αριθμό Ν, και να υπολογίζει το άθροισμα των περιττών μεταξύ 1 και Ν.

ΥΠΟΔΕΙΞΗ

Η διαδικασία αυτή χρησιμοποιεί την έννοια του αθροιστή.

A ■□□ 8. Να γραφεί πρόγραμμα που να διαβάζει 20 αριθμούς και να τυπώσει πόσοι από αυτούς είναι θετικοί και πόσοι αρνητικοί. Θεωρήστε το 0 θετικό.

ΥΠΟΔΕΙΞΗ

Η διαδικασία αυτή χρησιμοποιεί την έννοια του μετρητή

9. Να γραφεί πρόγραμμα που να διαβάζει έναν ακέραιο αριθμό Ν και να υπολογίζει το άθροισμα 1+2+3+...+Ν. A

10. Να γραφεί πρόγραμμα που να διαβάζει έναν ακέραιο αριθμό Ν και να υπολογίζει το γινόμενο 1*2*3...*Ν. A

11. Να γραφεί πρόγραμμα το οποίο να εναλλάσσει τα περιεχόμενα δύο μεταβλητών. A

ΥΠΟΔΕΙΞΗ

Η διαδικασία αυτή καλείται διαδικασία ανταλλαγής (swap).

12. Να γραφεί ένα πρόγραμμα που να διαβάζει το ποσό ενός κεφαλαίου και το ετήσιο επιτόκιο και να υπολογίζει το τελικό κεφάλαιο μετά από 10 χρόνια με ετήσιο ανατοκισμό. A

ΥΠΟΔΕΙΞΗ

Το κεφάλαιο που προκύπτει υπολογίζεται από τον τύπο:

$ΤΚ = Κ * (1+ Ε/100)^{Ν}$, όπου ΤΚ το τελικό κεφάλαιο, Ε το επιτόκιο και Ν ο αριθμός περιόδων ανατοκισμού.

13. Σε μια εταιρεία πωλήσεων, στο τέλος κάθε μήνα ο κάθε πωλητής εκτός από το μισθό του παίρνει και ένα bonus που είναι ανάλογο των πωλήσεων που έκανε. Να γράψετε ένα πρόγραμμα που να δέχεται στην είσοδο το όνομα και το ύψος των πωλήσεων ενός πωλητή και να τυπώνει στην έξοδο το όνομα και το αντίστοιχο bonus. Ο τρόπος υπολογισμού του bonus είναι: A

Ύψος πωλήσεων	Bonus
0 - 2.000.000	0
2.000.001 - 5.000.000	2%
5.000.001 - 10.000.000	4%
10.000.001 -	5%

Το ποσοστό του bonus υπολογίζεται χωριστά για κάθε τμήμα της κλίμακας.

ΥΠΟΔΕΙΞΗ

Παράδειγμα: Για ύψος πωλήσεων 6.000.000 το bonus είναι:

*3.000.000*0.02 + 1.000.000*0.04 = 100.000 Δρχ.*

14. Να γράψετε ένα πρόγραμμα που να διαβάζει έναν αριθμό δευτερολέπτων και να τυπώνει τον αντίστοιχο χρόνο σε ώρες, λεπτά και δευτερόλεπτα στη μορφή ΩΩ:ΛΛ:ΔΔ. Π.χ. 3:15:43. A

15. Να γράψετε ένα πρόγραμμα που να βρίσκει σε έναν κύκλο την περίμετρο και το εμβαδόν του, αν είναι γνωστή η ακτίνα του. A

16. Να γράψετε ένα πρόγραμμα που να βρίσκει το ΜΚΔ δύο ακεραίων θετικών αριθμών. A

Α 17. Να γράψετε ένα πρόγραμμα που να βρίσκει το ΕΚΠ δύο ακεραίων θετικών αριθμών.

Α 18. Να γράψετε ένα πρόγραμμα που να βρίσκει το ακέραιο πηλίκο και υπόλοιπο δύο ακεραίων αριθμών, μόνο με τη χρήση πρόσθεσης και αφαίρεσης.

Α 19. Μπορείτε να λύσετε το προηγούμενο πρόβλημα μόνο με τη χρήση πρόσθεσης;

Α 20. Σε ένα πείραμα Χημείας μετράμε τις τιμές ενός μεγέθους (π.χ. θερμοκρασίας). Έχουμε επαναλάβει το πείραμα είκοσι φορές και έχουμε καταγράψει τα αποτελέσματα της μέτρησης. Θέλουμε να κατασκευάσουμε ένα πρόγραμμα που να διαβάζει τις μετρήσεις και να τυπώνει το μέσο όρο και την τυπική απόκλιση των μετρήσεων.

