ΑΝΑΠΤΥΞΗ ΕΦΑΡΜΟΓΩΝ

ΣΕ ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΟ ΠΕΡΙΒΑΛΛΟΝ

ΑΛΓΟΡΙΘΜΟΙ

ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ «ΓΛΩΣΣΑ»

Επιμέλεια : Ι.Κονταξάκης

Μέρος

2

Δομές δεδομένων και Αλγόριθμοι

Περιεχόμενα

1. Δεδομένα -Δομές Δεδομένων

Ας υποθέσουμε ότι θέλουμε να υπολογίσουμε τον μισθό ενός υπαλλήλου. Αυτός εξαρτάται από το **βασικό** μισθό ,τα **χρόνια υπηρεσίας** που έχει από το **χρονοεπίδομα** που παίρνει ,από τον **αριθμό των παιδιών** του κλπ. Βλέπουμε ότι απαιτούνται κάποια ***στοιχεία (δεδομένα)*** πάνω στα οποία θα γίνει κάποια ***επεξεργασία*** για να υπολογιστεί ο μισθός του υπαλλήλου.

Με τον όρο ***Δεδομένα (data)*** θα εννοούμε κάθε παράσταση γεγονότων ή εννοιών που είναι κατάλληλη για επικοινωνία ή επεξεργασία από τον άνθρωπο ή αυτόματα μέσα (Η/Υ-Μηχανές)

Μπορεί να θεωρηθεί ότι ο αλγόριθμος είναι το μέσο που θα επεξεργαστεί τα δεδομένα για να παραχθούν τα αποτελέσματα (Πληροφορία) . Τα δεδομένα ενός προβλήματος αποθηκεύονται στον Η/Υ είτε στην Κύρια μνήμη (RAM) είτε σε αποθηκευτικά μέσα (περιφερειακές μονάδες Η/Υ) . Η αποθήκευση αυτή δεν γίνεται κατά ένα τυχαίο τρόπο αλλά συστηματικά , δηλαδή για την αποθήκευση των δεδομένων χρησιμοποιούμε μια ***δομή*** . Έτσι λοιπόν μπορούμε να πούμε ότι :

Δομή δεδομένων (data Structure) είναι ένα σύνολο αποθηκευμένων δεδομένων μαζί με ένα σύνολο επιτρεπτών λειτουργιών (πράξεων) πάνω σε αυτά

Η σχέση μεταξύ αλγορίθμων και δομών δεδομένων είναι ιδιαίτερα στενή. Ο Witth , δημιουργός της γλώσσας προγραμματισμού PASCAL είχε διατυπώσει τη χαρακτηριστική εξίσωση :

Αλγόριθμοι +Δομές δεδομένων =Προγράμματα

Μερικές βασικές λειτουργίες (πράξεις) πάνω στις δομές δεδομένων είναι:

Προσπέλαση (access) Πρόσβαση σε κάποιο δεδομένο με σκοπό να εξεταστεί ή να τροποποιηθεί

Εισαγωγή (Insertion) προσθήκη δεδομένων μέσα στην υπάρχουσα δομή

Διαγραφή (Deletion) Αφαίρεση δεδομένων μέσα από μια υπάρχουσα δομή

Αναζήτηση (searching) προσπέλαση των δεδομένων μέχρι να βρεθεί το αναζητούμενο δεδομένο που έχει κάποια χαρακτηριστική ιδιότητα , πάνω στην οποία βασίζεται η αναζήτηση.

Ταξινόμηση (Sorting) Διάταξη των δεδομένων κατά αύξουσα ή φθίνουσα σειρά ως προς κάποιο χαρακτηριστικό τους.

Αντιγραφή (Copying) πράξη που μας δίνει τη δυνατότητα να αντιγράψουμε ένα ή περισσότερα δεδομένα σε μια άλλη δομή

Συγχώνευση (merging) Ενέργεια (πράξη) κατά την οποία δύο ή περισσότερες δομές μπορούν να συνενωθούν σε μία νέα δομή

Διαχωρισμός (separation) Η αντίστροφη πράξη της συγχώνευσης , δηλαδή μία δομή να διασπάται σε δύο ή περισσότερες δομές .

ΠΑΡΑΤΗΡΗΣΗ
Δεν χρησιμοποιούνται όλες οι παραπάνω πράξεις (οκτώ) σε όλες τις δομές δεδομένων. Υπάρχουν δομές που είναι αποδοτικότερες σε κάποια από τις παραπάνω πράξεις , ενώ άλλες δομές είναι λιγότερο αποδοτικές . Αυτός είναι και ο λόγος που υπάρχουν πολλές δομές δεδομένων . Ανάλογα με τη εργασία που θέλουμε να κάνουμε επιλέγουμε να χρησιμοποιήσουμε και την κατάλληλη δομή.

2. Δυναμικές-Στατικές δομές δεδομένων

Οι δομές δεδομένων χωρίζονται σε δύο μεγάλες κατηγορίες :
A. Στατικές δομές δεδομένων (static)
B. Δυναμικές δομές δεδομένων (dynamic)
Στην πρώτη κατηγορία υπάγονται **, οι Πίνακες (array) , Εγγραφές (Records)** και άλλες ενώ στη δεύτερη τα **αρχεία (files) ,Στοίβες (stacks) ,Ουρές (Queues) κλπ**

Στις στατικές δομές δεδομένων , το μέγεθος τους είναι προκαθορισμένο από τη στιγμή της κατασκευής του προγράμματος και δεν είναι δυνατόν να αλλάξει κατά την εκτέλεση του προγράμματος. Τα στοιχεία μιας στατικής δομής δεδομένων αποθηκεύονται σε συνεχόμενες θέσεις της κύριας μνήμης.(RAM)

Στις δυναμικές δομές δεδομένων ,το μέγεθός τους δεν είναι προκαθορισμένο από την αρχή της κατασκευής του προγράμματος , αλλά προσδιορίζεται όταν εκτελείται το πρόγραμμα ανάλογα με τις ανάγκες του προβλήματος.. Τα στοιχεία μιας δυναμικής δομής δεδομένων συνήθως αποθηκεύονται σε βοηθητικές μνήμες και όχι σε συνεχόμενες θέσεις της κύριας μνήμης (RAM)

3. 1η Μορφή Δομής δεδομένων ΠΙΝΑΚΕΣ

Είναι χρήσιμο πολλές φορές να μπορούμε να ταξινομήσουμε διάφορες πληροφορίες , με τέτοιο τρόπο ώστε να μπορούμε να αναφερθούμε σ' αυτές χρησιμοποιώντας τη θέση που βρίσκονται . Ας υποθέσουμε για παράδειγμα ότι θέλουμε να παραστήσουμε το δυναμικό ενός σχολείου κατά φύλο και τάξη, θα μπορούσε να παρασταθεί όπως παρακάτω

