Ανάπτυξη Εφαρμογών σε Προγραμματιστικό Περιβάλλον

Μεθοδολογία Ασκήσεων

Μεθοδολογία προκάτ, γιατί μπαμπά;

Όχι, ο προγραμματισμός σίγουρα δεν διδάσκεται με συγκεκριμένες τυποποιημένες μεθόδους προγραμματισμού, καθώς τα προβλήματα που καλείσαι να λύσεις ως προγραμματιστής μπορεί να είναι τόσο διαφορετικά που να επιδέχονται άπειρους τρόπους επίλυσης.

Το συγκεκριμένο μάθημα όμως και η ύλη του, που εστιάζεται σε συγκεκριμένα είδη προβλημάτων, μας επιτρέπει να κάνουμε ένα είδος «οδηγού» για αυτά τα προβλήματα ως μία πρόταση για τον τρόπο που μπορούμε να τα προσεγγίσουμε. Σε κάθε περίπτωση όμως δεν μπορεί να χρησιμοποιηθεί ως τυφλοσούρτης, καθώς θα πρέπει να γνωρίζεις καλά τι είναι αυτό που κάνεις και να το προσαρμόσεις στο εκάστοτε πρόβλημα.

Όσοι λοιπόν δεν έχουν κανένα πρόβλημα με τη λογική πίσω από αυτόν τον οδηγό, μπορούν να τον διαβάσουν και ενδεχομένως να βοηθηθούν.

Με τις υγείες μας

Περιεχόμενα

Έλεγχος εισαγόμενων τιμών

Ας δούμε ένα παράδειγμα όπου δεν απαιτείται έλεγχος τιμών

Να γίνει αλγόριθμος που να διαβάζει τους βαθμούς 12 μαθητών και να υπολογίζει το μέσο όρο τους.

Αλγόριθμος έλεγχος
Σύνολο←0
Για i **από** 1 **μέχρι** 12
    Διάβασε ΒΑΘΜΟΣ
    Σύνολο ← Σύνολο + ΒΑΘΜΟΣ
Τέλος_επανάληψης
ΜΟ ← Σύνολο / 12
Εμφάνισε "Ο μέσος όρος των βαθμών είναι: ", ΜΟ
Τέλος έλεγχος

Για τον έλεγχο των τιμών που εισάγονται χρησιμοποιούμε μία επανάληψη η οποία ξαναζητάει την εισαγωγή της τιμής όσο δεν δίνεται η σωστή.

Να γίνει αλγόριθμος που να διαβάζει τους βαθμούς 12 μαθητών και να υπολογίζει το μέσο όρο τους. Η κάθε τιμή που εισάγεται να ελέγχεται αν είναι μέσα στο διάστημα από 1 μέχρι 20. Σε λανθασμένη εισαγωγή να εμφανίζεται ανάλογο μήνυμα

Αλγόριθμος έλεγχος
Σύνολο←0
Για i **από** 1 **μέχρι** 12
    Διάβασε ΒΑΘΜΟΣ
    Όσο (ΒΑΘΜΟΣ < 1) **ή** (ΒΑΘΜΟΣ > 20) **επανάλαβε**
        Εμφάνισε "Λάθος εισαγωγή, πρέπει να είναι μεταξύ 1 – 20"
        Διάβασε ΒΑΘΜΟΣ
    Τέλος_επανάληψης
    Σύνολο ← Σύνολο + ΒΑΘΜΟΣ
Τέλος_επανάληψης
ΜΟ ← Σύνολο / 12
Εμφάνισε "Ο μέσος όρος των βαθμών είναι: ", ΜΟ
Τέλος έλεγχος

Ο ίδιος έλεγχος μπορεί να γίνει με την άλλη δομή επανάληψης

```
Αλγόριθμος έλεγχος
Σύνολο←0
Για i από 1 μέχρι 12
        Αρχή_επανάληψης
                Διάβασε ΒΑΘΜΟΣ
                Αν (ΒΑΘΜΟΣ < 1) ή (ΒΑΘΜΟΣ > 20) τότε
                   Εμφάνισε "Λάθος εισαγωγή, πρέπει να είναι μεταξύ 1 – 20"
                Τέλος_αν
        Μέχρις_ότου (ΒΑΘΜΟΣ > 0) και (ΒΑΘΜΟΣ <21)
        Σύνολο ← Σύνολο + ΒΑΘΜΟΣ
Τέλος_επανάληψης
ΜΟ ← Σύνολο / 12
Εμφάνισε "Ο μέσος όρος των βαθμών είναι: ", ΜΟ
Τέλος έλεγχος
```

Είναι καλό, να μη μπερδεύουμε τον έλεγχο εισαγόμενων τιμών με οποιονδήποτε άλλον έλεγχο που απαιτεί το πρόγραμμά μας. Πρώτα θα γίνεται ο έλεγχος των εισαγόμενων τιμών και μετά οποιοσδήποτε άλλος έλεγχος σε αυτές τις τιμές, σύμφωνα με τις απαιτήσεις του προγράμματός μας.

Εύρεση μεγίστου

Χωρίς Πίνακα

Για την εύρεση μεγίστου ακολουθούμε τον παρακάτω αλγόριθμο:

- Θεωρούμε μία αρχική τιμή ως μέγιστη
- Συγκρίνουμε όλες τις υπόλοιπες τιμές με τη μέγιστη
- Αν βρεθεί κάποια τιμή **μεγαλύτερη** από τη μέγιστη τότε τίθεται αυτή ως μέγιστη

Στην περίπτωση που **δεν υπάρχει πίνακας**, ως μέγιστη τίθεται μία τιμή που είμαστε σίγουροι ότι θα αλλάξει με την πρώτη σύγκριση. Όταν δεν υπάρχει πίνακας η όλη διαδικασία εύρεσης μεγίστου γίνεται κατά το διάβασμα των τιμών.