ΥΠΟΔΕΙΞΗ

Η μέση τιμή: $\bar{x} = \frac{\sum x}{n}$

Τυπική απόκλιση: $\sigma = \sqrt{\frac{\sum (x - \bar{x})^2}{n}}$, *όπου x οι παρατηρήσεις και n το μέγεθος του δείγματος.*

Α 21. Να γραφεί ένα πρόγραμμα που να διαβάζει έναν ακέραιο από το 1 έως το 7 και να τυπώνει σε ποια ημέρα της εβδομάδας αυτός αντιστοιχεί. Το 1 αντιστοιχεί στη Δευτέρα.

Α 22. Κατασκευάστε πρόγραμμα που να εξετάζει αν ένας αριθμός είναι πρώτος.

ΥΠΟΔΕΙΞΗ

Εξετάζουμε αν διαιρείται με N, όπου $2 \le N \le \sqrt{M}$ *, και M ο υπό εξέταση αριθμός.*

Α 23. Να μετατρέψετε το προηγούμενο πρόγραμμα σε κατάλληλο υποπρόγραμμα και χρησιμοποιήστε το για να τυπώσετε τους πρώτους αριθμούς από 2 έως Ν. Το Ν δίνεται από το χρήστη.

Α 24. Η ημερομηνία του Πάσχα κάποιου έτους μπορεί να υπολογιστεί χρησιμοποιώντας τον τύπο του Gauss. Ο τύπος του Gauss είναι:
Η ημερομηνία του Πάσχα είναι 3 Απριλίου + Π, όπου
$\Pi = D + F$, $D = [19A + 16]_{30}$, $F = [2B + 4C + 6D]_7$, $A = [E]_{19}$, $B = [E]_4$, $C = [E]_7$
Ε: το έτος και $[\alpha]_\beta$ είναι το ακέραιο υπόλοιπο της διαίρεσης του α με το β.
Χρησιμοποιώντας τον τύπο του Gauss, κατασκευάστε ένα πρόγραμμα που να τυπώνει την ημερομηνία του Πάσχα για τα έτη από 2000 έως 2100.

Α 25. Να κατασκευάσετε πρόγραμμα που να δέχεται στην είσοδο τις μέσες θερμοκρασίες ενός αγνώστου αριθμού ημερών και να τυπώνει τη μέγιστη και την ελάχιστη καθώς και όλες τις ημερομηνίες που αυτές μετρήθηκαν. Το τέλος των δεδομένων δηλώνεται με το σύμβολο @ στη θέση της ημε-

ρομηνίας. Οι ημερομηνίες δίνονται στη μορφή «ΗΗ/ΜΜ/ΕΕΕΕ». Οι θερμοκρασίες έχουν μετρηθεί μέσα στο ίδιο έτος.

26. Κατασκευάστε πρόγραμμα που να μετατρέπει έναν ακέραιο θετικό αριθμό από το δεκαδικό σύστημα στα συστήματα δυαδικό, οκταδικό και δεκαεξαδικό. A

27. Ένας ταμίας θέλει να πληρώνει τους μισθούς σε μετρητά χρησιμοποιώντας τον ελάχιστο δυνατό αριθμό νομισμάτων. Κατασκευάστε ένα πρόγραμμα για να τον βοηθήσετε. Θεωρήστε ότι οι μισθοί είναι στρογγυλοποιημένοι στις 100 δραχμές A

28. Τροποποιήστε το προηγούμενο πρόγραμμα, ώστε να μπορεί, με βάση τους μισθούς ενός μήνα, να υπολογίζει τον ακριβή αριθμό νομισμάτων που θα πρέπει να προμηθεύεται ο ταμίας από την τράπεζα για να πληρώσει όλους τους μισθούς, χρησιμοποιώντας τον ελάχιστο δυνατό αριθμό νομισμάτων. A

29. Απλό παιχνίδι: Ο υπολογιστής «σκέπτεται» έναν αριθμό από το 1 έως το 1000. Στη συνέχεια προσπαθούμε να μαντέψουμε τον αριθμό με διαδοχικές δοκιμές. Σε κάθε μας προσπάθεια, ο υπολογιστής απαντά με ένα από τα: A
    - Μπράβο το βρήκες!
    - Ο αριθμός που έδωσες είναι πιο μικρός.
    - Ο αριθμός που έδωσες είναι πιο μεγάλος.