Τάξεις	**Α**	**Β**	**Γ**

Αγόρια	122	85	72
Κορίτσια	111	82	54

Έτσι καταλαβαίνουμε ότι τα κορίτσια της Β τάξεως είναι 82 , τα αγόρια της Γ' τάξεως είναι 72 κ.οκ. Αν παραλείψουμε τις κεφαλίδες (πρώτη γραμμή πρώτη στήλη) θα προκύψει η παρακάτω ορθογώνια διάταξη αριθμών που θα λέγεται

122	85	72
111	82	54

Πίνακας (Array) . Ο πίνακας αυτός θα λέμε ότι έχει δύο γραμμές και τρεις στήλες (2χ3) . Για να αναφερθούμε σε ένα στοιχείο ενός πίνακα χρησιμοποιούμε το όνομα του πίνακα και μέσα σε αγκύλες δύο δείκτες που ο πρώτος μας δείχνει τη γραμμή και ο δεύτερος τη στήλη . Για παράδειγμα αν το όνομα του πίνακα είναι Sxoleio για να αναφερθούμε στα αγόρια της Β ' τάξεως γράφουμε sxoleio[1,2] ή στα κορίτσια της Γ' τάξεως γράφουμε sxoleio[2,3]

Δείτε ένα άλλο παράδειγμα πίνακα :
Ας υποθέσουμε ότι θέλουμε να τοποθετήσουμε σένα πίνακα την πρόοδο 10 μαθητών ενός τμήματος . Έτσι θα έχω τον πίνακα

Bathmoi=[12,18,16,9,17,20,19,12,13,16]

Τώρα γνωρίζω ότι Bathmoi[3]=16 , που σημαίνει ότι ο τρίτος μαθητής έχει βαθμό 16. Εδώ παρατηρώ ότι για να αναφερθώ σε ένα στοιχείο του πίνακα χρησιμοποιώ ένα δείκτη αφού ο πίνακας μου είναι μιας γραμμής (ή μιας στήλης)

Στη πληροφορική αντιμετωπίζουμε τους πίνακες ***σαν μεταβλητές με δείκτες***. Οι δείκτες αυτοί προσδιορίζουν τη θέση του στοιχείου μέσα στον πίνακα.

Με το όνομα του πίνακα περιγράφουμε ένα σύνολο από τιμές

Αν ο πίνακας έχει μια γραμμή (ή μια στήλη) απαιτείται ένας δείκτης για τον προσδιορισμό της θέσης του στοιχείου, ονομάζεται πίνακας **μιας διάστασης**

Αν ο πίνακας έχει γραμμές και στήλες απαιτούνται δύο δείκτες για τον προσδιορισμό της θέσης ενός στοιχείου. Υποχρεωτικά ο πρώτος προσδιορίζει τη γραμμή και ο δεύτερος τη στήλη. Ο πίνακας αυτός ονομάζεται πίνακας **δύο διαστάσεων**

Μπορούμε να κατασκευάσουμε πίνακα **τριών διαστάσεων** Για παράδειγμα φανταστείτε μια καρτελοθήκη γεμάτη με τις καρτέλες προόδου των μαθητών του τμήματός σας . Η κάθε καρτέλα είναι ένας πίνακας δύο διαστάσεων (γραμμές και στήλες) και όλες μαζί είναι ένας πίνακας τριών διαστάσεων. Για να αναφερθούμε σ'ένα στοιχείο ενός πίνακα τριών διαστάσεων πρέπει να χρησιμοποιήσουμε τρεις δείκτες . Για το παραπάνω παράδειγμα Karteles[i,j,k] . Ο Πρώτος δείκτης μας δείχνει τη γραμμή , ο δεύτερος τη στήλη και ο τρίτος τη καρτέλα.

Ένας πίνακας που έχει δύο διαστάσεις ίσες θα ονομάζεται **τετραγωνικό**ς πίνακας..

☼ Στη πράξη οι **στατικές δομές** δεδομένων υλοποιούνται κυρίως με τη δομή του πίνακα που υποστηρίζονται από όλες τις γλώσσες προγραμματισμού .

Το μέγεθος ενός πίνακα είναι προκαθορισμένο από τη στιγμή του προγραμματισμού και όχι κατά την εκτέλεση του προγράμματος.

Τα στοιχεία της δομής του πίνακα αποθηκεύονται σε συνεχόμενες θέσεις της βοηθητικής μνήμης.

Κατά την εκτέλεση ενός αλγόριθμου επεξεργαζόμαστε συνήθως ένα στοιχείο του πίνακα κάθε φορά. Στις εντολές λοιπόν των αλγορίθμων εμφανίζονται, εν γένει, τα στοιχεία του πίνακα και όχι όλος ο πίνακας

ΔΙΑΧΕΙΡΙΣΗ ΠΙΝΑΚΩΝ

Καταχώριση στοιχείων μονοδιάστατου πίνακα στη μνήμη Η/Υ

Αν Θέλουμε να καταχωρίσουμε στη μνήμη του Η/Υ τα εισιτήρια που κόπηκαν σε θέατρο για ένα μήνα , μπορούμε να χρησιμοποιήσουμε 30 μεταβλητές και να εκχωρήσουμε την κάθε μία μεταβλητή τα εισιτήρια μίας μέρας. Αυτό όμως στους αλγορίθμους με τους πίνακες μπορεί να γίνει με τον παρακάτω τρόπο:

Για παράδειγμα έστω ότι ο πίνακας των εισιτηρίων
EIS=[205,197,89,231,148,150,176,122,134,145] μπορεί να καταχωριστεί στη μνήμη του Η/Υ δίνοντας τα στοιχεία από το πληκτρολόγιο με το παρακάτω τμήμα αλγορίθμου :

```
Για χ από 1 μέχρι 30
      Διάβασε  EIS[x]
Τέλος_επανάληψης
```

Ανάγνωση και εκτύπωση στοιχείων από τη μνήμη του Η/Υ

Αν θέλουμε να εκτυπώσουμε τα στοιχεία του παραπάνω πίνακα μπορούμε να γράψουμε το παρακάτω τμήμα αλγορίθμου :

```
Για χ από 1 μέχρι 30
      τύπωσε  EIS[x]
Τέλος_επανάληψης
```

Καταχώρηση στοιχείων διδιάστατου πίνακα στη μνήμη του Η/Υ

Ο παρακάτω πίνακας περιέχει τους βαθμούς τριών μαθητών .