Να γίνει αλγόριθμος που να διαβάζει τους 12 βαθμούς ενός μαθητή και να εμφανίζει το μέγιστο.

```
Αλγόριθμος εύρεση_μεγίστου
Max ← 0                    ! Αυτή η τιμή σίγουρα θα αλλάξει με την πρώτη
Για i από 1 μέχρι 12       ! σύγκριση. Θα μπορούσε να είναι -1 ή -100
        Διάβασε ΒΑΘΜΟΣ
        Αν ΒΑΘΜΟΣ > Max τότε
                Max ← ΒΑΘΜΟΣ
        Τέλος_αν
Τέλος_επανάληψης
Εμφάνισε "Ο μέγιστος βαθμός είναι: ",Max
Τέλος εύρεση_μεγίστου
```

Αν πέρα από τη μέγιστη τιμή μας ζητείται και σε ποια θέση βρέθηκε, τότε θα πρέπει να χρησιμοποιήσουμε μία μεταβλητή που να αποθηκεύει τη θέση αυτή. Π.χ.

Να γίνει αλγόριθμος που να διαβάζει τους 12 βαθμούς ενός μαθητή και να εμφανίζει το μέγιστο βαθμό και σε ποιο μάθημα βρέθηκε.

```
Αλγόριθμος εύρεση_μεγίστου
Max ← 0
ΘΕΣΗ ← 0
Για i από 1 μέχρι 12
        Διάβασε ΒΑΘΜΟΣ
        Αν ΒΑΘΜΟΣ > Max τότε
                Max ← ΒΑΘΜΟΣ
                ΘΕΣΗ ← i
        Τέλος_αν
Τέλος_επανάληψης
Εμφάνισε "Ο μέγιστος βαθμός είναι: ",Max, "στο μάθημα: ", ΘΕΣΗ
Τέλος εύρεση_μεγίστου
```

Μονοδιά -στατος Πίνακας

Στην περίπτωση που **υπάρχει πίνακας**, ως αρχική μέγιστη τιμή θεωρείται η τιμή στην πρώτη θέση του πίνακα και η σύγκριση των υπόλοιπων τιμών ξεκινάει από τη δεύτερη θέση του πίνακα.

Να γίνει αλγόριθμος που έχοντας ως δεδομένο τον πίνακα ΒΑΘΜΟΣ με τους 12 βαθμούς ενός μαθητή, να αναζητά και να εμφανίζει το μέγιστο βαθμό και σε ποιο μάθημα βρέθηκε.

```
Αλγόριθμος εύρεση_μεγίστου
Δεδομένα // ΒΑΘΜΟΣ[12] //
Max ← ΒΑΘΜΟΣ[1]
ΘΕΣΗ ← 1
Για i από 2 μέχρι 12
        Αν ΒΑΘΜΟΣ[i] > Max τότε
                Max ← ΒΑΘΜΟΣ[i]
                ΘΕΣΗ ← i
        Τέλος_αν
Τέλος_επανάληψης
Εμφάνισε "Ο μέγιστος βαθμός είναι: ",Max, "στο μάθημα: ", ΘΕΣΗ
Αποτελέσματα // Max , ΘΕΣΗ //
Τέλος εύρεση_μεγίστου
```

Στην περίπτωση που έχουμε δύο παράλληλους πίνακας η ΘΕΣΗ χρησιμοποιείται και για τους δύο πίνακες

Να γίνει αλγόριθμος που να διαβάζει σε αντίστοιχους πίνακες 12 θέσεων το μάθημα και το βαθμό του μαθήματος ενός μαθητή και να εμφανίζει το μέγιστο βαθμό και σε ποιο μάθημα ήταν αυτός.

```
Αλγόριθμος εύρεση_μεγίστου
Για i από 1 μέχρι 12
        Εμφάνισε "Δώσε μάθημα και βαθμό"
        Διάβασε ΜΑΘΗΜΑ[i],ΒΑΘΜΟΣ[i]
Τέλος_επανάληψης
Max ← ΒΑΘΜΟΣ[1]
ΘΕΣΗ ← 1
Για i από 2 μέχρι 12
        Αν ΒΑΘΜΟΣ[i] > Max τότε
                Max ← ΒΑΘΜΟΣ[i]
                ΘΕΣΗ ← i
        Τέλος_αν
Τέλος_επανάληψης
Εμφάνισε "Ο μέγιστος βαθμός είναι: ",Max, "στο μάθημα: ", ΜΑΘΗΜΑ[ΘΕΣΗ]
Τέλος εύρεση_μεγίστου
```

Εναλλακτικά θα μπορούσε να είναι:

```
Εμφάνισε "Ο μέγιστος βαθμός είναι: ", ΒΑΘΜΟΣ[ΘΕΣΗ], "στο μάθημα: ", ΜΑΘΗΜΑ[ΘΕΣΗ]
```

Εύρεση ελαχίστου

Για την εύρεση ελαχίστου ακολουθούμε την ίδια ακριβώς λογική με τη μόνη διαφορά στη σύγκριση των τιμών:

- Θεωρούμε μία αρχική τιμή ως ελάχιστη
- Συγκρίνουμε όλες τις υπόλοιπες τιμές με την ελάχιστη
- Αν βρεθεί κάποια τιμή **μικρότερη** από την ελάχιστη τότε τίθεται αυτή ως ελάχιστη

Δισδιά-στατος Πίνακας

Σε περίπτωση δισδιάστατου πίνακα θεωρείται ελάχιστη τιμή η θέση [1,1] του πίνακα και σαρώνεται ο υπόλοιπος πίνακας. Σε αυτήν την περίπτωση όμως δεν μπορούμε να ξεκινήσουμε από τη δεύτερη θέση γιατί έτσι θα χαθεί μία ολόκληρη γραμμή ή στήλη.