    α) Γράψτε το κατάλληλο πρόγραμμα.

    β) Μπορεί κάποιος τυχαία να βρει τον αριθμό στη πρώτη προσπάθεια, όμως είναι πιθανό να χρειαστεί και 1000 προσπάθειες. Μπορείτε να αναπτύξετε μια στρατηγική που να εγγυάται, στη χειρότερη περίπτωση, το μικρότερο αριθμό προσπαθειών, από οποιαδήποτε άλλη; Ποιος είναι αυτός ο αριθμός;

30. Κατασκευάστε ένα πρόγραμμα που να δέχεται στην είσοδο τον αριθμητή και τον παρανομαστή ενός κλάσματος και να τυπώνει το κλάσμα απλοποιημένο. A

31. Να κατασκευάσετε ένα πρόγραμμα που να δέχεται τους βαθμούς ενός μαθητή, προφορικούς και γραπτούς καθώς και τον αριθμό των απουσιών του και να τυπώνει πληροφορίες σχετικές με το αν προάγεται ή όχι. A

    ΥΠΟΔΕΙΞΗ

    Για λόγους απλότητας θεωρούμε ότι κάποιος προάγεται αν οι απουσίες του είναι μέχρι κάποιο όριο, ο τελικός βαθμός είναι ο μέσος όρος μιας προφορικής και μιας γραπτής βαθμολογίας. Προάγεται αν ο μέσος όρος είναι από 10 και πάνω, ενώ άριστα θεωρείται από 18 και πάνω.

32. Μετατρέψτε το πρόγραμμα της προηγούμενης άσκησης σε κατάλληλο υποπρόγραμμα και χρησιμοποιήστε το για να δώσετε τα αποτελέσματα μιας τάξης. Στο τέλος δώστε χρήσιμα στατιστικά στοιχεία. Π.χ. μέσο όρο, ποσοστό αυτών που αρίστευσαν, ποσοστό αυτών που προήχθησαν, κλπ. A

Α ■■□ 33. Θέλουμε να κατασκευάσουμε ένα πρόγραμμα που να εξομοιώνει την οριζόντια βολή από κάποιο ύψος. Η γραφική παράσταση της βολής δεν είναι απαραίτητη. Στην είσοδο θα δέχεται τις διάφορες παραμέτρους, οι οποίες καθορίζουν τη βολή (π.χ. το ύψος από το οποίο γίνεται η βολή) και να τυπώνει την απόστασή της. Πειραματιστείτε δίνοντας διάφορες τιμές εισόδου και γράψτε τις παρατηρήσεις σας. Τι θα συμβεί αν η βολή γίνεται στη σελήνη;

ΥΠΟΔΕΙΞΗ

Να βρείτε πρώτα την εξίσωση τροχιάς.

Δ ■■□ 34. Αναζητήστε και καταγράψτε από την ύλη των άλλων μαθημάτων θέματα, στα οποία ένα πρόγραμμα θα μπορούσε να βοηθήσει στην κατανόησή τους. Επιλέξτε ένα από αυτά και κατασκευάστε το αντίστοιχο πρόγραμμα.

7.4 Ερωτήσεις Αυτοαξιολόγησης

1. Έχω κατανοήσει τη διαφορά ανάμεσα στη γλώσσα μηχανής, στις συμβολογλώσσες και στις γλώσσες υψηλού επιπέδου;
2. Μπορώ να απαριθμήσω μερικές γλώσσες προγραμματισμού;
3. Γνωρίζω το σκοπό και τη χρήση των συμβολομεταφραστών, των μεταγλωττιστών και των διερμηνευτών; Ποια η διαφορά τους;
4. Γνωρίζω τι είναι ο δομημένος προγραμματισμός;
5. Μπορώ να αναφέρω τα διάφορα πρότυπα προγραμματισμού;
6. Μπορώ να επιλύσω ένα απλό πρόβλημα ακολουθώντας τα βήματα της μεθόδου ανάπτυξης λογισμικού;
7. Έχω κατανοήσει τι είναι ένα προγραμματιστικό περιβάλλον;