	Α ΤΡΙΜ	Β ΤΡΙΜ	Γ ΤΡΙΜ	ΓΡΑΠΤΑ
Μαθητής 1	12	16	17	15
Μαθητής 2	16	17	12	14
Μαθητής 3	11	15	16	17

Αν θέλουμε να καταχωρίσουμε στη μνήμη του Η/Υ τους βαθμούς αυτούς δίνοντας τα δεδομένα από το πληκτρολόγιο αρκεί να γράψουμε το παρακάτω τμήμα αλγορίθμου:

```
για Ι από 1 μέχρι 3
     Για j από 1 μέχρι 4
          Διάβασε  Β[Ι,j]
     Τελος_Επανάληψης
Τέλος_Επανάληψης
```

Ανάγνωση και εκτύπωση στοιχείων από τη μνήμη του Η/Υ

```
για Ι από 1 μέχρι 3
    Για j από 1 μέχρι 4
        τύπωσε  B[I,j]
    Τελος_Επανάληψης
Τέλος_Επανάληψης
```

ΠΑΡΑΤΗΡΗΣΕΙΣ

☼ Δύο Πίνακες Α και Β που έχουν την ίδια διάσταση μχν θα λέμε ότι είναι ίσοι αν και μόνο αν τα αντίστοιχα στοιχεία τους ταυτίζονται. Για παράδειγμα A[I,j]=B[I,j] για κάθε i=1,2,3,4,5, ….μ και j=1,2,3,4,……ν

☼ Η τιμή κάθε στοιχείου του πίνακα αποθηκεύεται σένα τμήμα της μνήμης του Η/Υ , που δεσμεύεται ειδικά γι' αυτό .Οι τιμές των στοιχείων του πίνακα καταχωρούνται σε γειτονικές θέσεις στη μνήμη και δεν είναι δυνατόν να υπερβούμε τα όρια της δήλωσης του πίνακα . Δηλαδή το μέγεθος του πίνακα προκαθορίζεται στην αρχή του αλγορίθμου.

☼ Το είδος των στοιχείων ενός πίνακα είναι καθορισμένο από την αρχή . Αυτό σημαίνει ότι ένας πίνακας περιέχει ενός μόνο είδους στοιχεία, που μπορεί να είναι Ακέραιοι , πραγματικοί, αλφαριθμητικά (string) ,λογικές τιμές (Αληθής ή ψευδής)

☼ Συνηθίζεται αφού το μέγεθος είναι προκαθορισμένο , συνηθίζεται στη διαχείριση ενός πίνακα να χρησιμοποιούμε την επαναληπτική δομή Για ….. τέλος_επανάληψης

☼ Έτσι μπορούμε να πούμε ότι οι πίνακες έχουν τα παρακάτω μειονεκτήματα και πλεονεκτήματα

Μειονεκτήματα

Σταθερό μέγεθος , άρα σχετική δυσκολία στον προγραμματισμό
Δέσμευση μνήμης Η/Υ καθ' όλη τη διάρκεια εκτέλεσης του προγράμματος. Άρα χρησιμοποιούνται σε μικρό πλήθος δεδομένων

Πλεονεκτήματα

Σχετικά εύκολη χρήση των πινάκων
Με το ίδιο όνομα αναφερόμαστε σε όλα τα στοιχεία του πίνακα, χρησιμοποιώντας δείκτες αυτό σημαίνει ότι δεν χάνονται τα δεδομένα κατά την εκτέλεση του προγράμματος.

ΤΥΠΙΚΕΣ ΕΠΕΞΕΡΓΑΣΙΕΣ ΠΙΝΑΚΩΝ

Υπολογισμός αθροισμάτων στοιχείων πίνακα

Για μονοδιάστατο πίνακα

```
sum←0
Για χ από 1 μέχρι N
   sum←sum +pin[x]
τέλος_επανάληψης
```

Για πίνακα δύο διαστάσεων

```
sum←0
Για i από 1 μέχρι N
    Για j από 1 μέχρι M
```

```
                sum←sum +pin[I,j]
    τέλος_επανάληψης
τέλος_επανάληψης
```

Εύρεση του μέγιστου και του ελάχιστου στοιχείων πίνακα

Παράδειγμα 1 Εύρεση του μικρότερου στοιχείου μονοδιάστατου πίνακα
Δίνεται ένας μονοδιάστατος πίνακας table[100]. Να σχεδιαστεί αλγόριθμος που να βρίσκει το μικρότερο στοιχείο του.

```
Αλγόριθμος Min
Δεδομένα // table //
min ← table[1]
Για I από 2 μέχρι 100
   Αν table[i]<min  τότε
      min ←table[i]
   τέλος_αν
τέλος_επανάληψης
Αποτελέσματα // min//
Τέλος Min
```

Παράδειγμα 2 Εύρεση του μικρότερου στοιχείου διδιάστατου πίνακα

```
Αλγόριθμος Min2
Δεδομένα // Table//
min← table[1]
Για i από 2 μέχρι N
    Για j από 1 μέχρι M
        Αν table[I,j]<min τότε
            min← table[I,j]
        Τέλος_αν
    τέλος_επανάληψης
τέλος_επανάληψης
Αποτελέσματα // min //
Τέλος  Min2
```

Θα μπορούσαμε να κρατήσουμε τη θέση που βρίσκεται το μικρότερο στοιχείο του πίνακα σε μια μεταβλητή Position έτσι ο παραπάνω αλγόριθμος θα γινόταν :

```
Αλγόριθμος Min_Θέση
Δεδομένα // table //
min ← table[1]
position ← 1
Για I από 2 μέχρι 100
   Αν table[i]<min  τότε
      min ←table[i]
      position ←  i
   τέλος_αν
```

```
τέλος_επανάληψης
Αποτελέσματα // min , position//
Τέλος Min_Θέση
```

Με την ίδια λογική θα μπορούσαμε να κρατούσαμε τη θέση του μικρότερου στοιχείου ενός διδιάστατου πίνακα . Θα χρειαζόμασταν όμως δύο μεταβλητές μία μεταβλητή για να κρατήσουμε τη γραμμή (Gr) και μία μεταβλητή για να κρατήσουμε τη στήλη (St). Έτσι θα είχαμε τον παρακάτω αλγόριθμό :

```
Αλγόριθμος Min2_Θέση
Δεδομένα // Table//
min← table[1,1]
Gr ← 1
St ← 1
Για i από 2 μέχρι N
    Για j από 1 μέχρι M
        Αν table[I,j]<min τότε
                min← table[I,j]
                Gr ← i
                St ← j
        Τέλος_αν
    τέλος_επανάληψης
τέλος_επανάληψης
Αποτελέσματα // min ,Gr , St//
Τέλος Min2
```

Αναζήτηση στοιχείων πίνακα

Η Αναζήτηση στοιχείου σε κάποιο πίνακα είναι ένα θέμα ιδιαίτερα ενδιαφέρον αφού σε πάρα πολλές εφαρμογές ,απαιτείται να ψάξουμε να βρούμε μια πληροφορία σένα πίνακα. Υπάρχουν πολλές μέθοδοι αναζήτησης (searching) που οι διαφορές οφείλονται στο αν είναι ταξινομημένος ο πίνακας ή όχι.