Να γίνει αλγόριθμος που να διαβάζει τους βαθμούς 8 μαθητών σε 12 μαθήματα και να εμφανίζει ποιος είναι ο ελάχιστος βαθμός από όλους.

```
Αλγόριθμος εύρεση_ελαχίστου
Για i από 1 μέχρι 8
   Για j από 1 μέχρι 12
        Εμφάνισε "Δώσε βαθμό του μαθητή ",i ,"στο μάθημα ", j
        Διάβασε ΒΑΘΜΟΛΟΓΙΑ[i,j]
   Τέλος_επανάληψης
Τέλος_επανάληψης

Min ← ΒΑΘΜΟΛΟΓΙΑ[1,1]
Για i από 1 μέχρι 8
   Για j από 1 μέχρι 12
        Αν ΒΑΘΜΟΛΟΓΙΑ[i,j] < Min τότε
                Min ← ΒΑΘΜΟΛΟΓΙΑ[i,j]
        Τέλος_αν
   Τέλος_επανάληψης
Τέλος_επανάληψης

Εμφάνισε "Ο ελάχιστος βαθμός είναι: ",Min
Τέλος εύρεση_ελαχίστου
```

Αυτή είναι η πιο απλή περίπτωση εύρεσης ελαχίστου σε δισδιάστατο πίνακα. Συνήθως έχει νόημα η παραπάνω εύρεση όταν ζητείται και αυτός που έχει αυτήν την τιμή. Στην περίπτωση βέβαια ενός δισδιάστατου πίνακα όπως στο προηγούμενο παράδειγμα μπορούμε να ζητήσουμε όχι μόνο ποιος είχε τον ελάχιστο βαθμό αλλά και σε ποιο μάθημα:

Να γίνει αλγόριθμος που να διαβάζει τους βαθμούς 8 μαθητών σε 12 μαθήματα και να εμφανίζει ποιος είναι ο ελάχιστος βαθμός από όλους, ποιος μαθητής τον έχει και σε ποιο μάθημα.

Αλγόριθμος εύρεση_ελαχίστου

```
Για i από 1 μέχρι 8
   Για j από 1 μέχρι 12
        Εμφάνισε "Δώσε βαθμό του μαθητή ",i ,"στο μάθημα ", j
        Διάβασε ΒΑΘΜΟΛΟΓΙΑ[i,j]
   Τέλος_επανάληψης
Τέλος_επανάληψης

Min ← ΒΑΘΜΟΛΟΓΙΑ[1,1]
ΘΕΣΗ_ΜΑΘΗΤΗ ← 1
ΘΕΣΗ_ΜΑΘΗΜΑΤΟΣ ← 1
Για i από 1 μέχρι 8
   Για j από 1 μέχρι 12
        Αν ΒΑΘΜΟΛΟΓΙΑ[i,j] < Min τότε
                Min ← ΒΑΘΜΟΛΟΓΙΑ[i,j]
                ΘΕΣΗ_ΜΑΘΗΤΗ ← i
                ΘΕΣΗ_ΜΑΘΗΜΑΤΟΣ ← j
        Τέλος_αν
   Τέλος_επανάληψης
Τέλος_επανάληψης
Εμφάνισε "Ο ελάχιστος βαθμός είναι: ",Min
Εμφάνισε "Τον είχε ο μαθητής",ΘΕΣΗ_ΜΑΘΗΤΗ, "στο μάθημα",ΘΕΣΗ_ΜΑΘΗΜΑΤΟΣ
Τέλος εύρεση_ελαχίστου
```

Αυτό το παράδειγμα βέβαια θα ήταν πιο κοντά στην πραγματικότητα εάν αντί για αριθμούς χρησιμοποιούσαμε ονόματα μαθητών και την ονομασία των μαθημάτων, κάτι βέβαια που προϋποθέτει την ύπαρξη πινάκων για κάθε μία από τις παραπάνω περιπτώσεις

Να γίνει αλγόριθμος που να διαβάζει

- *τα ονόματα 8 μαθητών, σε μονοδιάστατο πίνακα 8 θέσεων*
- *τα 12 μαθήματά τους, σε μονοδιάστατο πίνακα 12 θέσεων*
- *τους βαθμούς των παραπάνω μαθητών σε κάθε ένα μάθημα σε δισδιάστατο πίνακα 8 Χ 12*

