ΚΕΦΑΛΑΙΟ 14
ΣΤΑΤΙΚΕΣ ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ

Σκοπός κεφαλαίου:

Να κατανοήσουμε την έννοια της Στατικής Δομής Δεδομένων.

Ειδικοί σκοποί:

- Να κατανοήσουμε την αναγκαιότητα του πίνακα ως Δομής Δεδομένων.
- Να κατανοήσουμε τους τρόπους ορισμού μονοδιάστατου και πολυδιάστατου πίνακα.
- Να κατανοήσουμε την αναγκαιότητα της εγγραφής ως Δομής Δεδομένων.
- Να κατανοήσουμε τον τρόπο ορισμού μιας εγγραφής.
- Να γνωρίσουμε ιεραρχικού τύπου εγγραφές.
- Να γνωρίσουμε την δυνατότητα ορισμού ενός πίνακα με στοιχεία και εγγραφές.

14.1. ΠΙΝΑΚΕΣ

Για να διαβάσουμε 1000 βαθμούς μαθητών και να τους τυπώσουμε κατά αντίστροφη σειρά θα γράφαμε ένα πρόγραμμα με την εξής μορφή:

```
Program antistrofi_seira;
var vath1,vath2,vath3,
   vath999,vath1000 : integer;
begin
readln(vath1);
readln(vath2);
...
readln(vath1000);
    writeln (vath1000);
    writeln (vath999);
.....
writeln (vath1;
end
```

```
program ant_sei;
uses wincrt;
var
deikti: 1..1000;
vath:array [1..1000] of integer;
begin
for deikti:=1 to 1000 do
        readln(vath[deikti]);
for deikti:=1000 downto 1 do
        writeln(vath[deikti]);
end.
```

Για το πρόγραμμα αυτό, όπως εμφανίζεται στην πρώτη στήλη του παραπάνω πίνακα, θα χρειαζόμαστε πάνω από 3000 γραμμές Το βασικό πρόβλημα είναι ότι χρειαζόμαστε 1000 διαφορετικές μεταβλητές παρόλο που είναι ίδιου τύπου κι έχουν το ίδιο όνομα, με μία μόνο διαφορετική κατάληξη για να ξεχωρίζουν. Θα ήταν πολύ πιο εύκολο αν μπορούσαμε να χρησιμοποιήσουμε μία μεταβλητή ως δείκτη για τη διαφορετική κατάληξη κάθε μετα-

βλητής. Στη δεύτερη στήλη του προηγούμενου πίνακα γράφουμε το αντίστοιχο τμήμα του κυρίου προγράμματος.

Το προηγούμενο τμήμα είναι σωστό πρόγραμμα αν ορίσουμε τη μεταβλητή vath ως ένα **μονοδιάστατο πίνακα.** Αυτό γίνεται με τη δήλωση **vath: array [1..1000] of integer.** Προσθέτουμε στη δεύτερη στήλη και τις απαραίτητες δηλώσεις για τον πίνακα των βαθμών.

Ένας μονοδιάστατος πίνακας είναι τύπος με **δόμηση** και αποτελείται από έναν προκαθορισμένο αριθμό στοιχείων τα οποία είναι -ανεξαιρέτως όλα- δεδομένα του ιδίου τύπου. Ο τύπος αυτός ονομάζεται βασικός τύπος του πίνακα. Οι επεξεργασίες σε πίνακα μπορεί να είναι αντικαταστάσεις για ολόκληρο τον πίνακα ή προσπέλαση σε μεμονωμένα στοιχεία του. Τα δεδομένα είναι καταχωρημένα στον πίνακα έτσι ώστε να ορίζεται μία διάταξη. Σύμφωνα με τη διάταξη αυτή, η προσπέλαση σε κάθε στοιχείο του πίνακα είναι άμεση. Η τιμή του δείκτη προσδιορίζει τη θέση του στοιχείου μέσα στον πίνακα. Ο δείκτης είναι ένας **τακτικός τύπος (ordinal type)**.

Με τη δήλωση ορισμού του πίνακα στην περιοχή δηλώσεων των μεταβλητών, καθορίζονται και τα δύο παραπάνω, δηλαδή και ο τύπος των στοιχείων του και ο τύπος του δείκτη. Ο **μονοδιάστατος πίνακας** είναι ο απλούστερος πίνακας, αυτός δηλαδή που έχει μία διάσταση (μια γραμμή ή μία στήλη). Ο τύπος του δείκτη προσδιορίζει για κάθε διάσταση τον αριθμό των στοιχείων του πίνακα. Οι απλοί τύποι είναι όλοι, εκτός από τον πραγματικό τύπο, τακτικοί τύποι. Στην περίπτωση του ακέραιου τύπου χρησιμοποιείται μία υποπεριοχή ακεραίων, γιατί ειδάλως ο πίνακας θα ήταν πολύ μεγάλος για να χω-ρέσει στη μνήμη.

Δήλωση μονοδιάστατου πίνακα

Ο τρόπος δήλωσης ενός πίνακα με όνομα Α είναι:

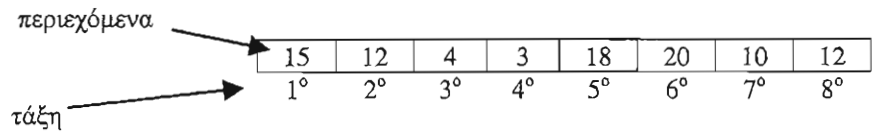
var

A: **array** [τύπος-δείκτη] **of** τύπος-στοιχείων-πίνακα

Παραδείγματα:

α.

Ο πίνακας που ακολουθεί είναι πίνακας μιας διάστασης (1 γραμμής και 8 στηλών) στον οποίο έχουν καταχωρηθεί 8 ακέραιοι αριθμοί.



Η αντίστοιχη δήλωση θα είναι:

var

pin: **array[1..8]** of **integer**;

όνομα πίνακα pin

τύπος-δείκτη **η υποπεριοχή 1..8**

(το σύνολο των τιμών του δείκτη είναι οι ακέραιοι 1 έως 8)

τύπος των στοιχείων του πίνακα ***integer***

β.

```
const
  max=640;
var
  bathmoi:  array[1..20]   of  integer;
  onoma:    array[1..30]   of  char;
  memory:   array[0..max]  of  integer;
```

γ.

Ο **αλφαριθμητικός τύπος** (**string**) μπορεί να θεωρηθεί ως μονοδιάστατος πίνακας χαρακτήρων του οποίου το πρώτο στοιχείο έχει τάξη 1.

```
var
  met:string[26];
  x:char;
.....
met:='abcdefghijklmnopqrstuvwxyz';
.....
x:=met[5]; {η μεταβλητή x έχει ως περιεχόμενο το 5ο στοιχείο της αλφαριθμητικής μεταβλητής που είναι ο χαρακτήρας e.
```

Παραδείγματα αναφοράς σε στοιχεία πινάκων:

α. pin[k]:=a+2

Το στοιχείο k τάξης του πίνακα αντικαθίσταται από το άθροισμα a+2 όπου a είναι μια μεταβλητή

β. X[1, 2]:=k+r

Το στοιχείο της πρώτης γραμμής και δεύτερης στήλης ενός πίνακα με το όνομα X αντικαθίσταται από το άθροισμα των μεταβλητών k και r.

Το στοιχείο ενός πίνακα μπορεί να είναι ένας άλλος πίνακας και τότε ο πίνακας λέγεται δύο διαστάσεων ή **δισδιάστατος**. Υπάρχουν επομένως πίνακες με περισσότερους από ένα δείκτες που ονομάζονται πίνακες δύο, τριών ή και περισσότερων διαστάσεων, ανάλογα με τον αριθμό των δεικτών. Η προσπέλαση των στοιχείων των πινάκων αυτών γίνεται με τον ανάλογο αριθμό δεικτών.

Δήλωση πολυδιάστατου πίνακα

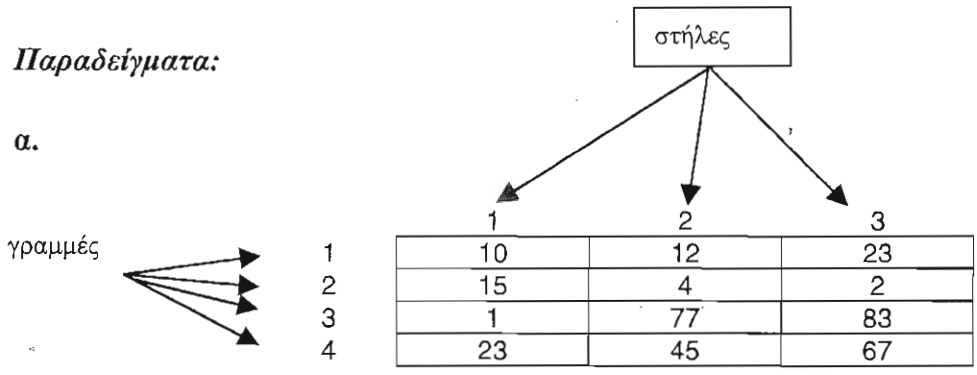
Ο τρόπος δήλωσης ενός πίνακα με όνομα A με περισσότερους από ένα δείκτες είναι:

var

A: **array** [**τύπος 1ου δείκτη, τύπος 2ου δείκτη,...**] **of** **τύπος-στοιχείων-πίνακα**

Παραδείγματα:

α.