Η πιο απλή μορφή αναζήτησης σένα πίνακα είναι η σειριακή ή γραμμική μέθοδος που θα περιγράψουμε παρακάτω : Με αυτή τη μέθοδο ψάχνουμε το αναζητούμενο στοιχείο σειριακά –ακολουθιακά ξεκινώντας από το πρώτο στοιχείο του πίνακα. Ο αλγόριθμος συγκρίνει κάθε στοιχείο του πίνακα με την αναζητούμενη τιμή. Έτσι αν εντοπιστεί το αναζητούμενο στοιχείο , η μέθοδος σταματά διαφορετικά ο αλγόριθμος μας δίνει κατάλληλο μήνυμα.

Αυτή η μέθοδος είναι αρκετά χρονοβόρα και εφαρμόζεται σε πίνακες μη ταξινομημένους και με μικρό μέγεθος γενικά.

1. Σειριακή αναζήτηση σε μη ταξινομημένο πίνακα , κάθε στοιχείο του πίνακα είναι μοναδικό.(μία φορά)

```
Αλγόριθμος Σειριακή_1
        Δεδομένα / / N, A, key/ /
        done ← ψευδής
```

```
        position ← 0
        i ← 1
        Όσο (done = ψευδής) και (i < = N) επανάλαβε
                Αν A[i] = key τότε
                        done ← αληθής
                        position ← i
                        αλλιώς
                                i ← i + 1
                Τέλος_αν
        Τέλος_επανάληψης
        Αποτελέσματα / / done, position / /
Τέλος Σειριακή_1
```

Στην ανάπτυξη αυτής της μεθόδου χρησιμοποιήθηκε μια μεταβλητή (done) που παίρνει την τιμή «Ψευδής» μόλις βρεθεί το αναζητούμενο στοιχείο παίρνει την τιμή «Αληθής» και με την εντολή position←I καταγράφεται η θέση του στοιχείου. Και σταματά η διαδικασία. Αν δεν βρεθεί τότε η μεταβλητή done θα συνεχίσει να έχει την τιμή «Ψευδής» και η μεταβλητή position την τιμή 0

2. Σειριακή αναζήτηση σε μη ταξινομημένο πίνακα , κάποια στοιχεία του πίνακα εμφανίζονται παραπάνω από μία φορά

Αν κάποιο στοιχείο εμφανίζεται στον πίνακα παραπάνω από μία φορά , τότε ο αλγόριθμος πρέπει να τροποποιηθεί η μεταβλητή done είναι περιττή. Σ' αυτή τη περίπτωση η αναζήτηση συνεχίζεται μέχρι το τέλος του πίνακα αφού η μοναδική συνθήκη θα είναι i<=N .

```
Αλγόριθμος Σειριακή_2
Δεδομένα / / N, A, key / /
position ← 0
i ← 1
Όσο i < = N επανάλαβε
        Αν A[i] = key τότε
                position ← i
        αλλιώς
                i ← i + 1
        Τέλος_αν
Τέλος_επανάληψης
Αποτελέσματα / / done, position / /
Τέλος Σειριακή_2
```

Προφανώς αν κάποιο στοιχείο του πίνακα επαναλαμβάνεται για παράδειγμα τρεις φορές θα μας δώσει τη θέση του τρίτου

3. Σειριακή αναζήτηση σε ταξινομημένο πίνακα

```
Αλγόριθμος Σειριακή_3
Δεδομένα // N, Table ,key //
done← Ψευδής
position ←0
i←1
```

```
Οσο (done=Ψευδής) και (i<=N) και (key>table[i]) επανάλαβε
  Αν table[i]=key τότε
            done ←Αληθής
             position←i
         αλλιώς
            i←i+1
    Τέλος_αν
Τέλος_επανάληψης
Αποτελέσματα // done,position//
Τέλος Σειριακή_3
```

Στο παραπάνω αλγόριθμο η επανάληψη σταματά όταν 1) Το στοιχείο Βρεθεί 2) Όταν εξαντληθεί ο πίνακας και το στοιχείο δεν βρέθηκε και 3) Όταν το στοιχείο που αναζητάμε είναι μεγαλύτερο από κάποιο στοιχείο του πίνακα.

Ταξινόμηση στοιχείων πίνακα

Η διάταξη των στοιχείων ενός πίνακα κατά αύξουσα ή φθίνουσα σειρά (ταξινόμηση-Sorting) είναι μια πολύ σπουδαία εργασία και πάρα πολύ χρήσιμη σε ζητήματα όχι μόνο της καθημερινής ζωής αλλά και στα περισσότερα θέματα πληροφορικής. Μπορούμε να ταξινομήσουμε πίνακες που περιέχουν αριθμούς , αλλά και αλφαριθμητικά (π.χ Ονόματα και επίθετα)

Η Ταξινόμηση στοιχείων ενός πίνακα μας κάνει την αναζήτηση στοιχείων πίνακα ευκολότερη και γρηγορότερη.
Ταξινόμηση στοιχείων πίνακα είναι η διαδικασία της διάταξης των στοιχείων σε αύξουσα ή φθίνουσα σειρά.
Υπάρχουν πολλοί μέθοδοι ταξινόμησης και η κάθε μια έχει τα πλεονεκτήματα ή τα μειονεκτήματά της . Εμείς θα ασχοληθούμε παρακάτω με τη με την μέθοδο ***«Ευθείας ανταλλαγής»*** ή μέθοδος της ***«Φυσαλίδας»*** (Bubble sort) Η μεγάλη δημοτικότητά της προέρχεται από το παράξενο όνομά της , αλλά και από την απλότητά της . Έχει το χαρακτηριστικό της απλότητας δεν θεωρείται όμως αρκετά γρήγορη , ιδιαίτερα σε μεγάλους πίνακες.

Η μέθοδος αυτή στηρίζεται στην ιδέα ότι μπορούμε να συγκρίνουμε δύο γειτονικά στοιχεία του πίνακα και αν το πρώτο έχει χαμηλότερη αξία από το δεύτερο, τα εναλλάσσουμε ,διαφορετικά τα αφήνουμε στη θέση τους.

Πιο συγκεκριμένα :
Στο πρώτο σάρωμα του πίνακα (π.χ Α[ν]) συγκρίνουμε το Α[10] με Α[9] στοιχείο και αν Α[10]>Α[9] τότε τα αφήνουμε στη θέση τους , αν όμως Α[10] <Α[9] τότε τους αλλάζουμε τη θέση τους. Αυτό συνεχίζεται για τα στοιχεία Α[9] και Α[8] κ.οκ . Η διαδικασία σταματά όταν δεν έχω να κάνω άλλες εναλλαγές .
Εφαρμόζουμε τα παραπάνω στον πίνακα Α=[18 15 23 12 5 7 13 28 14 10]

Αρχική Τιμή	ΣΑΡΩΜΑΤΑ ΠΙΝΑΚΑ							
	1	2	3	4	5	6	7	8

A[1]	**18**	5	5	5	5	5	5	5	5
A[2]	**15**	**18**	7	7	7	7	7	7	7
A[3]	23	**15**	**18**	12	10	10	10	10	10
A[4]	12	23	**15**	**18**	12	12	12	12	12
A[5]	5	12	23	**15**	**18**	13	13	13	13
A[6]	7	7	12	23	**15**	**18**	14	14	14
A[7]	13	10	10	10	23	**15**	**18**	15	15
A[8]	28	13	13	13	13	23	**15**	**18**	**18**
A[9]	14	28	14	14	14	14	23	23	23
A[10]	10	14	28	28	28	28	28	28	28

ΠΑΡΑΤΗΡΗΣΕΙΣ

☼ Όπως φαίνεται από τον παραπάνω πίνακα (παρατηρήστε τους έντονα γραμμένους αριθμούς) οι τιμές με τη μεγαλύτερη αξία (βαρύτερες) τείνουν να κατεβαίνουν προς τα κάτω , Ενώ οι τιμές με τη μικρότερη αξία τείνουν να ανεβαίνουν προς τα επάνω σαν φυσαλίδες , Έτσι έχει πάρει και το όνομά της.