Ο αλγόριθμος να εμφανίζει ποιος μαθητής και σε ποιο μάθημα είχε τον χαμηλότερο βαθμό

```
Αλγόριθμος εύρεση_ελαχίστου
Για i από 1 μέχρι 8
        Εμφάνισε "Δώσε το όνομα του μαθητή "
        Διάβασε ΟΝΟΜΑ[i]
Τέλος_επανάληψης

Για i από 1 μέχρι 12
        Εμφάνισε "Δώσε το μαθημα "
        Διάβασε ΜΑΘΗΜΑ[i]
Τέλος_επανάληψης

Για i από 1 μέχρι 8
   Εμφάνισε "Βαθμολογία μαθητή ",ΟΝΟΜΑ[i]
   Για j από 1 μέχρι 12
        Εμφάνισε "Δώσε βαθμό στο μάθημα ", ΜΑΘΗΜΑ[j]
        Διάβασε ΒΑΘΜΟΛΟΓΙΑ[i,j]
   Τέλος_επανάληψης
Τέλος_επανάληψης

Min ← ΒΑΘΜΟΛΟΓΙΑ[1,1]
ΘΕΣΗ_ΜΑΘΗΤΗ ← 1
ΘΕΣΗ_ΜΑΘΗΜΑΤΟΣ ← 1
Για i από 1 μέχρι 8
   Για j από 1 μέχρι 12
        Αν Min < ΒΑΘΜΟΛΟΓΙΑ[i,j] τότε
                Min ← ΒΑΘΜΟΛΟΓΙΑ[i,j]
                ΘΕΣΗ_ΜΑΘΗΤΗ ← i
                ΘΕΣΗ_ΜΑΘΗΜΑΤΟΣ ← j
        Τέλος_αν
   Τέλος_επανάληψης
Τέλος_επανάληψης
Εμφάνισε "Ο ελάχιστος βαθμός είναι: ",Min
Εμφάνισε "Τον είχε ο μαθητής ", ΟΝΟΜΑ[ΘΕΣΗ_ΜΑΘΗΤΗ],
& "στο μάθημα",ΜΑΘΗΜΑ[ΘΕΣΗ_ΜΑΘΗΜΑΤΟΣ]
Τέλος εύρεση_ελαχίστου
```

Ταξινόμηση

Για την ταξινόμηση ενός πίνακα χρησιμοποιούμε δύο επαναλήψεις, την μία μέσα στην άλλη. Μέσα στις δύο αυτές επαναλήψεις συγκρίνουμε δύο διαδοχικές θέσεις του πίνακα και ανάλογα με το είδος της ταξινόμησης (αύξουσα ή φθίνουσα) τις αντιμεταθέτουμε.

Ένας Πίνακας

Να γίνει πρόγραμμα που να καταχωρεί τους βαθμούς δέκα μαθητών και να κάνει αύξουσα ταξινόμηση

Αλγόριθμος ταξινόμηση

Για i **από** 1 **μέχρι** 10
    Εμφάνισε "Δώσε βαθμό του μαθητή"
    Διάβασε ΒΑΘΜΟΣ[i]
Τέλος_επανάληψης

Για i **από** 2 **μέχρι** 10
  Για j **από** 10 **μέχρι** i με_βήμα -1
    Αν ΒΑΘΜΟΣ[j-1]> ΒΑΘΜΟΣ[j] **τότε**
      temp ← ΒΑΘΜΟΣ[j-1]
      ΒΑΘΜΟΣ[j-1]← ΒΑΘΜΟΣ[j]
      ΒΑΘΜΟΣ[j] ← temp
    Τέλος_αν
  Τέλος_επανάληψης
Τέλος_επανάληψης
Τέλος ταξινόμηση

Να θυμάσαι ότι

- το i ξεκινάει πάντα από 2
- όλες οι συγκρίσεις και οι αλλαγές των θέσεων του πίνακα γίνονται με δείκτη το j
- αν θέλουμε αύξουσα ταξινόμηση, η σύγκριση θα είναι
  Αν ΒΑΘΜΟΣ[j-1] **>** ΒΑΘΜΟΣ[j] **τότε**
- αν θέλουμε φθίνουσα ταξινόμηση, η σύγκριση θα είναι
  Αν ΒΑΘΜΟΣ[j-1] **<** ΒΑΘΜΟΣ[j] **τότε**

Δύο Πίνακες

Μία συνηθισμένη περίπτωση ταξινόμησης είναι αυτή όπου πρέπει να ταξινομήσουμε δύο πίνακες ταυτόχρονα, πρόκειται δηλαδή για **παράλληλη ταξινόμηση** δύο πινάκων που σχετίζονται μεταξύ τους.

*Να γίνει πρόγραμμα που να καταχωρεί τους βαθμούς δέκα μαθητών και τα ονόματά τους και να κάνει **αύξουσα** ταξινόμηση ως προς τους βαθμούς*

Αλγόριθμος ταξινόμηση

Για i **από** 1 **μέχρι** 10
    Εμφάνισε "Δώσε το όνομα του μαθητή "
    Διάβασε ΟΝΟΜΑ[i]
    Εμφάνισε "Δώσε βαθμό του μαθητή"
    Διάβασε ΒΑΘΜΟΣ[i]
Τέλος_επανάληψης

Για i **από** 2 **μέχρι** 10
  Για j **από** 10 **μέχρι** i με_βήμα -1
    Αν ΒΑΘΜΟΣ[j-1]> ΒΑΘΜΟΣ[j] **τότε**
      temp ← ΒΑΘΜΟΣ[j-1]
      ΒΑΘΜΟΣ[j-1]← ΒΑΘΜΟΣ[j]
      ΒΑΘΜΟΣ[j] ← temp
      temp2 ← ΟΝΟΜΑ[j-1]
      ΟΝΟΜΑ[j-1]← ΟΝΟΜΑ[j]
      ΟΝΟΜΑ[j] ← temp2
    Τέλος_αν
  Τέλος_επανάληψης
Τέλος_επανάληψης
Τέλος ταξινόμηση

Αν θέλουμε η ταξινόμηση να γίνει ως προς το όνομα τότε η εντολή σύγκρισης θα έπρεπε να είναι:
    Αν ΟΝΟΜΑ[j-1]> ΟΝΟΜΑ[j] **τότε**

Για την αντιμετάθεση του δεύτερου πίνακα δεν χρησιμοποιείται η μεταβλητή temp αλλά η μεταβλητή temp2. Πρέπει να είναι διαφορετική, γιατί συνήθως οι πίνακες που ταξινομούμε έχουν διαφορετικού τύπου δεδομένα (χαρακτήρες, πραγματικοί) οπότε δεν μπορούμε να τα αποθηκεύσουμε στην ίδια μεταβλητή.