Ο παραπάνω πίνακας αποτελείται από 4 γραμμές και 3 στήλες ανά γραμμή, όπου αποθηκεύονται ακέραιοι αριθμοί, και δηλώνεται ως εξής:

```
var
    pinakas:   array[1..4,1..3] of integer;
```

Η εκτύπωση του στοιχείου της 3ης γραμμής και 2ης στήλης γίνεται με την εντολή **writeln**(pinakas[3,2]);

β

```
var
    pin:    array[0..n,0..m] of real;
    count:  array[0..sxol, 0..diam, 0..nomos] of integer;
```

Στις παραπάνω δηλώσεις οι δείκτες είναι υποπεριοχές ακεραίου και πρέπει να έχουν προσδιοριστεί προηγουμένως οι n, m, sxol, diam, nomos να έχουν ορισθεί ως σταθερές.

γ.

```
type
    pinakas =array [1..100] of real;
var
    table :array [1..10] of pinakas
```

δ.

Ο ορισμός του παραδείγματος γ είναι ισοδύναμος με τον παρακάτω:

```
var
    table: array [1..10,1..100] of real;
```

Παραδείγματα χρήσης πινάκων

Να γραφεί πρόγραμμα το οποίο να διαβάζει τα στοιχεία ενός μονοδιάστατου πίνακα (10 στοιχείων), να βρίσκει το μικρότερο και το μεγαλύτερο στοιχείο του πίνακα καθώς και τις θέσεις τους στον πίνακα και να τυπώνει στην οθόνη το κατάλληλο μήνυμα.

```
program  pinakas1;
 uses wincrt;
 const
  n=100;
 var
  x:array[1..n] of integer;
  m,min,max:integer;
  pos_min,pos_max:integer;
procedure  eisagogi;
 var
  k:integer;
   begin
     for k:=1 to n do
       begin
         write('δωσε το στοιχείο X[',k,']=');
         readln(x[k]);
       end;
   end;

procedure min_max;
begin
  min:=x[1];
  max:=x[1];
  pos_min:=1;
  pos_max:=1;
  for m:=2 to n do
    begin
     if x[m]<min
     then
      begin
       min:=x[m] ;
       pos_min:=m;
      end;
     if x[m]>max
     then
      begin
       max:=x[m]  ;
       pos_max:=m;
     end;
  end;
  end;
procedure apotelesmata;
begin
  writeln('Max=', max:5,' στη θέση ',pos_max:3);
  writeln('Min=', min:5,' στη θέση ',pos_min:3);
end;
{κύριο πρόγραμμα}
begin
 eisagogi;
 min_max;
 apotelesmata;
end.
```

Να γραφεί πρόγραμμα που να διαβάζει το επώνυμο και το βαθμό κάθε μαθητή, να ταξινομεί τους μαθητές κατά φθίνουσα σειρά βαθμολογίας και να τυπώνει τους μαθητές κατά την ταξινόμηση που προέκυψε.

```
program pinakas2;
{*bubble sort αλγόριθμος ταξινόμησης
Στον αλγόριθμο αυτό η μεταβλητή allagi ελέγχει αν η
ταξινόμηση έχει τελειώσει, στην περίπτωση αυτή η
μεταβλητή allagi έχει τιμή false και η διαδικασία
επανάληψης τερματίζεται.*}

uses wincrt;
Const
      Plithos=40;{μέγιστο πλήθος μαθητών}
var
   i,j,no_of_marks,temp2:     integer;
   ma:array[1..plithos]   of integer;
   onoma:array[1..plithos] of string[30];
   name,temp:string[30];
   allagi:boolean;
begin
 {εισαγωγή δεδομένων }
 write('δώσε τον αριθμό των βαθμών ');
 readln(no_of_marks);
 for i:=1 to no_of_marks do
  begin
   write('δώσε το ',i:2,' βαθμό,όνομα : ');
   readln(ma[i],onoma[i]);
  end;

  {ταξινόμηση }
  allagi:=true;
while allagi do
  begin
    allagi:=false;
    for i:=1 to  no_of_marks-1  do
       if ma[i]<ma[i+1]
          then
             begin
               temp2    := ma[i];
               ma[i]    := ma[i+1];
               ma[i+1]  := temp2;
               name       := onoma[i];
               onoma[i]   := onoma[i+1];
               onoma[i+1]      := name;
               allagi     := true;
             end;
  end;

 {εκτύπωση αποτελεσμάτων}
 for i:=1 to no_of_marks  do
    writeln(i:2,' ',onoma[i],' ',ma[i]);
end.
```

14.2. ΕΓΓΡΑΦΕΣ

Ο τύπος **εγγραφής (record)** είναι μια συλλογή από συστατικά διαφόρων τύπων που το καθένα δηλώνεται με ένα όνομα. Τα συστατικά μιας εγγραφής ονομάζονται **πεδία (fields)**.Το κάθε πεδίο εγγραφής έχει το δικό του όνομα και ανήκει σ' ένα τύπο δεδομένων. Ένα πεδίο μπορεί να είναι και μια άλλη εγγραφή. Η εγγραφή είναι ένας τύπος δεδομένων χωρίς δόμηση, αφού ο μηχανισμός προσπέλασης δεν εξαρτάται από καμία δόμηση. Κάθε συστατικό στοιχείο της εγγραφής έχει το δικό του όνομα και έτσι μπορούμε να έχουμε προσπέλαση σε πεδία της εγγραφής. Μπορούμε ακόμη να εκχωρήσουμε την τιμή μιας μεταβλητής τύπου εγγραφής σε άλλη μεταβλητή ίδιου τύπου.

Δήλωση εγγραφής

Η δήλωση της εγγραφής γίνεται στην περιοχή δηλώσεων νέων τύπων και αρχίζει με το όνομα του record ακολουθούμενο από το σύμβολο = και τη λέξη **record**. Στη συνέχεια παρεμβάλλεται η λίστα με τα πεδία της εγγραφής και οι δηλώσεις των τύπων τους και ο ορισμός τερματίζεται με τη λέξη **end** ακολουθούμενη από το ερωτηματικό (;). Στο πρώτο παράδειγμα που ακολουθεί, η εγγραφή περιγράφει ένα αυτοκίνητο.

Παράδειγμα-1	Παρατηρήσεις
Type xrwma_type=(kokkino,mple, mavro, prasino, lefko) autokinito_eggrafi = **record** kostos: **real**; typos: **string**[10]; theseis: 2..5; xrwma: xrwma_type; kylindrismos: **integer**; **end**; **var** autokinito_mou, autokinito_sou: tokinito_eggrafi;	Δηλώνονται δύο τύποι: Ο απαριθμητός τύπος xrwma_type και η εγγραφή autokinito_eggrafi Όνομα της εγγραφής autokinito_eggrafi ακολουθούμενο από το σύμβολο = και τη λέξη record kostos, typos, theseis, xrwma, kylindrismos είναι τα πεδία της εγγραφής, όπου για καθένα δηλώνεται ο τύπος του και end; το τέλος του ορισμού της εγγραφής. Δηλώνονται δύο μεταβλητές autokinito_mou, autokinito_sou του τύπου εγγραφής autokinito_eggrafi
autokinito_mou := autokinito_sou	Η εντολή αντικατάστασης του Περιεχομένου ολόκληρης της εγγραφής είναι επιτρεπτή πράξη σε μεταβλητές τύπου εγγραφής

Παρατηρούμε ότι οι δηλώσεις πεδίων της εγγραφής έχουν τη μορφή δηλώσεων μεταβλητών. Πρέπει όμως να τονίσουμε ότι είναι δηλώσεις τύπων και ότι για να έχουμε προσπέλαση στη μνήμη απαιτούνται δηλώσεις μεταβλητών.

Προσπέλαση στα πεδία εγγραφής

Προσπελάσεις στα πεδία μιας εγγραφής φαίνονται στο παράδειγμα 2 που ακολουθεί.

Η αναφορά σε πεδίο μιας εγγραφής γίνεται με το όνομα της μεταβλητής τύπου record (π.χ. mathitis) που ακολουθείται από μία τελεία και το όνομα του πεδίου όπως εμφανίζεται στη δήλωση τύπου της εγγραφής (π.χ. kodikos, mathimatika κλπ). Στο παράδειγμα 2 φαίνονται προσπελάσεις της μορφής mathitis.kodikos, mathitis.mathimatika κλπ.

Για να μπορεί ο προγραμματιστής να χρησιμοποιεί το πεδίο του record χωρίς να αναφέρει το όνομα της μεταβλητής της συγκεκριμένης εγγραφής, χρησιμοποιείται η εντολή **with** που έχει τη μορφή:

with όνομα μεταβλητής τύπου record do

εντολή;

Στο παράδειγμα 2 φαίνεται η προσπέλαση στα πεδία χωρίς τη χρήση του with ή με τη χρήση του with

χωρίς τη χρήση του with

```
mo:=(mathitis.mathimatika+mathitis.fysiki+
            mathitis.ximeia+mathitis.ekthesi)/4;
```

με τη χρήση του with

```
with mathitis  do
mo:= (mathimatika+fysiki+ximeia+ekthesi)/4;
```

Χρήση μεταβλητής τύπου εγγραφής.

Στο παράδειγμα 2 που ακολουθεί γίνεται μια στοιχειώδης επεξεργασία με μεταβλητές, mathitis1, mathitis2 τύπου spoudastis ο οποίος είναι εγγραφή. Η επεξεργασία περιλαμβάνει τον ορισμό της εγγραφής spoudastis καθώς και τη δήλωση των μεταβλητών mathitis1, mathitis2 τύπου εγγραφής. Γίνεται εισαγωγή τιμών στις μεταβλητές που αντιστοιχούν σε πεδία εγγραφών. Υπολογίζεται ο μέσος όρος της βαθμολογίας με δύο τρόπους χωρίς τη χρήση του with και με τη χρήση του with. Τυπώνονται και τα δύο αποτελέσματα και διαπιστώνουμε ότι είναι τα ίδια.