☼ Η μέθοδος σταματά όταν δεν υπάρξει καμία εναλλαγή .

Ο παρακάτω αλγόριθμος μας υλοποιεί την παραπάνω μέθοδο.

```
Αλγόριθμος Bubble_Sort
plithos← 10
! Διάβασμα του πίνακα
Για χ από 1 μέχρι plithos
  Διάβασε A[χ]
Τέλος_επανάληψης
! Αρχή μεθόδου
Για I από 2 μέχρι plithos
        για j από plithos μέχρι I με_βήμα -1
                Αν A[j-1]> A[j] τοτε
                        αντιμετάθεσε A[j-1] , A[j]
                τέλος_αν
        τέλος_επανάληψης
Τέλος_επανάληψης
!Αποτελέσματα
Για χ από 1 μέχρι plithos
        τύπωσε A[χ]
Τέλος επανάληψης
Τέλος   Bubble_Sort
```

☼ Στον παραπάνω πίνακα ταξινομήσαμε ακεραίους. Ο ακέραιος τύπος επιλέχτηκε χάριν απλότητας.. Θα μπορούσαμε να ταξινομήσουμε πραγματικούς τύπο αριθμών ή και τύπο χαρακτήρα (Γράμματα- Λέξεις -string)

☼ Η εντολή «**αντιμετάθεσε**» στον παραπάνω αλγόριθμο θα μπορούσε να υλοποιηθεί χρησιμοποιώντας μια βοηθητική μεταβλητή (temp)

```
temp← A[j-1]
A[j-1] ←A[j]
A[j]←temp
```

Παρατήρηση

Σαν απάντηση στην άσκηση του Τετραδίου του μαθητή (ΔΤ2 σελίδα 33)

Ο Αλγόριθμος της φυσσαλίδας δεν είναι αρκετά «έξυπνος» ώστε να διαπιστώνει στην αρχή ή στο τέλος αν ο πίνακας είναι ταξινομημένος , ώστε να αποφεύγονται τα περιττά περάσματα (που σε συνολικό αριθμό είναι Ν-1) .

Για τη λύση του παραπάνω προβλήματος θα χρησιμοποιηθεί μια λογική μεταβλητή (Flag) , που **πριν** από κάθε πέρασμα **αρχικοποιείται ως ψευδής** και αλλάζει ως αληθής αν σε κάποιο πέρασμα γίνει έστω και μία ανταλλαγή . Έτσι αν σε κάποιο πέρασμα δεν γίνει καμία ανταλλαγή , τότε η flag παραμένει ψευδής και αυτομάτως τελειώνει ο αλγόριθμος,

```
Αλγόριθμος φυσσαλίδα_2
Δεδομένα // table//
Αρχή επανάληψης
Flag← ψευδής
   Για ι από 1 μέχρι n-1
      Αν table[i+1]<table[i]  τότε
                  αντιμετάθεσε table[i+1]  ,table[i]
                  flag ← αληθής
      Τέλος_αν
   Τέλος_επανάληψης
Μέχρις_ότου flag=ψευδής
Αποτελέσματα // table//
τέλος Φυσσαλίδας_2
```

Συγχώνευση δύο πινάκων

Η συγχώνευση είναι μια ακόμα βασική εργασία σε πίνακες. Ο σκοπός της συγχώνευσης δύο ή περισσοτέρων πινάκων , είναι η δημιουργία ενός άλλου πίνακα με διπλάσιο αριθμό στηλών.

Πρόβλημα 1 : Δίνονται δύο πίνακες γραμμή Α και Β

$A=[a_1 \quad a_2 \quad a_3 \quad a_4 \quad a_5 \quad a_6]$ *και* $B=[b_1 \quad b_2 \quad b_3 \quad b_4 \quad b_5 \quad b_6]$

Ζητείται να δημιουργηθεί τρίτος πίνακας

$\Gamma=[\alpha_1 \quad \alpha_2 \quad \alpha_3 \quad \alpha_4 \quad \alpha_5 \quad \alpha_6 \quad b_1 \quad b_2 \quad b_3 \quad b_4 \quad b_5 \quad b_6]$

Λύση του προβλήματος 1

```
Αλγόριθμος Συγχώνευση_1
Δεδομένα / / Α, Β / /
Για i από 1 μέχρι 6
                  Γ[ i] ← Α[ i]
                  Γ[ i + 6] ← Β[ i]
      Τέλος_επανάληψης
Τέλος_επανάληψης
Αποτελέσματα / / Γ / /
Τέλος Συγχώνευση_1
```

Πρόβλημα 2 : **Να γραφεί αλγόριθμος που να συγχωνεύει (συνενώνει) δύο δεδομένους πίνακες Α και Β διαστάσεων 5×5, σε έναν τρίτο πίνακα διαστάσεων 5×10.**

$$A=\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{bmatrix} \qquad B=\begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} & b_{15} \\ b_{21} & b_{22} & b_{23} & b_{24} & b_{25} \\ b_{31} & b_{32} & b_{33} & b_{34} & b_{35} \\ b_{41} & b_{42} & b_{43} & b_{44} & b_{45} \\ b_{51} & b_{52} & b_{53} & b_{54} & b_{55} \end{bmatrix}$$

$$A=\left[\begin{array}{ccccc:ccccc} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & b_{11} & b_{12} & b_{13} & b_{14} & b_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & b_{21} & b_{22} & b_{23} & b_{24} & b_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & b_{31} & b_{32} & b_{33} & b_{34} & b_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & b_{41} & b_{42} & b_{43} & b_{44} & b_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & b_{51} & b_{52} & b_{53} & b_{54} & b_{55} \end{array}\right]$$

Παρατήρηση: Για τη συγκεκριμένη άσκηση υπάρχουν πολλές παραλλαγές, π.χ. ο πίνακας Β να έχει διαστάσεις 5×7, ή να δίνεται ο πίνακας Γ και να ζητείται ο διαχωρισμός του σε δύο άλλους πίνακες κατάλληλων διαστάσεων, κ.ο.κ.