Σε πιο πολύπλοκες περιπτώσεις μπορεί να έχουμε ταξινόμηση ως προς και τους δύο πίνακες:

*Να γίνει πρόγραμμα που να καταχωρεί τους βαθμούς δέκα μαθητών και τα ονόματά τους και να κάνει **φθίνουσα** ταξινόμηση ως προς τους βαθμούς. Αν δύο μαθητές έχουν τον ίδιο βαθμό τότε η ταξινόμηση να γίνεται αλφαβητικά (αύξουσα)*

Αλγόριθμος ταξινόμηση

Για i **από** 1 **μέχρι** 10
    Εμφάνισε "Δώσε το όνομα του μαθητή "
    Διάβασε ΟΝΟΜΑ[i]
    Εμφάνισε "Δώσε βαθμό του μαθητή"
    Διάβασε ΒΑΘΜΟΣ[i]
Τέλος_επανάληψης

Για i **από** 2 **μέχρι** 10
  Για j **από** 10 **μέχρι** i με_βήμα -1
   Αν (ΒΑΘΜΟΣ[j-1]<ΒΑΘΜΟΣ[j]) ή
      ((ΒΑΘΜΟΣ[j-1]=ΒΑΘΜΟΣ[j]) και (ΟΝΟΜΑ[j-1]>ΟΝΟΜΑ[j])) **τότε**
          temp ← ΒΑΘΜΟΣ[j-1]
          ΒΑΘΜΟΣ[j-1]← ΒΑΘΜΟΣ[j]
          ΒΑΘΜΟΣ[j] ← temp
          temp2 ← ΟΝΟΜΑ[j-1]
          ΟΝΟΜΑ[j-1]← ΟΝΟΜΑ[j]
          ΟΝΟΜΑ[j] ← temp2
    Τέλος_αν
  Τέλος_επανάληψης
Τέλος_επανάληψης
Τέλος ταξινόμηση

Σειριακή αναζήτηση

Στη σειριακή αναζήτηση σαρώνουμε έναν πίνακα μέχρι να βρούμε αυτό που ψάχνουμε ή μέχρι να φτάσουμε στο τέλος του πίνακα. Αν δηλαδή βρούμε αυτό που ψάχνουμε σταματάει και η αναζήτηση. Σειριακή αναζήτηση χρησιμοποιούμε όταν μας ζητούν να απαντήσουμε στο ερώτημα αν υπάρχει **έστω μία** τιμή από αυτό που ψάχνουμε.
Για το σκοπό αυτό χρησιμοποιούμε μία λογική μεταβλητή η οποία έχει αρχική τιμή ΨΕΥΔΗΣ και αν βρούμε αυτό που ψάχνουμε την θέτουμε ΑΛΗΘΗΣ.

Να γίνει αλγόριθμος που να διαβάζει δέκα ονόματα σε έναν πίνακα και να αναζητά αν υπάρχει το όνομα 'Μαρία' ανάμεσά σε αυτά.

```
Αλγόριθμος σειριακή_αναζήτηση
Για i από 1 μέχρι 10
        Εμφάνισε "Δώσε το όνομα "
        Διάβασε ΟΝΟΜΑ[i]
Τέλος_επανάληψης

ΒΡΕΘΗΚΕ ← ψευδής
i ← 1
οσο (ΒΡΕΘΗΚΕ = ψευδής) και (i≤10) επανάλαβε
   αν ΟΝΟΜΑ[i] = "Μαρία" τότε
        ΒΡΕΘΗΚΕ ← αληθής
   Αλλιώς
        i← i + 1
   Τέλος_αν
Τέλος_επανάληψης

Αν ΒΡΕΘΗΚΕ = αληθής τότε
        Εμφάνισε "Υπάρχει το όνομα «Μαρία» μέσα στον πίνακα"
αλλιώς
        Εμφάνισε "Δεν υπάρχει το όνομα «Μαρία» μέσα στον πίνακα"
Τέλος_αν

Τέλος σειριακή_αναζήτηση
```

Η σειριακή αναζήτηση δεν είναι κατάλληλη όταν το ερώτημα που τίθεται αφορά το πόσες φορές υπάρχει μία τιμή σε έναν πίνακα.

Να γίνει αλγόριθμος που να διαβάζει δέκα ονόματα σε έναν πίνακα και να αναζητά πόσες φορές υπάρχει το όνομα 'Μαρία' ανάμεσά σε αυτά.