Παράδειγμα 2

```
program  tee_record1;
 {*Επεξεργασία εγγραφών*}
  uses wincrt;
    type
      spoudastis = record
        kodikos:      integer;
        eponymo:            string[20];
        onoma:        string[15];
        mathimatika: integer;
        fysiki:       integer;
        ximeia:       integer;
        ekthesi:            integer;
      end;
    var
    mathitis1,mathitis2:     spoudastis;
    mo1, mo2 :                real;


 begin
   (*εισαγωγή τιμών στα πεδία εγγραφής*)
   write   ('κωδικός=');,
   readln  ( mathitis1.kodikos);
   write   ('επώνυμο=');
   readln  ( mathitis1.eponymo);
   write   ('όνομα=');
   readln  ( mathitis1.onoma);
   write   ('Μαθηματικά=');
   readln  ( mathitis1.mathimatika);
   write   ('Φυσική=');
   readln  ( mathitis1.fysiki);
   write   ('Χημεία=');
   readln  ( mathitis1.ximeia);
   write   ('Έκθεση=');
   readln  ( mathitis1.ekthesi);


  (*χωρίς τη χρήση του with**********)
```

Αποτελέσματα
Παρατηρήσεις - Επεξηγήσεις σε πλάγια μορφή

Ορισμός τύπου εγραφής
Όνομα εγγραφής **spoudastis**

Μεταξύ του ονόματος της εγγραφής και του τέλους του ορισμού της των παρεμβάλλονται οι δηλώσεις των τύπων των πεδίων της.

Έχουμε 7 πεδία: κωδικό, επώνυμο, όνομα, μαθηματικά, φυσική, χημεία, έκθεση
Τέλος ορισμού της εγγραφής

Δηλώσεις δύο μεταβλητών του τύπου εγγραφής **spoudastis**

Δίνονται τιμές στα πεδία της εγγραφής
Κωδικός =11

Επώνυμο =Ευσταθίου

όνομα =Γεώργιος

Μαθηματικά =13

Φυσική =16

Χημεία =17

Εκθεση =16

Μέσος όρος χωρίς τη χρήση του with

```
   mo1:=(mathitis1.mathimatika+
      mathitis1.fysiki+ mathitis1.ximeia+
      mathitis1.ekthesi)/4;
   writeln('μέσος όρος = ' , mo1:10:2);

 (*με τη χρήση του with ************)
     with mathitis1  do
     mo2:= (mathimatika+fysiki+
           ximeia+ekthesi)/4;
 writeln('μέσος όρος(με with) = ' ,
mo1:10:2);
```

μέσος όρος =15.50

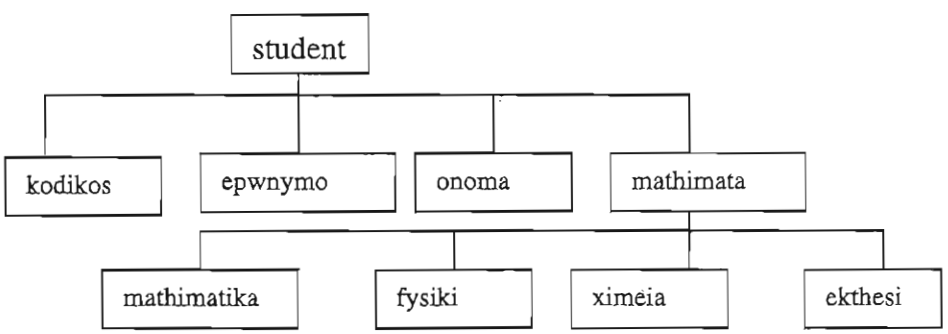
Μέσος όρος με τη χρήση του with

μέσος όρος(με with) =15.50

αποτελέσματα ίδια και με τους δύο τρόπους

Εγγραφή με πεδίο μια άλλη εγγραφή

Τα συστατικά μιας εγγραφής μπορεί να είναι οποιουδήποτε τύπου. Υπάρχουν πεδία αλφαριθμητικού τύπου καθώς και πεδία τύπου εγγραφής. Εγγραφές με πεδία άλλες εγγραφές ονομάζονται **ιεραρχικές εγγραφές.** Όπως φαίνεται στο σχεδιάγραμμα που ακολουθεί, η εγγραφή student είναι μια ιεραρχική εγγραφή, αφού το πεδίο της mathimata είναι μια άλλη εγγραφή.



Ιεραρχική εγγραφή student

Οι δηλώσεις για την ιεραρχική εγγραφή **student** φαίνονται στον πίνακα που ακολουθεί.

Η προσπέλαση στα πεδία της ιεραρχικής εγγραφής μπορεί να γίνει:

Χωρίς τη χρήση του with

```
mo:= (mathitis.bathmoi.mathimatika +
mathitis.bathmoi.fysiki+mathitis.bathmoi.ximeia+
      mathitis.bathmoi.ekthesi)/4;
```

Με τη χρήση του with

```
with mathitis  do
 mo:=(bathmoi.mathimatika+bathmoi.fysiki+bathmoi.ximeia+
      bathmoi.ekthesi)/4;

with mathitis.bathmoi do
       mo:= (mathimatika+fysiki+ximeia+ekthesi)/4;
```

Παράδειγμα 3	*Παρατηρήσεις*
Type mathimata =**record** mathimatika: **integer**; fysiki: **integer**; ximeia: **integer**; ekthesi: **integer**; **end**;	Αρχικά δηλώνεται η εγγραφή μαθήματα Η εγγραφή μαθήματα είναι πεδίο της ιεραρχικής εγγραφής student
student = **record** kodikos: **integer**; eponymo: **string**[20]; onoma: **string**[15]; bathmoi: **mathimata**; **end**;	Η εγγραφή student έχει ως πεδίο το βαθμοί τύπου μαθήματα το οποίο ορίστηκε προηγούμενα ως εγγραφή, επομένως είναι ιεραρχική εγγραφή
var mathitis: student;	Δηλώνεται η μεταβλητή mathitis τύπου student(εγγραφή)

Με τον ορισμό μιας εγγραφής ορίζεται ένα νέος τύπος δεδομένου. Ο τύπος αυτός μπορεί να χρησιμοποιηθεί για τον ορισμό ενός πίνακα ο οποίος θα περιέχει στοιχεία τύπου εγγραφής.

Παράδειγμα 4.

Για να ορίσουμε έναν πίνακα 100 θέσεων ο οποίος θα περιέχει κωδικό, επώνυμο, όνομα και τηλέφωνο μαθητών, θα πρέπει να δηλωθούν τα παρακάτω:

```
type
    student = record
    am          :integer;
    eponymo     :string[30];
    onoma       :string[20];
    telefone    :string[7];
end;
var
   mathites :array [1..100] of student;
```

Για να γίνει εμφάνιση στην οθόνη του 10ου επωνύμου θα πρέπει να γραφεί ή εξής εντολή:

```
writeln(mahtites[10].eponymo)
```

Ανακεφαλαίωση

Κατανοήσαμε τις στατικές δομές δεδομένων με τη μορφή πινάκων και εγγραφών. Γνωρίσαμε τον τρόπο ορισμού και χειρισμού πινάκων μίας και δύο διαστάσεων. Κατανοήσαμε τον τρόπο ορισμού του τύπου της εγγραφής, απλής και ιεραρχικής. Αντιληφθήκαμε επίσης τον τρόπο να ορίζουμε πίνακες που τα στοιχεία τους είναι εγγραφές. Είδαμε τέλος τη χρήση τους μέσα από χαρακτηριστικά παραδείγματα.

Ασκήσεις

1. Να γράψετε τον ορισμό τύπου για μία εγγραφή με όνομα time με τρία πεδία hour,minute,second με τύπους δεδομένων τις υποπεριοχές 0..23 και 0..59 αντίστοιχα.
2. Αν η μεταβλητή wra είναι τύπου time να γράψετε τις κατάλληλες εντολές ώστε η μεταβλητή wra να έχει τιμή 18h 30m 45sec.
3. Να γράψετε πρόγραμμα που να εκτελεί τα παρακάτω:
   α. Να διαβάζει 10 ακεραίους και να τους τοποθετεί διαδοχικά σ' έναν πίνακα μιας διάστασης.
   β. Να βρίσκει και να τυπώνει τη μικρότερη και τη μεγαλύτερη από τις τιμές αυτές καθώς και τις αντίστοιχες θέσεις τους στον πίνακα.
4. Να γράψετε πρόγραμμα που να εκτελεί τα παρακάτω:
   α. Την εισαγωγή 10 αριθμητικών στοιχείων του πίνακα.
   β. Την εκτύπωση του πίνακα.
   γ. Την ταξινόμηση κατ' αύξουσα σειρά των στοιχείων του πίνακα.
   δ. Την εκτύπωση του ταξινομημένου πίνακα.

Υπόδειξη: Να συγκριθεί το πρώτο στοιχείο με όλα τα άλλα και τελικά το μικρότερο στοιχείο να πάει στην πρώτη θέση. Να γίνει η ίδια διαδικασία για τα υπόλοιπα και στη θέση του δεύτερου να πάει το αμέσως μεγαλύτερο. Να επαναληφθεί η ίδια διαδικασία μέχρι και το προτελευταίο στοιχείο και έτσι όλα τα στοιχεία να τοποθετηθούν στον πίνακα κατ' αύξουσα σειρά.

5. Να γράψετε πρόγραμμα που να εκτελεί τα παρακάτω:
   α. Την εισαγωγή 10 αριθμητικών στοιχείων πίνακα.
   β. Την εκτύπωση του πίνακα.
   γ. Την ταξινόμηση κατά φθίνουσα σειρά των στοιχείων του πίνακα.
   δ. Την εκτύπωση του ταξινομημένου πίνακα.