Αλγόριθμος Συγχώνευση_2
Δεδομένα / / Α, Β / /
Για i **από** 1 **μέχρι** 5
    Για j **από** 1 **μέχρι** 5
        Γ[i, j] ← A[i, j]
        Γ[i, j + 5] ← B[i, j]
    Τέλος_επανάληψης
Τέλος_επανάληψης
Αποτελέσματα / / Γ / /
Τέλος Συγχώνευση_2

Τώρα τίθεται το ερώτημα : Μπορούμε να συνενώσουμε **δύο ταξινομημένους πίνακες. Και δημιουργήσουμε ένα άλλο πίνακα που να είναι και αυτός ταξινομημένος. ;**

Παρακάτω θα δοθεί ο αλγόριθμος συγχώνευσης δύο ταξινομημένων πινάκων και η δημιουργία ενός τρίτου πίνακα επίσης ταξινομημένου. Θα θεωρήσουμε σαν δεδομένα τους πίνακες Α[ ] και Β[ ] και το πλήθος των στοιχείων τους m και n αντίστοιχα.

Η μέθοδος στηρίζεται στην ιδέα ότι : Συγκρίνουμε Α[1]<Β[1] αν ναι τοποθετούμε στον νέο πίνακα Γ[1] ←Α[1] αν όχι Γ[1]←Β[1] και η διαδικασία συνεχίζεται μέχρι να τελειώσει ο ένας πίνακας από τους δεδομένους. Η συνέχιση της διαδικασίας απαιτεί να χρησιμοποιήσουμε τρεις δείκτες I, J, K για να παρακολουθούμε σε ποιο σημείο βρισκόμαστε μέσα σε κάθε πίνακα

.Αφού τελειώσει ο ένας πίνακας αντιγράφουμε τα υπόλοιπα στοιχεία του άλλου στον τρίτο πίνακα Γ[ ]

```
Αλγόριθμος Συνένωση
Δεδομένα // Α[ ] ,Β[ ], n ,m//
DA ←1
DB ←1
DΓ ←1
¨Οσο (DA<=n) ΚΑΙ (DB<=m) επανέλαβε
   Αν Α[DA]<Β[DB] τότε
         Γ[DΓ]←Α[DA]
          DA←DA+1
          αλλιώς
          Γ[DΓ]←Β[DB]
           DB←DB+1
   Τέλος_αν
    DΓ←DΓ+1
Τέλος_επανάληψης
Αν DA>n τότε
      Για t από DB μέχρι m
            Γ[DΓ+t-DB] ← Β[t]
       τέλος_επανάληψης
      αλλιώς
      Για t από DA  μέχρι n
               Γ[DΓ+t-DA] ← Α[t]
      Τέλος_επανάληψης
Τέλος_αν
Αποτελέσματα // Γ[]//
Τέλος Συγχώνευση
```

4. 2η Μορφή δομής δεδομένων ΕΓΓΡΑΦΗ (Record)

Ένα σημαντικό μειονέκτημα των πινάκων είναι ότι υποχρεωτικά όλα του τα στοιχεία πρέπει **να είναι του ίδιου τύπου** (Ακέραιος, πραγματικός , τύπος χαρακτήρα κ.λ.π) . Το πρόβλημα αυτό λύνεται με την 2η μορφή δεδομένων **«Εγγραφή-Record»** Θα παρουσιάσουμε τη νέα δομή δεδομένων με ένα παράδειγμα.

Πολλές φορές στο σχολείο σας χρειάστηκε να φτιάξετε καταστάσεις που να περιέχουν το σύνολο των μαθητών του τμήματος και σε διπλανές στήλες να βάζετε ημερομηνίες ή κάποιο ποσό χρημάτων, όπως

ΑΡ.ΜΗΤΡΩΟΥ	ΕΠΙΘΕΤΟ	ΟΝΟΜΑ	ΗΜΕΡΟΜΗΝΙΑ	ΠΟΣΟ €
234	ΣΗΦΑΚΑΚΗΣ	ΝΙΚΟΣ	12-11-04	120
286	ΚΡΗΤΙΚΑΚΗΣ	ΠΟΛΟΣ	10-11-04	110
……..				

…….				
286	ΝΙΚΟΛΟΥΔΑΚΗΣ	ΣΙΦΗΣ	10-10-04	89

Παρατηρούμε ότι κάθε γραμμή είναι ένα σύνολο πληροφοριών που αναφέρονται στην οντότητα μαθητής. Θα λέμε ότι κάθε γραμμή που αποτελεί μια ***εγγραφή (Record)*** . Η κάθε στήλη μπορεί να είναι διαφορετικού τύπου και ονομάζεται ***πεδίο (field)*** της εγγραφής. Έτσι η πρώτη γραμμή (πρώτη εγγραφή) περιγράφει την οντότητα «ΣΗΦΑΚΑΚΗΣ ΝΙΚΟΣ» και αποτελείται από τα πεδία 1) ΑΡ.ΜΗΤΡΩΟΥ 2) ΕΠΙΘΕΤΟ 3) ΟΝΟΜΑ 4)ΗΜΕΡΟΜΗΝΙΑ 5)ΠΟΣΟ . Η διάταξη αυτή της εγγραφής ονομάζεται ***γραμμογράφηση*** της εγγραφής
Γενικά λοιπόν μπορούμε να πούμε ότι :

Εγγραφή είναι ένα σύνολο αλληλοσυσχετιζόμενων πεδίων που περιγράφουν μια οντότητα.

Ένα σύνολο εγγραφών λέμε ότι δημιουργεί μια νέα δομή δεδομένων που ονομάζεται ***Αρχείο (Δυναμική δομή δεδομένων)***

☼ Σε ένα αρχείο δεν είναι προκαθορισμένο το μέγεθός του κατά τη στιγμή του προγραμματισμού, τα στοιχεία αυτής της δομής δεν αποθηκεύονται σε συνεχόμενες θέσεις της βοηθητικής μνήμης.

5. 3η Μορφή δομής Δεδομένων ΣΤΟΙΒΑ (Stack)

(Από Βιβλίο Σελίδα 59 μέχρι 60)

6. 4η Μορφή δομής δεδομένων ΟΥΡΑ (Queue)

(Από βιβλίο σελίδα 60 μέχρι 63)

ΑΣΚΗΣΕΙΣ ΣΤΟΥΣ ΠΙΝΑΚΕΣ

ΑΣΚΗΣΗ 1 Να γραφεί αλγόριθμος ο οποίος να διαβάζει ένα μονοδιάστατο πίνακα ακέραιων αριθμών Α με Ν θέσεις και υπολογίζει και να τυπώνει το μέσο όρο, το άθροισμα και το γινόμενο των στοιχείων του πίνακα Α.

ΑΣΚΗΣΗ 2 Να γραφεί αλγόριθμος ο οποίος να διαβάζει ένα μονοδιάστατο πίνακα ακέραιων αριθμών Α με 10 θέσεις και υπολογίζει και να τυπώνει το μέγιστο και το ελάχιστο στοιχείο του πίνακα Α και τις θέσεις τους.