Στην περίπτωση αυτή είμαστε εκ των πραγμάτων αναγκασμένοι να σαρώσουμε τον πίνακά μας από την αρχή μέχρι το τέλος

```
Αλγόριθμος ολική_σάρωση
Για i από 1 μέχρι 10
    Εμφάνισε "Δώσε το όνομα "
    Διάβασε ΟΝΟΜΑ[i]
Τέλος_επανάληψης

k ← 0
Για i από 1 μέχρι 10
   αν ΟΝΟΜΑ[i] = "Μαρία" τότε
        k← k + 1
   Τέλος_αν
Τέλος_επανάληψης

Αν k≠ 0 τότε
        Εμφάνισε "Υπάρχει το όνομα «Μαρία» μέσα στον πίνακα ",k," φορές
αλλιώς
        Εμφάνισε "Δεν υπάρχει το όνομα «Μαρία» μέσα στον πίνακα"
Τέλος_αν

Τέλος ολική_σάρωση
```

Το πρώτο παράδειγμα είναι η πιο απλή μορφή σειριακής αναζήτησης όπου δεν μας ζητούν σε ποια θέση βρήκαμε αυτό που ψάχναμε. Αν μας ζητούν και τη θέση τότε θα πρέπει να κάνουμε την παρακάτω τροποποίηση. Μία ακόμα διαφορά στον επόμενο παράδειγμα είναι επίσης ότι ο αλγόριθμος ζητάει από τον ίδιο το χρήστη να δώσει το όνομα προς αναζήτηση.

Να γίνει αλγόριθμος που να διαβάζει δέκα ονόματα σε έναν πίνακα, να διαβάζει ένα όνομα προς αναζήτηση και να αναζητά αν υπάρχει το όνομα αυτό ανάμεσά στα ονόματα του πίνακα. Αν υπάρχει να εμφανίζει και τη θέση του στον πίνακα

```
Αλγόριθμος σειριακή_αναζήτηση
Για i από 1 μέχρι 10
        Εμφάνισε "Δώσε το όνομα "
        Διάβασε ΟΝΟΜΑ[i]
Τέλος_επανάληψης

Εμφάνισε "Δώσε το όνομα προς αναζήτηση"
Διάβασε ΝΑΜΕ

ΒΡΕΘΗΚΕ ← ψευδής
ΘΕΣΗ ← 0
i ← 1
οσο (ΒΡΕΘΗΚΕ = ψευδής) και (i≤10) επανάλαβε
   αν ΟΝΟΜΑ[i] = ΝΑΜΕ τότε
        ΒΡΕΘΗΚΕ ← αληθής
        ΘΕΣΗ ← i
   Αλλιώς
        i← i + 1
   Τέλος_αν
Τέλος_επανάληψης

Αν ΒΡΕΘΗΚΕ = αληθής τότε
        Εμφάνισε "Υπάρχει το όνομα ",ΝΑΜΕ, " στη θέση ", ΘΕΣΗ
αλλιώς
        Εμφάνισε "Δεν υπάρχει το όνομα  ",ΝΑΜΕ, "  μέσα στον πίνακα"
Τέλος_αν

Τέλος σειριακή_αναζήτηση
```

Για να δώσουμε ένα παράδειγμα πιο κοντά στην πραγματικότητα μπορούμε να το συνδυάσουμε το προηγούμενο παράδειγμα με έναν αντίστοιχο πίνακα τηλεφώνων.

Να γίνει αλγόριθμος που να διαβάζει δέκα ονόματα και τα αντίστοιχα τηλέφωνά τους σε δύο μονοδιάστατους πίνακες. Μετά να διαβάζει ένα όνομα προς αναζήτηση, να το αναζητά αν υπάρχει το όνομα αυτό να το εμφανίζει με το αντίστοιχο τηλέφωνο. Αν όχι να εμφανίζει ανάλογο μήνυμα.

```
Αλγόριθμος σειριακή_αναζήτηση
Για i από 1 μέχρι 10
        Εμφάνισε "Δώσε το όνομα και το τηλέφωνο"
        Διάβασε ΟΝΟΜΑ[i], ΤΗΛΕΦΩΝΟ[i]
Τέλος_επανάληψης

Εμφάνισε "Δώσε το όνομα προς αναζήτηση"
Διάβασε NAME

ΒΡΕΘΗΚΕ ← ψευδής
ΘΕΣΗ ← 0
i ← 1
οσο (ΒΡΕΘΗΚΕ = ψευδής) και (i≤10) επανάλαβε
   αν ΟΝΟΜΑ[i] = NAME τότε
        ΒΡΕΘΗΚΕ ← αληθής
        ΘΕΣΗ ← i
   Αλλιώς
        i← i + 1
   Τέλος_αν
Τέλος_επανάληψης

Αν ΒΡΕΘΗΚΕ = αληθής τότε
        Εμφάνισε "Το όνομα ",NAME, " έχει τηλέφωνο ", ΤΗΛΕΦΩΝΟ[ΘΕΣΗ]
αλλιώς
        Εμφάνισε "Δεν υπάρχει το όνομα  ",NAME, "  μέσα στον πίνακα"
Τέλος_αν

Τέλος σειριακή_αναζήτηση
```

Δισδιάστατοι πίνακες

Για οποιαδήποτε επεξεργασία δισδιάστατων πινάκων χρειαζόμαστε δύο επαναλήψεις, τη μία μέσα στην άλλη προκειμένου να τους σαρώσουμε και να επεξεργαστούμε τα στοιχεία τους.

Σε έναν πίνακα καταχωρούνται οι εισπράξεις μιας αλυσίδας 8 καταστημάτων τα τελευταία 6 χρόνια. Να γίνει αλγόριθμος που να καταχωρεί τα παραπάνω στοιχεία σε δισδιάστατο πίνακα και να εμφανίζει τα σύνολα του κάθε καταστήματος σε όλα τα χρόνια.