Υπόδειξη: Κάθε στοιχείο συγκρίνεται με το επόμενο και, αν είναι μικρότερο αλλάζουν θέση. Η διαδικασία επαναλαμβάνεται μέχρις ότου όλα τα στοιχεία πάνε στη σωστή θέση και δεν υπάρχουν άλλες αλλαγές θέσης. Έτσι, όλα τα στοιχεία θα τοποθετηθούν στον πίνακα κατά φθίνουσα σειρά.

6. Να γραφεί πρόγραμμα το οποίο, αφού διαβάσει τη διάσταση ενός μονοδιάστατου πίνακα με 40 το πολύ στοιχεία, εκτελεί τις παρακάτω διαδικασίες:

   α. Διαβάζει από την οθόνη και αποθηκεύει σε πίνακα τα ονόματα και τους 4 βαθμούς Ν μαθητών.

   β. Για κάθε μαθητή υπολογίζει το μέσο όρο και τον αποθηκεύει στην αντίστοιχη θέση του πίνακα.

   γ. Ταξινομεί τον πίνακα αλφαβητικά και τυπώνει το ονοματεπώνυμο και το μέσο όρο για κάθε μαθητή.

   γ. Ταξινομεί τον πίνακα κατά φθίνουσα σειρά του μέσου όρου και τυπώνει το ονοματεπώνυμο και το μέσο όρο για κάθε μαθητή.

   Υπόδειξη:
   Κάθε στοιχείο του πίνακα να είναι μία εγγραφή με την εξής δομή:

   ονοματεπώνυμο
   βαθμός-1
   βαθμός-2
   βαθμός-3
   βαθμός-4
   Μέσος όρος

7. Να γραφεί πρόγραμμα το οποίο, αφού διαβάσει τη διάσταση Ν ενός μονοδιάστατου πίνακα με 40 το πολύ στοιχεία, εκτελεί τις παρακάτω διαδικασίες:

   α. Διαβάζει από την οθόνη και αποθηκεύει σε πίνακα τα ονόματα και το βαθμό Ν μαθητών.

   β. Υπολογίζει το μέσο όρο των βαθμών των μαθητών και τυπώνει αντίστοιχο μήνυμα.

   γ. Ταξινομεί τον πίνακα κατά φθίνουσα σειρά βαθμολογίας.

   δ. Δίνει χαρακτηρισμό για κάθε μαθητή ανάλογα με το βαθμό του μέσα στο σύνολο του πίνακα και τυπώνει τα αποτελέσματα σύμφωνα με το υπόδειγμα:

   Ονοματεπώνυμο, βαθμός, χαρακτηρισμός, απόκλιση από ΜΟ.

Υπόδειξη: Για το χαρακτηρισμό ορίζονται 4 κλάσεις που ισοδυναμούν με: Άριστα, Λίαν Καλώς, Καλώς, Σχεδόν Καλώς. Το πλήθος των μαθητών κάθε κλάσης ορίζεται Ν/4. Οι κλάσεις ορίζονται από πάνω προς τα κάτω, δηλαδή από την υψηλότερη προς τη χαμηλότερη βαθμολογία. Σε περίπτωση ισοβαθμίας όλοι οι ισοβαθμούντες με τον τελευταίο της κλάσης ανήκουν στην υψηλότερη κλάση και το πλήθος τους αφαιρείται από την επόμενη κλάση.

8. Να γραφεί πρόγραμμα το οποίο, αφού διαβάσει τη διάσταση N, δύο μονοδιάστατων πινάκων A,B με 20 το πολύ στοιχεία εκτελεί τις παρακάτω διαδικασίες.
   α. Με τη χρήση της συνάρτησης RANDOM(N) παράγει και αποθηκεύει N στοιχεία για κάθε πίνακα.
   β. Συγκρίνει τα αντίστοιχα στοιχεία (με τον ίδιο δείκτη) των δύο πινάκων και ανάλογα με το αποτέλεσμα της σύγκρισης δημιουργεί έναν τρίτο πίνακα C με στοιχεία <, > και = αντίστοιχα.
   γ. Εκτυπώνει τα περιεχόμενα των τριών πινάκων σύμφωνα με το υπόδειγμα:

A	C	B
15	>	10
20	=	20
40	<	120

   δ. Τυπώνει τον αριθμό των <, >, = που προκύπτουν από την παραπάνω σύγκριση.

Δραστηριότητες

1. Να γραφεί πρόγραμμα το οποίο αφού διαβάσει τη διάσταση N, ενός μονοδιάστατου πινάκα A για την αποθήκευση των στοιχείων των μαθητών της τάξης σας, εκτελεί τις παρακάτω διαδικασίες:
   α. Διαβάζει από την οθόνη και αποθηκεύει σε πίνακα τον κωδικό, το όνομα και το βαθμό N μαθητών.
   β. Διαβάζει από την οθόνη έναν αριθμό που δηλώνει τον κωδικό μαθητή.
   γ. Ταξινομεί τον πίνακα κατ' αύξουσα σειρά κωδικών.
   δ. Κάνει σειριακή αναζήτηση στον πίνακα για ανεύρεση του συγκεκριμένου μαθητή. Αν ο μαθητής υπάρχει στον πίνακα, τότε τυπώνονται τα στοιχεία του, αλλιώς τυπώνεται κατάλληλο μήνυμα.
   ε. Κάνει δυαδική (binary) αναζήτηση στον πίνακα για ανεύρεση του συγκεκριμένου μαθητή.

Ψευδοκώδικας του αλγόριθμου δυαδικής αναζήτησης.

```
pd=1 {πρώτος δείκτης του πίνακα των κωδικών}
td=N {τελευταίος δείκτης του πίνακα των κωδικών }
br='οχι'{ ένδειξη εύρεσης του ζητούμενου κωδικού }
επανάλαβε εφόσον (pd<td)  και  (br='οχι')
 αρχή
mesos=(pd+td)/2
αν kodikos(mesos)=code
τότε
  αρχή
    br='ναι'
    thesi=mesos
  τέλος
```

```
αλλιώς
   αν kodikos(mesos)>code
   τότε  td=mesos-1
   αλλιώς pd=mesos+1
 τέλος
αν br='ναι'
τότε τύπωσε onoma(thesi),bathmos(thesi)
αλλιώς τύπωσε 'δεν υπάρχει αυτός ο κωδικός '.
```

2. Δίνεται πίνακας δύο διαστάσεων που περιέχει τις τιμές πώλησης 5 προϊόντων(στήλες πίνακα) ανάλογα με την ποσότητα (γραμμές πίνακα) πώλησης. Ο πίνακας έχει 3 γραμμές και 5 στήλες, δηλαδή έχουμε 3 διαφορετικές τιμές πώλησης, ανάλογα με την ποσότητα, για καθένα από τα 5 προϊόντα. Τα δεδομένα για τον πίνακα είναι:

		προϊόντα				
	τιμές	1	2	3	4	5
ποσότητα <=100	1	30	28	40	59	60
ποσότητα > 100	2	28	20	30	45	50
ποσότητα > 500	3	18	15	20	30	40

Να γράψετε πρόγραμμα που:

α. να δημιουργεί τον παραπάνω πίνακα διπλής εισόδου με είσοδο των στοιχείων του από την οθόνη,

β. να διαβάζει από την οθόνη τον κωδικό του προϊόντος(1,2,3,4,5) και την ποσότητα πώλησης,

γ. να υπολογίζει και να τυπώνει την αξία πωληθείσης ποσότητας με προσπέλαση στον πίνακα διπλής εισόδου.

3. Να γραφεί ένα πρόγραμμα το οποίο αφού διαβάσει τη διάσταση N δύο τετραγωνικών πινάκων (A(n,n), B(n,n)) εκτελεί τις παρακάτω εργασίες:

α. διαβάζει τα στοιχεία των δύο πινάκων,

β. υπολογίζει τον πίνακα C(n,n) που προκύπτει από το άθροισμα των πινάκων A και B,

γ. υπολογίζει τον πίνακα D(n,n) που προκύπτει από το γινόμενο των πινάκων A και B και

δ. τυπώνει το περιεχόμενο των τεσσάρων πινάκων στην οθόνη.

Υπόδειξη:

1. Κάθε στοιχείο του πίνακα C, το c(i,j) υπολογίζεται από τον τύπο:

$$c(i,j) = a(i,j)+b(i,j) \text{ όταν } (i,j = 1,..n)$$

2. Κάθε στοιχείο του πίνακα D, το d(i,j) υπολογίζεται από τον τύπο:

$$d(i,j) = \sum_{k=1}^{n} a(i,k)*b(k,j) \text{ όταν } (i,j = 1,..n)$$

Σε μια δημοσκόπηση για ένα εκπαιδευτικό θέμα, τα αποτελέσματα καταχωρήθηκαν στον παρακάτω πίνακα διπλής εισόδου.

Γνώμη	Μαθητές	Γονείς	Καθηγητές
Υπέρ			
Κατά			
Αδιάφοροι			

Να γραφεί πρόγραμμα το οποίο να εκτελεί τις παρακάτω διαδικασίες:

α. να διαβάζει από την οθόνη τα δεδομένα και να καταχωρεί σε πίνακα δύο διαστάσεων στη μνήμη,

β. να τυπώνει στην οθόνη τον παρακάτω πίνακα διπλής εισόδου που περιλαμβάνει τα αθροίσματα κατά γραμμές και στήλες,

γ. να τυπώνει στην οθόνη ένα πίνακα διπλής εισόδου που θα περιλαμβάνει τα ποσοστά κατά γραμμές και,

δ. να τυπώνει στην οθόνη ένα πίνακα διπλής εισόδου που θα περιλαμβάνει τα ποσοστά κατά στήλες.