ΑΣΚΗΣΗ 3 Να γραφεί αλγόριθμος ο οποίος να διαβάζει ένα μονοδιάστατο πίνακα ακέραιων αριθμών Α με Ν θέσεις και υπολογίζει και να τυπώνει το μέσο όρο, και πόσοι από τους αριθμούς είναι μεγαλύτεροι από τον πιο πάνω μέσο όρο.

ΑΣΚΗΣΗ 4 Να γραφεί αλγόριθμος ο οποίος να δημιουργεί και να εμφανίζει μονοδιάστατο πίνακα Α 100 θέσεων σύμφωνα με την σχέση:

$$A_i = \begin{cases} (2*x)^i, & \text{όταν } x<10 \\ i^x, & \text{διαφορετικά} \end{cases}$$

ΑΣΚΗΣΗ 5 Να γραφεί αλγόριθμος ο οποίος να διαβάζει δύο μονοδιάστατους πίνακες με όνομα Α και Β 5 θέσεων και να εμφανίζει το πίνακα C ο οποίος είναι το άθροισμα των Α και Β.

ΑΣΚΗΣΗ 6 Να γραφεί αλγόριθμος ο οποίος να διαβάζει ένα πίνακα ακέραιων αριθμών Α, δύο διαστάσεων Ν x Μ και να υπολογίζει και να τυπώνει το μέσο όρο, το άθροισμα και το γινόμενο των στοιχείων του πίνακα Α.

ΑΣΚΗΣΗ 7 Να γραφεί αλγόριθμος ο οποίος να διαβάζει ένα πίνακα ακέραιων αριθμών Α, δύο διαστάσεων Ν x Μ και να υπολογίζει και να τυπώνει το μέγιστο και ελάχιστο στοιχείο του πίνακα καθώς και τις θέσεις τους στο πίνακα.

ΑΣΚΗΣΗ 8 Να γραφεί αλγόριθμος ο οποίος να διαβάζει ένα πίνακα ακέραιων αριθμών Α, δύο διαστάσεων 3 x 3 και να υπολογίζει και να τυπώνει α)το μέγιστο στοιχείο (και τη θέση του) της 3ης γραμμής β)το ελάχιστο στοιχείο (και τη θέση του) της 2ης στήλης γ)το άθροισμα των στοιχείων της 1ης γραμμής δ)το γινόμενο των στοιχείων της 3ης στήλης και ε)το άθροισμα των στοιχείων της κυρίας διαγωνίου.

$$\begin{matrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{matrix}$$

Τα στοιχεία της κυρίας διαγωνίου είναι τα: α_{11}, α_{22} και α_{33}.

ΑΣΚΗΣΗ 9 Να γραφεί αλγόριθμος ο οποίος να διαβάζει ένα πίνακα ακέραιων αριθμών Α, δύο διαστάσεων 3 x 3 και να υπολογίζει και να τυπώνει το άθροισμα των στοιχείων της δευτερεύουσας διαγωνίου.

$$\begin{matrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{matrix}$$

Τα στοιχεία της δευτερεύουσας διαγωνίου είναι τα: α_{13}, α_{22} και α_{31}.

ΑΣΚΗΣΗ 10 Να γραφεί αλγόριθμος ο οποίος να κατασκευάζει δύο μονοδιάστατους πίνακες με όνομα ODD και EVEN αντίστοιχα ως εξής. Ο πρώτος να περιέχει τους περιττούς αριθμούς από το 1 ως το 100 και ο δεύτερος τους άρτιους αριθμούς

ΛΥΣΕΙΣ ΤΩΝ ΑΣΚΗΣΕΩΝ-αλγόριθμοι

ΛΥΣΕΙΣ ΤΩΝ ΑΣΚΗΣΕΩΝ-ΔΟΜΗ ΑΚΟΛΟΥΘΙΑΣ

1. **Αλγόριθμος** Παράσταση_Ε1
! Εισαγωγή δεδομένων
Α← 2
Β ← 1
C ← 4
! Υπολογισμοί
Ε1 ← (Β+C)^Α/(C-Α)
! Αποτελέσματα
Τύπωσε Ε1
Τέλος Παράσταση_Ε1

2.**Αλγόριθμος** Υπολογισμοί
! Εισαγωγή δεδομένων
Διάβασε Α
Διάβασε Β
Διάβασε C
Διάβασε D

```
    ! Υπολογισμοί
      SUM ← A+B+C+D
      PRODUCT ← A*B*C*D
      AVER ← SUM/4
      ! Αποτελέσματα
       Εκτύπωσε SUM
        Εκτύπωσε PRODUCT
        Εκτύπωσε AVER
  Τέλος Υπολογισμοί
```

ΛΥΣΕΙΣ ΤΩΝ ΑΣΚΗΣΕΩΝ -ΔΟΜΗ ΕΠΙΛΟΓΗΣ

1)
```
Αλγόριθμος Αριθμός_Θετικός1
    Διάβασε Ar
    Αν Ar > 0 τότε
          sxolio ← " Ο Αριθμός είναι θετικός»
     Τέλος_αν
      Τύπωσε sxolio
Τέλος Αριθμός _Θετικός1
```

2)
```
Αλγόριθμος Αριθμός_Θετικός2
    Διάβασε Ar
    Αν Ar > 0 τότε
          sxolio ← " Ο Αριθμός είναι θετικός»
         αλλιώς
          sxolio ← " Ο Αριθμός είναι αρνητικός»
     Τέλος_αν
      Τύπωσε sxolio
Τέλος Αριθμός _Θετικός2
```

3)
```
  Αλγόριθμος Βαθμός_Μαθητή
    ! Εισαγωγή δεδομένων
    Γράψε «Δώσε ένα γράμμα Α , Β, C , D»
    Διάβασε Xaract
    ! Επεξεργασία δεδομένων
    Επίλεξε Xaract
        Περίπτωση "A"
              Bathmos ← "ΑΡΙΣΤΑ»
        Περίπτωση  "B"
              Bathmos  ← "ΚΑΛΑ»
       Περίπτωση  "C"
              Bathmos  ← "ΜΕΤΡΙΑ»
        Περίπτωση  "D"
              Bathmos  ← "ΑΠΕΤΥΧΕ»
        Περίπτωση αλλιώς
               Bathmos  ← "ΕΔΩΣΕΣ ΛΑΘΟΣ ΧΑΡΑΚΤΗΡΑ»
      Τέλος_επιλογών
      ! Αποτελέσματα
      Τύπωσε Bathmos
   Τέλος Βαθμός_Μαθητή
```