Αλγόριθμος δισδιάστατος

Για i **από** 1 **μέχρι** 8
  Για j **από** 1 **μέχρι** 6
    Εμφάνισε "Δώσε τις εισπράξεις του ",i ,"καταστήματος στο έτος ", j
    Διάβασε ΕΙΣΠΡΑΞΗ[i,j]
  Τέλος_επανάληψης
Τέλος_επανάληψης

Για i **από** 1 **μέχρι** 8
  Σ ← 0
  Για j **από** 1 **μέχρι** 6
    Σ ← Σ + ΕΙΣΠΡΑΞΗ[i,j]
  Τέλος_επανάληψης
  Εμφάνισε "Οι εισπράξεις του ",i ,"ου καταστήματος είναι ", Σ
Τέλος_επανάληψης

Τέλος δισδιάστατος

Για την καταχώρηση των στοιχείων χρησιμοποιήθηκε πίνακας 8 Χ 6 ο οποίος σαρώθηκε με τη βοήθεια δύο επαναλήψεων, η εξωτερική επανάληψη για της γραμμές και η εσωτερική επανάληψη για τις στήλες.

Υπολογισμός των εσόδων του κάθε καταστήματος σημαίνει ότι πρέπει να υπολογίσουμε και να εμφανίσουμε το σύνολο της κάθε γραμμής. Για να υπολογίσουμε τα σύνολα για κάθε γραμμή χρησιμοποιούμε τη μεταβλητή Σ την οποία μηδενίζουμε πριν από την εσωτερική επανάληψη προκειμένου να υπολογίσουμε ξανά το σύνολο για κάθε νέα γραμμή. Μετά το τέλος της εσωτερικής επανάληψης έχουμε υπολογίσει το σύνολο μιας ολόκληρης γραμμής οπότε και εμφανίζουμε το αποτέλεσμα. Αν η εμφάνιση του Σ δεν γίνει εκείνη τη στιγμή, το σύνολο θα χαθεί γιατί στην επόμενη επανάληψη του i θα μηδενιστεί.

Στο προηγούμενο παράδειγμα η σάρωση γίνεται κατά γραμμή. Αν θέλουμε η σάρωση να γίνει κατά στήλη τότε στην εξωτερική επανάληψη θα είναι οι στήλες (j) και στην εσωτερική επανάληψη οι γραμμές (i).

Σε έναν πίνακα καταχωρούνται οι εισπράξεις μιας αλυσίδας 8 καταστημάτων τα τελευταία 6 χρόνια. Να γίνει αλγόριθμος που να καταχωρεί τα παραπάνω στοιχεία σε δισδιάστατο πίνακα και να εμφανίζει τα σύνολα της κάθε χρονιάς για όλα τα καταστήματα

Αλγόριθμος δισδιάστατος

Για i **από** 1 **μέχρι** 8
  Για j **από** 1 **μέχρι** 6
    Εμφάνισε "Δώσε τις εισπράξεις του ",i ,"καταστήματος στο έτος ", j
    Διάβασε ΕΙΣΠΡΑΞΗ[i,j]
  Τέλος_επανάληψης
Τέλος_επανάληψης

Για j **από** 1 **μέχρι** 6
  Σ ← 0
  Για i **από** 1 **μέχρι** 8
    Σ ← Σ + ΕΙΣΠΡΑΞΗ[i,j]
  Τέλος_επανάληψης
  Εμφάνισε "Οι εισπράξεις της ",j ,"ης χρονιάς είναι ", Σ
Τέλος_επανάληψης

Τέλος δισδιάστατος

Εδώ το σύνολο που μας ζητάει η άσκηση αφορά μία ολόκληρη χρονιά για όλα τα υποκαταστήματα, άρα το σύνολο της κάθε στήλης. Για το σκοπό αυτό πρέπει η εξωτερική επανάληψη να αφορά της στήλες και η εσωτερική επανάληψη τις γραμμές. Αυτή είναι και η μόνη διαφορά από το προηγούμενο παράδειγμα. Η λογική κατά τα άλλα είναι ακριβώς η ίδια.

Όταν το αποτέλεσμα της επεξεργασίας του πίνακα δεν χρειάζεται περαιτέρω επεξεργασία τότε απλώς το εμφανίζουμε. Διαφορετικά, το αποθηκεύουμε σε έναν μονοδιάστατο πίνακα προκειμένου να το επεξεργαστούμε αργότερα. Αν για παράδειγμα στην προηγούμενη άσκηση μας ζητήσουν να ταξινομήσουμε τα σύνολα, τότε πρέπει αυτά πρώτα να αποθηκευτούν σε πίνακα και μετά να γίνει η ταξινόμηση.

Σε έναν πίνακα καταχωρούνται οι εισπράξεις μιας αλυσίδας 8 καταστημάτων τα τελευταία 6 χρόνια. Να γίνει αλγόριθμος που να καταχωρεί τα παραπάνω στοιχεία σε δισδιάστατο πίνακα και να υπολογίζει τα σύνολα του κάθε καταστήματος σε όλα τα χρόνια. Τα σύνολα να εμφανίζονται ταξινομημένα κατά αύξουσα σειρά.