Γνώμη	Μαθητές	Γονείς	Καθηγητές	Σύνολα
Υπέρ				
Κατά				
Αδιάφοροι				

ΚΕΦΑΛΑΙΟ 16
ΔΥΝΑΜΙΚΕΣ ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ

Σκοπός κεφαλαίου:

Να κατανοήσουμε τη διαφορά των δυναμικών δομών δεδομένων από τις αντίστοιχες στατικές.

Ειδικοί σκοποί:

- Να γνωρίσουμε τον τρόπο χειρισμού μιας απλής συνδεδεμένης λίστας.
- Να γνωρίσουμε τη δομή και τον τρόπο χειρισμού μιας στοίβας.
- Να γνωρίσουμε τη δομή και τον τρόπο χειρισμού μιας ουράς.
- Να γνωρίσουμε τη δενδρική δομή.

16.1. ΒΑΣΙΚΕΣ ΕΝΝΟΙΕΣ

Οι δυναμικές δομές δεδομένων δε δεσμεύουν σταθερό μέγεθος μνήμης, όπως συμβαίνει με τις στατικές δομές, αλλά μεταβλητό το οποίο αυξομειώνεται κατά τις ανάγκες του προγράμματος. Δυναμικές δομές δεδομένων είναι οι **συνδεδεμένες λίστες (linked lists)**, η **στοίβα (stack)**, **η ουρά (queue)**, τα **δυαδικά δένδρα (binary trees)** κλπ.

Για τη δημιουργία δυναμικών δομών χρησιμοποιείται ένας άλλος τύπος της Pascal ο **δείκτης (pointer)**. Μια μεταβλητή τύπου δείκτη περιέχει τη διεύθυνση μιας άλλης μεταβλητής. Οι δυναμικές δομές αναπτύσσονται στη διάρκεια της εκτέλεσης του προγράμματος. Αποτελούνται από **κόμβους (nodes)** που ο καθένας έχει δύο πεδία, το στοιχείο της δομής και το δείκτη στη διεύθυνση μιας άλλης μεταβλητής.

16.2. ΣΥΝΔΕΔΕΜΕΝΕΣ ΛΙΣΤΕΣ

Μια κατηγορία δυναμικών δομών είναι οι συνδεδεμένες λίστες, όπου τα δεδομένα συνδέονται μεταξύ τους με συνδέσμους. Κάθε στοιχείο μιας τέτοιας δομής, που ονομάζεται **κόμβος,** περιέχει, εκτός από τις τιμές των δεδομένων και ένα ή περισσότερα πεδία-δείκτες που περιέχουν διευθύνσεις άλλων στοιχείων της ίδιας δομής. Τα πεδία-δείκτες είναι ένας τύπος του ο-

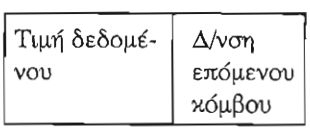
ποίου οι τιμές του είναι διευθύνσεις άλλων κόμβων. Υπάρχουν λίστες με ένα σύνδεσμο και λίστες με περισσότερους συνδέσμους.

Οι συνδεδεμένες λίστες είναι μια δομή δεδομένων στην οποία οι κόμβοι της λίστας δε βρίσκονται σε διαδοχικές θέσεις στην κύρια μνήμη, αλλά βρίσκονται σε τυχαίες θέσεις και συνδέονται μεταξύ τους με ειδικούς συνδέσμους. Με τους συνδέσμους αυτούς καθορίζεται και η διαδοχή των όρων της λίστας. Ο δείκτης συνδέει κάθε κόμβο με τον επόμενό του, δηλαδή περιέχει τη διεύθυνση του επόμενου κόμβου. Ο δείκτης του τελευταίου κόμβου παίρνει την τιμή nil που δηλώνει ότι δεν υπάρχει επόμενος κόμβος στη λίστα.

Τέλος, η διεύθυνση του πρώτου κόμβου της λίστας υπάρχει σε μια μεταβλητή, τύπου δείκτη, που τη λέμε **ΑΡΧΗ.**

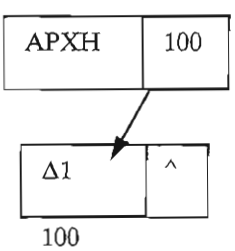
Κατά τη επεξεργασία μιας λίστας μπορούν να γίνουν οι εξής βασικές λειτουργίες:

- Εισαγωγή ενός νέου κόμβου.
- Αναζήτηση κάποιου κόμβου για επεξεργασία του περιεχομένου του ή εκτύπωση.
- Διαγραφή ενός κόμβου.

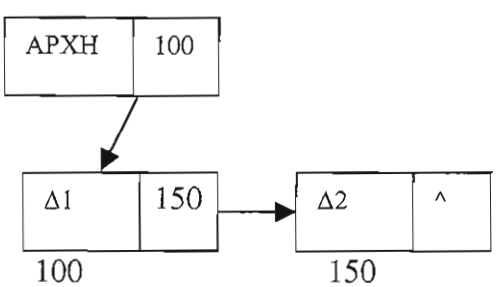


Σχήμα 16-1 Δομή κόμβου

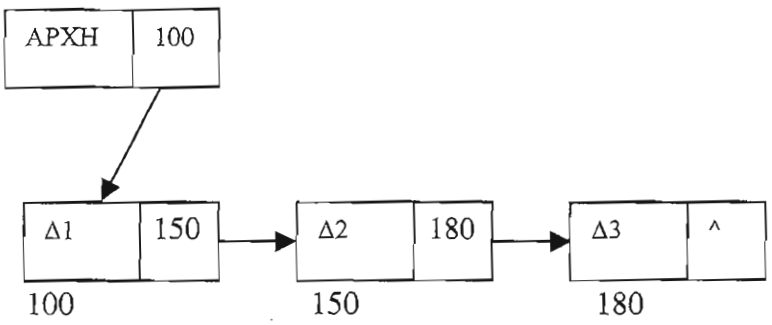
Παράδειγμα γραμμικής λίστας.



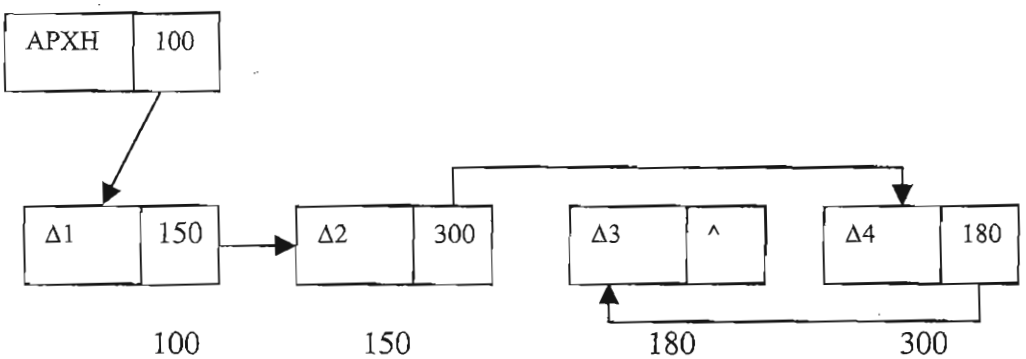
Σχήμα 16-2 Εισαγωγή του πρώτου κόμβου με δεδομένο το Δ1 στη διεύθυνση 100



Σχήμα 16-3 Εισαγωγή δεύτερου κόμβου με δεδομένο το Δ2 στη διεύθυνση 150



Σχήμα 16-4 *Εισαγωγή του τρίτου κόμβου με δεδομένο το Δ3 στη διεύθυνση 180*



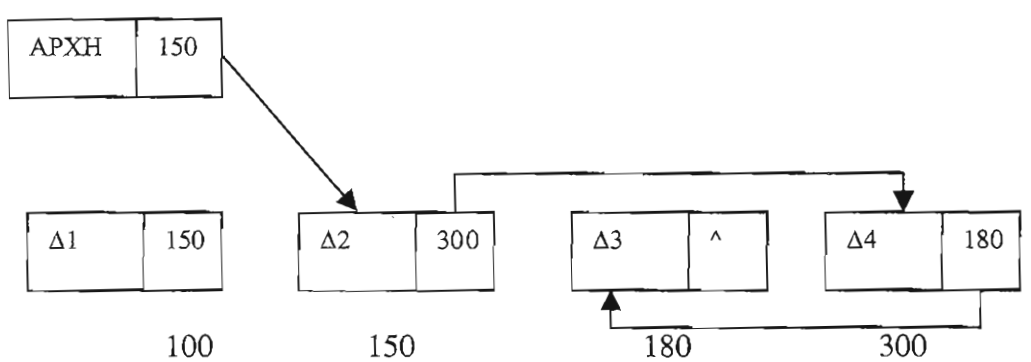
Σχήμα 16-5 *Εισαγωγή του τέταρτου κόμβου με δεδομένο το Δ4 στη διεύθυνση 300*

Στην περίπτωση αυτή, επισημαίνεται ότι η φυσική σειρά των δεδομένων διαφέρει από τη λογική, η οποία καθορίζεται από το δείκτη, όπως φαίνεται στον πίνακα που ακολουθεί.