ΛΥΣΕΙΣ ΑΣΚΗΣΕΩΝ ΣΤΗΝ ΕΜΦΩΛΕΥΜΕΝΗ ΕΠΙΛΟΓΗ

ΑΣΚΗΣΗ 1

```
Αλγόριθμος Μηνιαίος_Λογαριασμός
Διάβασε ΧΑ
Διάβασε ΣΔ
Διάβασε ΚΝ
ΠΑΓΙΟ← 0
Αν ΚΝ>0 τότε
        Αν ΚΝ<=20 τότε
                ΚΟΣΤΟΣ ← (ΚΝ*60)
                Αλλιώς
                    ΚΟΣΤΟΣ ← (20*60)+(ΚΝ-20)*150
          Τέλος_αν
 Τέλος_Αν
 ΣΥΝ-ΚΟΣΤΟΣ ← ΠΑΓΙΟ +ΚΟΣΤΟΣ
 Τύπωσε ΣΥΝ-ΚΟΣΤΟΣ
Τέλος Μηνιαίος_Λογαριαμός
```

ΑΣΚΗΣΗ 2

Μια λύση του παραπάνω προβλήματος είναι η παρακάτω :

```
Αλγόριθμος MAX.1
Διάβασε Α
Διάβασε Β
Διάβασε C
Αν (A>B) ΚΑΙ (A>C) τότε
          Max ← A
Τέλος_Αν
Αν (B>A) ΚΑΙ (B>C) τότε
        Max ← B
Τέλος_αν
Αν C>A ΚΑΙ C>B Τότε
        Max ← C
Τέλος_Αν
Τύπωσε MAX
Τέλος MAX1
```

Η λύση αυτή δεν είναι η καλύτερη, διότι χρησιμοποιούμε τρία διαφορετικά Αν….τότε. Δείτε την παρακάτω λύση

```
Αλγόριθμος MAX2.
Διάβασε Α
Διάβασε Β
Διάβασε C
Αν B>A τότε
        Αν C> B τότε
                MAX← C
            αλλιώς
                MAX←B
```

```
            Τέλος_Αν
        αλλιώς
            Αν C>A τότε
                    MAX←C
                    Αλλιώς
                        MAX←A
             Τέλος_Αν
τέλος_αν
Τύπωσε MAX
Τέλος MAX2
```

Μια Τρίτη εναλλακτική λύση είναι η παρακάτω

```
Αλγόριθμος Max3
Δεδομένα //A,B,C//
Αν A>B και A>C τότε
            MAX←A
αλλιώς_αν B>A και B>C τότε
            MAX ← B
Αλλιώς
      MAX ←C
τέλος_αν
Αποτελέσματα // MAX//
Τέλος Max3
```

Στα παραπάνω παραδείγματα φαίνεται η διαφορά στη χρήση των διαφόρων μορφών εντολών Αν…. τότε . Προφανώς μπορούν να υλοποιηθούν και άλλοι αλγόριθμοι για την επίλυση του ίδιου προβλήματος.

Λύση άσκησης 4

Αλγόριθμος Υπολογισμοί

```
Διάβασε A ,B , C
Max ← A
Αν B>C τότε
            AFAIRESH ← B-C
                Αλλιώς
                    AFAIRESH← C-B
             Τέλος_Αν
Αν B>MAX τότε
        Max ← B
        Αν A>C τότε
                AFAIRESH ← A-C
                Αλλιώς
                    AFAIRESH ← C-A
        Τέλος_Αν
        αλλιώς
            Αν C> MAX τότε
                MAX← C
                Αν A>B τότε
                    AFAIRESH← A-B
```

```
                    Αλλιώς
                        AFAIRESH←B-A
                Τέλος_Αν
            Τέλος_Αν
Τέλος_Αν
Apot ← MAX*AFAIRESH
Τύπωσε Apot
Τέλος Υπολογισμοί
```

ΛΥΣΗ ΑΣΚΗΣΗΣ 5

```
Αλγόριθμος ασκηση_5
Δεδομένα // Β1,Β2,β3,β4//
Αν (Β1>=8) ΚΑΙ (Β2>=8) ΚΑΙ (Β3.=8) ΚΑΙ (Β4>=8) τότε
ΜΟ ←(Β1+Β2+Β3+Β4)/4
    Αν ΜΟ>=10 τότε
        Τύπωσε «ΠΡΟΑΓΕΤΑΙ»
            Αλλιώς
                Τύπωσε «ΑΠΟΡΡΙΠΤΕΤΑΙ»
    Τέλος_αν
     Αλλιώς
       Τύπωσε «ΚΑΠΟΙΟΣ ΒΑΘΜΟΣ <8»
Τέλος_Αν
Τέλος ασκηση_5
```

ΛΥΣΕΙΣ ΑΣΚΗΣΕΩΝ ΣΤΙΣ ΕΠΑΝΑΛΗΠΤΙΚΕΣ ΔΟΜΕΣ

ΛΥΣΗ ΑΣΚΗΣΗΣ 1

```
Αλγόριθμος Αθροισμα1
! Εισαγωγή δεδομένων
Διάβασε Ν
! Επεξεργασία δεδομένων
sum←0
gin←1
 Για χ από 1 μέχρι Ν
          sum← sum+x
          gin ← gin*x
τέλος επανάληψης
mo← sum/n
! Αποτελέσματα
Τύπωσε mo
Τύπωσε sum
Τύπωσε gin
Τέλος Αθροισμα1
```

ΛΥΣΗ ΑΣΚΗΣΗΣ 2

```
Αλγόριθμος Αθροισμα2
! Εισαγωγή δεδομένων
Διάβασε N
! Επεξεργασία δεδομένων
m←0
sum←0
gin←1
 Για χ από 1 μέχρι N βήμα 2
          m←m+1
          sum← sum+x
          gin ← gin*x
τέλος επανάληψης
mo← sum/m
! Αποτελέσματα
Τύπωσε mo
Τύπωσε sum
Τύπωσε gin
Τέλος Αθροισμα2
```

ΛΥΣΗ ΑΣΚΗΣΗΣ 3

```
Αλγόριθμος Αθροισμα3
! Εισαγωγή δεδομένων
Διάβασε N
! Επεξεργασία δεδομένων
m←0
sum←0
gin←1
 Για χ από 2 μέχρι N βήμα 2
          sum← sum+x
          gin ← gin*x
           m←m+1
τέλος επανάληψης
mo← sum/m
! Αποτελέσματα
Τύπωσε mo
Τύπωσε sum
Τύπωσε gin
Τέλος Αθροισμα3
```

ΛΥΣΕΙΣ ΑΣΚΗΣΕΩΝ στις ΜΕΤΑΤΡΟΠΕΣ αλγορίθμων

Λύση άσκησης 1

```
Αλγόριθμος Εμφωλευμένες
Διάβασε α
Αν α < 0 τότε
          εμφάνισε "Αρνητικό"
```

```
    αλλιώς_αν α < = 100 τότε
        εμφάνισε "Μεταξύ 0 και 100"
    αλλιώς_αν α < = 200 τότε
        εμφάνισε "Μεταξύ 100 και 200"
    αλλιώς
        εμφάνισε "Μεγαλύτερο από 200"
Τέλος_αν
Τέλος Εμφωλευμένες
```