Σειρά διευθύνσεων	100	150	180	300
Φυσική σειρά δεδ.	Δ1	Δ2	Δ3	Δ4
Λογική σειρά δεδ.	Δ1	Δ2	Δ4	Δ3

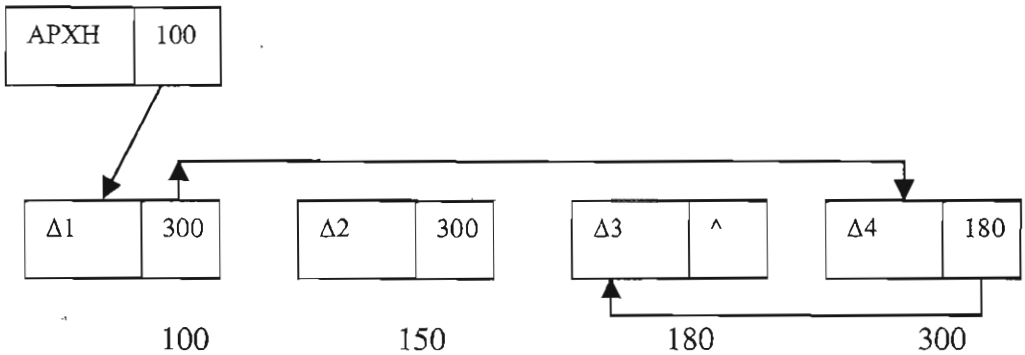
Η διαγραφή ενός κόμβου σε μία λίστα, όπως η παραπάνω, μπορεί να διακριθεί σε τρεις περιπτώσεις:

- **Διαγραφή του λογικά πρώτου κόμβου που δείχνει ο δείκτης ΑΡΧΗ**



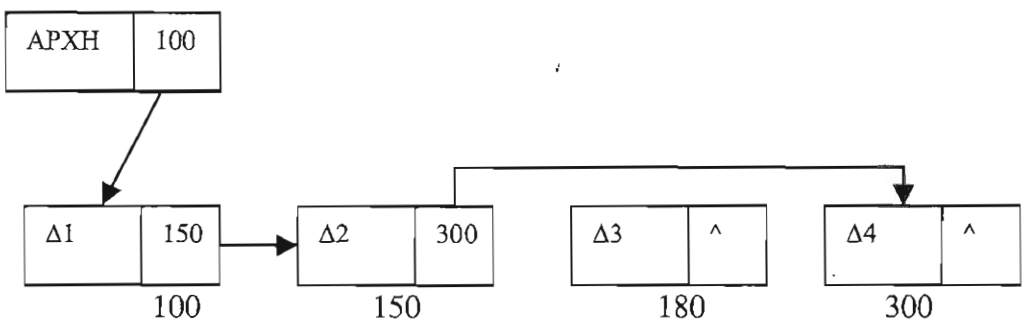
Σχήμα 16-6 *Διαγραφή του πρώτου κόμβου με περιεχόμενο Δ1 στη διεύθυνση 100*

- **Διαγραφή ενός κόμβου μεταξύ δύο άλλων.**



Σχήμα 16-7 Διαγραφή του κόμβου με περιεχόμενο Δ2 στη διεύθυνση 150

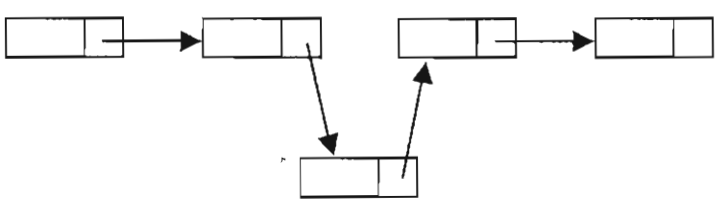
- **Διαγραφή του λογικά τελευταίου κόμβου, δηλαδή αυτού του οποίου ο δείκτης δεν δείχνει σε άλλο κόμβο.**



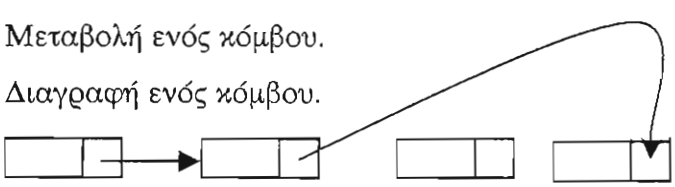
Σχήμα 16-8 Διαγραφή του κόμβου με περιεχόμενο Δ3 στη διεύθυνση 180

Κατά τη επεξεργασία μιας λίστας μπορούν να γίνουν οι εξής βασικές λειτουργίες:

- Εισαγωγή ενός νέου κόμβου.



- Αναζήτηση κάποιου κόμβου.
- Μεταβολή ενός κόμβου.
- Διαγραφή ενός κόμβου.

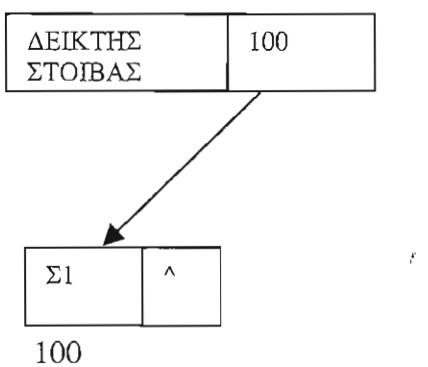


16.3. ΣΤΟΙΒΑ

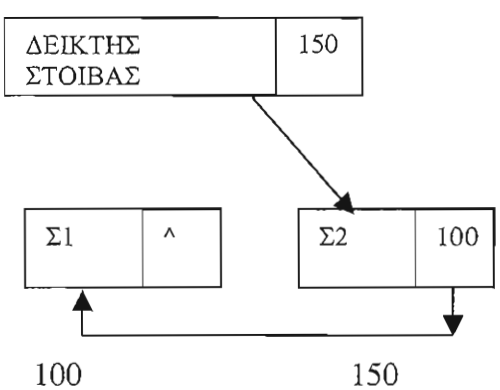
Γενικά σε μία λίστα, η εισαγωγή ή διαγραφή κάποιου στοιχείου μπορεί να γίνει σε οποιαδήποτε θέση της λίστας. Μια ειδική κατηγορία λίστας που επιτρέπει την εισαγωγή και εξαγωγή στοιχείων μόνο από το ένα άκρο της, ονομάζεται **στοίβα** ή **σωρός(stack).**

Η δομή αυτή θυμίζει στοίβα από πιάτα ή τον κερματοδέκτη εισπράκτορα ενός λεωφορείου. Και στις δύο περιπτώσεις παίρνουμε από την κορυφή ή τοποθετούμε στην κορυφή (το ένα άκρο) της στοίβας. Για τους λόγους αυτούς η επεξεργασία δεδομένων της στοίβας έχει καθιερωθεί με την ονομασία ΤΜΠΕ (Τελευταίο Μέσα Πρώτο Έξω) που αποτελεί μετάφραση μετάφραση του αντίστοιχου αγγλικού **LIFO** (**L**ast **I**n **F**irst **O**ut). Η εισαγωγή και η εξαγωγή στοιχείων αποτελούν λειτουργίες που είναι επιτρεπτές μόνο στο ένα άκρο της στοίβας που λέγεται κορυφή. Η διεύθυνση της κορυφής μιας στοίβας υπάρχει σ' έναν ειδικό δείκτη που λέγεται ***Δείκτης Στοίβας ή Δείκτης Σωρού*** (**Stack Pointer**). Η δομή της στοίβας χρησιμοποιείται από τις γλώσσες προγραμματισμού για την υλοποίηση της αναδρομής.

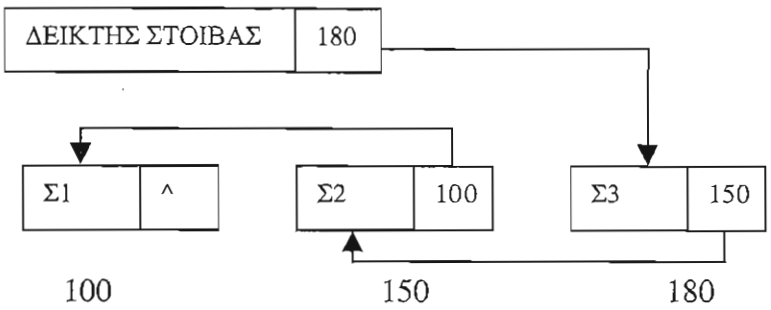
Παράδειγμα υλοποίησης στοίβας με συνδεδεμένη λίστα:



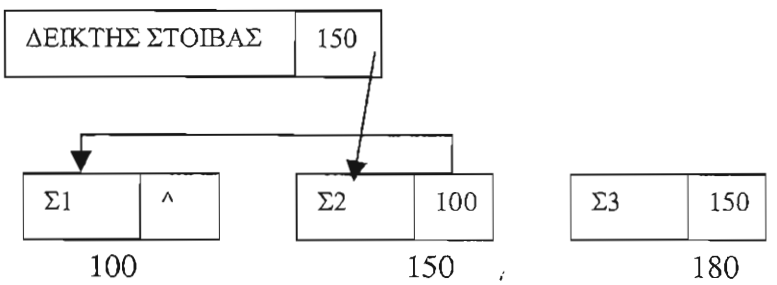
Σχήμα 16-9 *Εισαγωγή πρώτου κόμβου με δεδομένα Σ1 στη διεύθυνση 100*



Σχήμα 16-10 *Εισαγωγή δεύτερου κόμβου με δεδομένα Σ2 στη διεύθυνση 150*



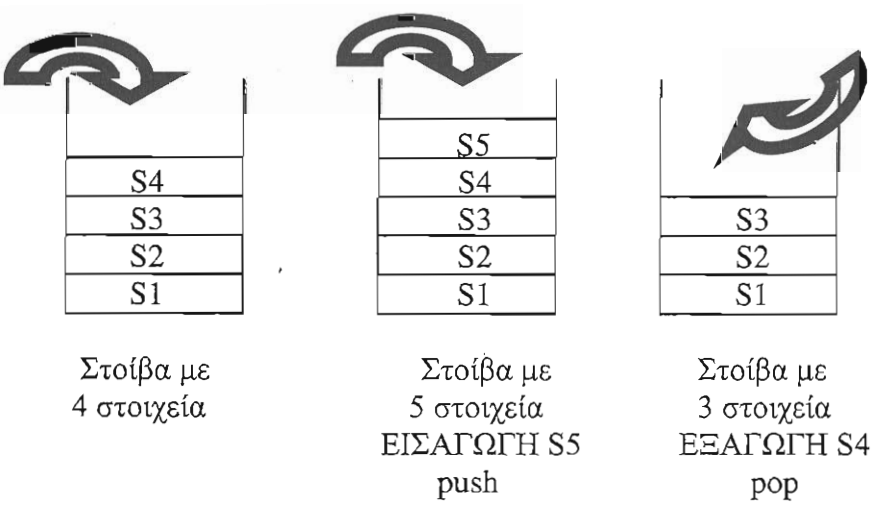
Σχήμα 16-11 *Εισαγωγή του κόμβου με περιεχόμενο Σ3 στη διεύθυνση 180*



Σχήμα 16-12 *Διαγραφή κόμβου με περιεχόμενο Σ3 στην κορυφή της στοίβας*

Η διαγραφή του κόμβου στο σωρό θα γίνει στο κόμβο με περιεχόμενο Σ3 στη δ/νση 180 που δείχνει ο κόμβος ΔΕΙΚΤΗΣ ΣΤΟΙΒΑΣ. Στη συνέχεια ο ΔΕΙΚΤΗΣ ΣΤΟΙΒΑΣ θα έχει ως περιεχόμενο τη διεύθυνση που ήταν καταχωρημένη στον κόμβο που διαγράφηκε και έδειχνε στον αμέσως προηγούμενο κόμβο, δηλαδή στη διεύθυνση 150.

Παρακάτω δίνεται οι αλγορίθμοι για την εισαγωγή και εξαγωγή στοιχείων σ' ένα σωρό σε σχεδιάγραμμα, ψευδοκώδικα και πρόγραμμα pascal.



Σχήμα 16-13 *Χειρισμός στοίβας*

ΑΛΓΟΡΙΘΜΟΣ ΕΙΣΑΓΩΓΗΣ (PUSH)

ΛΑΒΕ τη διεύθυνση μια νέας θέσης στη μνήμη
ΒΑΛΕ τη διεύθυνση της νέας θέσης στο δείκτη στοίβας
ΒΑΛΕ το στοιχείο στη διεύθυνση που έχει ο δείκτης στοίβας

ΑΛΓΟΡΙΘΜΟΣ ΕΞΑΓΩΓΗΣ (POP)

ΒΓΑΛΕ το στοιχείο από τη διεύθυνση που έχει ο δείκτης στοίβας

ΒΑΛΕ στο δείκτη στοίβας τη διεύθυνση που έχει ο δείκτης του κόμβου που διεγράφη

```
program stack;
uses wincrt;
type
 deiktis=^kombos;{μεταβλητή(deiktis) ως δείκτη του κόμβου kombos}
   kombos=record                {Δήλωση του κόμβου με όνομα kombos}
   info:string[20];             {πεδίο δεδομένων του κόμβου}
   epomenos:deiktis;            {δείκτης, διεύθυνσης του επόμενου κόμβου}
 end;
var
  arxi,neos,next :deiktis; {Βοηθητικές μεταβλητές  δεικτών}
  stoixeio:string[20];
  epil:integer;
  again:boolean;
```

```
Procedure create;  {διαδικασία δημιουργίας της στοίβας}
Begin
  arxi:=nil;
end;
```

```
procedure eisagogi; { εισαγωγής ενός νέου  κόμβου στη στοίβα}
begin
  write('Δώσε το στοιχείο ');
  readln(stoixeio);     { εισαγωγή  από την  { οθόνη}
  if arxi =nil    {Η στοίβα είναι άδεια  και εισάγεται ο πρώτος κόμβος }
    then
     begin
       new(neos);{Χώρος στη μνήμη για ένα νέο κόμβο με όνόμα neos }
       neos^.info:=stoixeio; { αντικατάσταση του πεδίου info του νέου
                               κόμβου με το περιεχόμενο της  μεταβλητής
                               stoixeio}
       neos^.epomenos:=nil;{Ως διεύθυνση επόμενου κόμβου τίθεται  nil }
       arxi:=neos;     { Ως διεύθυνση  του πρώτου κόμβου της στοίβας
                         τίθεται  η διεύθυνση  του νέου κόμβου }
     end
    else
      begin      {η στοίβα περιέχει τουλάχιστον ένα κόμβο }
       new(neos);     {Κατάληψη χώρου στη μνήμη για ένα νέο κόμβο
                       με όνομα neos }
       neos^.info:=stoixeio;  {αντικατάσταση του πεδίου info του νέου
                              κόμβου με το περιεχόμενο της μεταβλητής
                              stoixeio}
         neos^.epomenos:=arxi; {Ως διεύθυνση επόμενου κόμβου  η
                                διεύθυνση που μέχρι τώρα  έδειχνε στο
                                πρώτο κόμβο }
          arxi:=neos;      { Ως διεύθυνση πρώτου κόμβου της στοίβας
                             τίθεται η διεύθυνση του νέου κόμβου }
       end;
end;
```

```
procedure ektyposi;
begin
  next:=arxi ; { η μεταβλητή next λαμβάνει τιμή την διεύθυνση του
                 πρώτου προς εξαγωγή κόμβου}
  while next<>nil do   {επαναληπτική διαδικασία εφόσον η μεταβλητή
                        next έχει τιμή διαφορετική της nil}
    begin
      writeln(next^.info);  {Εκτύπωση στην οθόνη του περιεχομένου του
                             κόμβου με διεύθυνση αυτή που έχει η μεταβλητή
                             next }
      next:=next^.epomenos;  {Η διεύθυνση του επόμενου κόμβου θα είναι
                              ότι περιέχεται στο πεδίο epomenos του
                              κόμβου που εκτυπώθηκε}
    end;
end;
```

```
procedure diagrafi;
begin
 if arxi=nil           {Εάν η στοίβα είναι άδεια}
    then
      writeln('Η στοίβα είναι άδεια')
    else           {εάν η στοίβα δεν είναι άδεια }
      begin
        writeln(arxi^.info);  {Εκτύπωση στην οθόνη του περιεχομένου
              του κόμβου με διεύθυνση αυτή που έχει η μεταβλητή arxi η
              οποία δείχνει στο πρώτο προς διαγραφή κόμβο}
        arxi:=arxi^.epomenos;  {Ο νέος προς διαγραφή κόμβος θα έχει
                                διεύθυνση αυτή που δείχνει η μεταβλητή
                                epomenos του διαγεγραμμένου κόμβου}
      end;
end;
```

```
begin   again:=true;      create;
   while again do
     begin
       writeln('ΕΙΣΑΓΩΓΗ   1');
       writeln('ΔΙΑΓΡΑΦΗ   2');
       writeln('ΕΚΤΥΠΩΣΗ   3');
       writeln('ΤΕΛΟΣ      0');
       readln (epil);
       if epil=1 then eisagogi;
       if epil=2 then diagrafi;
       if epil=3 then ektyposi;
       if epil=0 then again:=false;
     end;
 end.
```

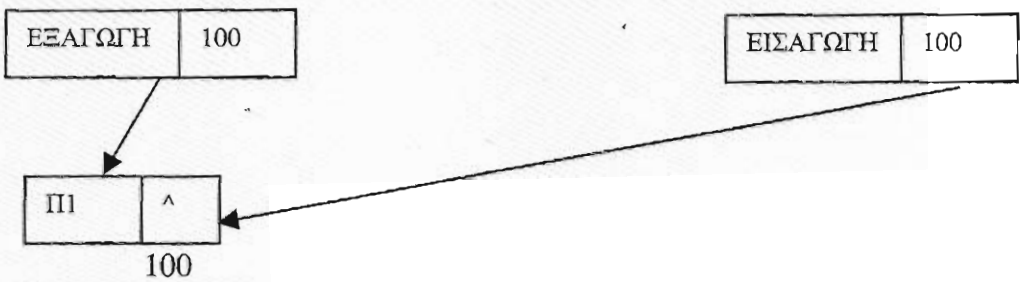
16.4. ΟΥΡΑ

Η **Ουρά (Queue)** είναι μια άλλη Δομή Δεδομένων, όπου οι εισαγωγές γίνονται από το ένα άκρο της και οι εξαγωγές από το άλλο. Σε μια τέτοια δομή, το στοιχείο που εξάγεται πρώτο είναι αυτό που έχει εισαχθεί πρώτο, γι αυτό και η επεξεργασία δεδομένων λέγεται ΠΜΠΕ (Πρώτο Μέσα Πρώτο Έξω), που απαοτελεί μετάφραση του αγγλικού **FIFO (First In - First Out)**.

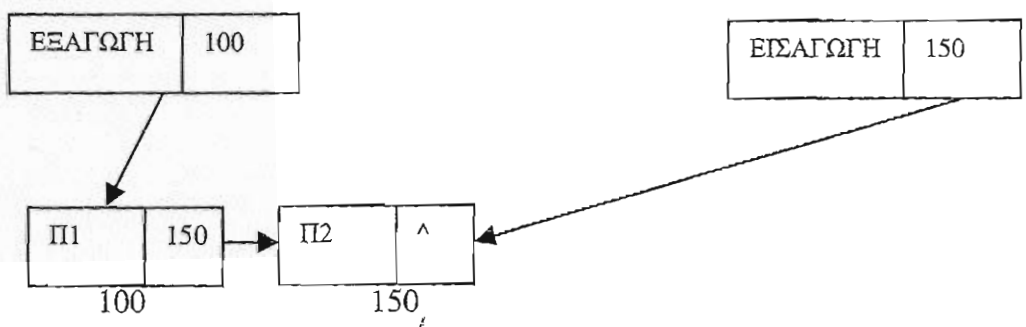
Ένα παράδειγμα ουράς είναι η εξυπηρέτηση των πελατών στο ταμείο μίας τράπεζας ή οι λίστες αναμονής επιβατών σε αεροπορικές πτήσεις.

Για τον χειρισμό των κόμβων της ουράς χρειάζονται δύο δείκτες. Ο δείκτης ΑΡΧΗ θα δείχνει το πρώτο στοιχείο που τοποθετήθηκε στην ουρά ενώ ο δείκτης ΤΕΛΟΣ θα δείχνει αντίστοιχα το τελευταίο. Η εισαγωγή θα γίνεται με βάση το δείκτη ΕΙΣΑΓΩΓΗΣ και η εξαγωγή με βάση το δείκτη ΕΞΑΓΩΓΗΣ.

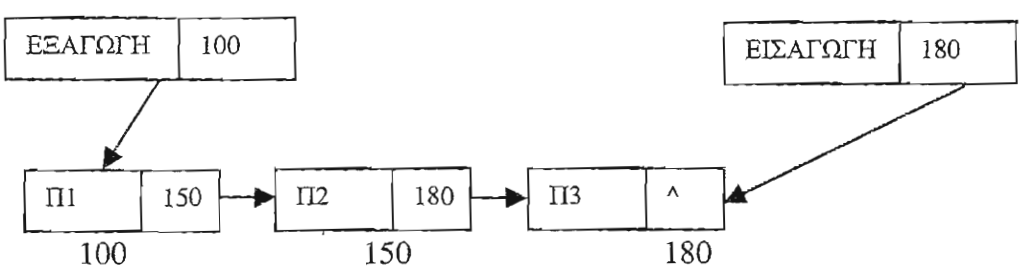
Κάθε κόμβος της ουράς περιέχει εκτός των δεδομένων και ένα δείκτη που δείχνει τη διεύθυνση του επομένου προς διαγραφή κόμβου.



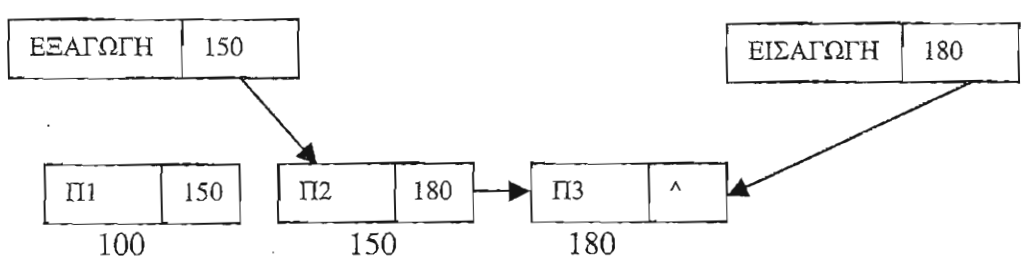
Σχήμα 16-14 *Εισαγωγή του πρώτου κόμβου με δεδομένα Π1 στη διεύθυνση 100*



Σχήμα 16-15 *Εισαγωγή του κόμβου με δεδομένα Π2 στη διεύθυνση 150*



Σχήμα 16-16 *Εισαγωγή του κόμβου με δεδομένα το Π3 στη διεύθυνση 180*



Σχήμα 16-17 *Διαγραφή του κόμβου με δεδομένα το Π1 στη διεύθυνση 100*

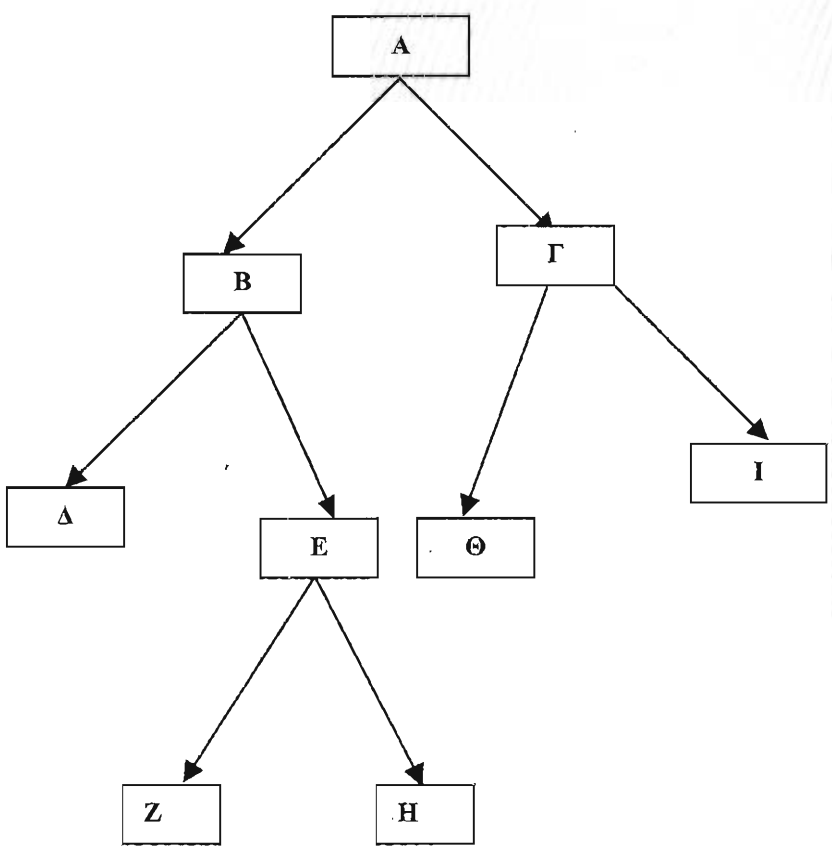
16.5. ΔΕΝΔΡΑ

Οι προηγούμενες δυναμικές δομές είναι γραμμικές, επειδή στις δομές αυτές κάθε κόμβος σχετίζεται μόνο μ' ένα άλλο κόμβο (σχέση ένα με ένα). Σε μια λίστα, για παράδειγμα, ένας κόμβος έχει μόνο έναν επόμενο κόμβο.

Τα **δένδρα (trees)** είναι ιεραρχικές δομές δεδομένων όπου κάθε κόμβος σχετίζεται με πολλούς (σχέση ένα με πολλά). Οι ιεραρχικές δομές αναπαριστάνουν, μ' ένα φυσικό τρόπο, τις σχέσεις των στοιχείων του κόσμου μας κι έτσι, η χρήση τους στην επεξεργασία δεδομένων αποκτά μεγάλη σπουδαιότητα. Η σχέση του πατέρα με τα παιδιά είναι μια σχέση ένα με πολλά (οι σχέσεις ένα με κανένα και ένα με ένα μπορούν να θεωρηθούν ως ειδικές περιπτώσεις της σχέσης ένα με πολλά). Η σχέση μεταξύ οχημάτων και επιβατών, ομάδων ποδοσφαίρου και παικτών είναι επίσης σχέσεις ιεραρχικές (ένα με πολλά).

Στην επεξεργασία δεδομένων τις ιεραρχικές δομές δεδομένων τις λέμε **δένδρα (trees)**.

Σε κάθε δέντρο υπάρχει ένας και μοναδικός κόμβος που δεν έχει προηγούμενο κόμβο και λέγεται **ρίζα (root)**.



Σχήμα 9-18 Δομή δένδρου

Στο προηγούμενο σχήμα ο κόμβος Α είναι η ρίζα του δένδρου. Οι κόμβοι που δεν έχουν επόμενους κόμβους λέγονται **φύλλα (leaf)**. Στο σχήμα 9-18 κόμβοι-φύλλα είναι οι **Δ, Ζ, Η, Θ** και **Ι**. Κάθε κόμβος που δεν είναι ούτε ρίζα, ούτε φύλλο έχει ένα και μοναδικό προηγούμενο κόμβο και τουλάχιστον έναν επόμενο κόμβο. Στο σχήμα 9-18, ο κόμβος **Β** έχει προηγούμενο τον κόμβο **Α** και επόμενους τους **Ζ** και **Η**, ο κόμβος **Γ** έχει προηγούμενο τον **Α** και επόμενους τους **Ι** και **Θ**. Κάθε κόμβος σ ' ένα δένδρο, μαζί με όλους τους επόμενους κόμβους του, ορίζει ένα μέρος του δένδρου που το λέμε **υποδένδρο (subtree)**. Στο σχήμα 9-18 ο κόμβος **Β** ορίζει ένα υποδένδρο που έχει τους κόμβους **Β, Δ, Ε, Ζ, Η**.

Ένα δένδρο, που κάθε κόμβος του έχει το πολύ δύο επόμενους κόμβους, τον αριστερό και τον δεξιό ,λέγεται **δυαδικό δέντρο (binary tree)**. Το σχήμα 9-18 είναι ένα δυαδικό δένδρο.

Ανακεφαλαίωση

Γνωρίσαμε και κατανοήσαμε τη διαφορά των δυναμικών δομών από τις στατικές, καθώς και τη χρησιμότητα των δυναμικών δομών στην επεξεργασία δεδομένων. Γνωρίσαμε τις διάφορες κατηγορίες δυναμικών δομών όπως συνδεδεμένες λίστες, στοίβες, ουρές και δένδρα. Αντιληφθήκαμε τέλος τις αρχές λειτουργίας για καθεμιά από τις παραπάνω δομές.