Αλγόριθμος δισδιάστατος

```
Για i από 1 μέχρι 8
   Για j από 1 μέχρι 6
        Εμφάνισε "Δώσε τις εισπράξεις του ",i ,"καταστήματος στο έτος ", j
        Διάβασε ΕΙΣΠΡΑΞΗ[i,j]
   Τέλος_επανάληψης
Τέλος_επανάληψης
```


```
Για i από 1 μέχρι 8
   Σ ← 0
   Για j από 1 μέχρι 6
        Σ ← Σ + ΕΙΣΠΡΑΞΗ[i,j]
   Τέλος_επανάληψης
   ΣΥΝΟΛΟ[i] ← Σ
Τέλος_επανάληψης
```

```
Για i από 1 μέχρι 8
   ΣΥΝΟΛΟ[i]← 0
   Για j από 1 μέχρι 6
        ΣΥΝΟΛΟ[i] ← ΣΥΝΟΛΟ[i]+ ΕΙΣΠΡΑΞΗ[i,j]
   Τέλος_επανάληψης
Τέλος_επανάληψης
```

```
Για i από 2 μέχρι 8
   Για j από 8 μέχρι i με_βήμα -1
        Αν ΣΥΝΟΛΟ[j-1] > ΣΥΝΟΛΟ[j] τότε
              temp ← ΣΥΝΟΛΟ[j-1]
              ΣΥΝΟΛΟ[j-1]← ΣΥΝΟΛΟ[j]
              ΣΥΝΟΛΟ[j] ← temp
        Τέλος_αν
   Τέλος_επανάληψης
Τέλος_επανάληψης

Εμφάνισε "Οι εισπράξεις των καταστημάτων ταξινομημένες"
Για i από 1 μέχρι 8
   Εμφάνισε ΣΥΝΟΛΟ[i]
Τέλος_επανάληψης

Τέλος δισδιάστατος
```

Το προηγούμενο παράδειγμα μπορούμε να το εμπλουτίσουμε και να το κάνουμε λίγο πιο ενδιαφέρον

Προκειμένου να γίνουν κάποιες στατιστικές μετρήσεις σε μία αλυσίδα 8 καταστημάτων για τις εισπράξεις 6 ετών, να γίνει αλγόριθμος που να:

- *καταχωρεί τα ονόματα των διευθυντών του κάθε καταστήματος σε πίνακα 8 θέσεων*
- *καταχωρεί τη χρονιά (π.χ. 2002) σε έναν πίνακα 8 θέσεων*
- *καταχωρεί τις εισπράξεις του κάθε καταστήματος για κάθε χρονιά σε έναν δισδιάστατο πίνακα 8 Χ 6*
- *υπολογίζει και να εμφανίζει ποιο υποκατάστημα (όνομα διευθυντή) και σε ποια χρονιά έκανε την μεγαλύτερη είσπραξη*
- *υπολογίζει και να εμφανίζει το μέσο όρο εισπράξεων κάθε χρονιάς*
- *υπολογίζει την ελάχιστη είσπραξη σε όλες τις χρονιές του κάθε υποκαταστήματος.*
- *εμφανίζει τους ιδιοκτήτες και τις παραπάνω μέγιστες εισπράξεις ταξινομημένα ως προς τις εισπράξεις κατά φθίνουσα σειρά*

Αλγόριθμος δισδιάστατος

Για i **από** 1 **μέχρι** 8
  Εμφάνισε "Δώσε το όνομα του διευθυντή του ",i ," καταστήματος
  Διάβασε ΔΙΕΥΘΥΝΤΗΣ[i]
Τέλος_επανάληψης

Για i **από** 1 **μέχρι** 6
  Εμφάνισε "Δώσε το έτος"
  Διάβασε ΕΤΟΣ[i]
Τέλος_επανάληψης

Για i **από** 1 **μέχρι** 8
  Εμφάνισε "Υποκατάστημα του ", ΔΙΕΥΘΥΝΤΗΣ[i]
  Για j **από** 1 **μέχρι** 6
    Εμφάνισε "Δώσε τις εισπράξεις για το έτος ", ΕΤΟΣ[j]
    Διάβασε ΕΙΣΠΡΑΞΗ[i,j]
  Τέλος_επανάληψης
Τέλος_επανάληψης

ΜΑΧ ← ΕΙΣΠΡΑΞΗ[1,1]
ΘΕΣΗΔ ← 1
ΘΕΣΗΕ ← 1
Για i **από** 1 **μέχρι** 8
  Για j **από** 1 **μέχρι** 6
    Αν ΕΙΣΠΡΑΞΗ[i,j] > ΜΑΧ **τότε**
      ΜΑΧ ← ΕΙΣΠΡΑΞΗ[i,j]
      ΘΕΣΗΔ ← i
      ΘΕΣΗΕ ← j
    Τέλος_αν
  Τέλος_επανάληψης
Τέλος_επανάληψης

```
Εμφάνισε "Το υποκατάστημα του ", ΔΙΕΥΘΥΝΤΗΣ[ΘΕΣΗΔ], " τη χρονιά ",
&                ΕΤΟΣ[ΘΕΣΗΕ], " είχε τις μεγαλύτερες εισπράξεις ", ΜΑΧ

Για j από 1 μέχρι 6
   Σ ← 0
   Για i από 1 μέχρι 8
        Σ ← Σ + ΕΙΣΠΡΑΞΗ[i,j]
   Τέλος_επανάληψης
   ΜΟ ← Σ/8
   Εμφάνισε "Ο μέσος όρος εισπράξεων για το έτος ", ΕΤΟΣ[j], " είναι ", ΜΟ
Τέλος_επανάληψης

Για i από 1 μέχρι 8
   MIN ← [i,1]
   Για j από 2 μέχρι 6
        Αν MIN < ΕΙΣΠΡΑΞΗ[i,j] τότε
            MIN ← ΕΙΣΠΡΑΞΗ[i,j]
        Τέλος_αν
   Τέλος_επανάληψης
   ΕΛΑΧΙΣΤΟ[i] ← MIN
Τέλος_επανάληψης

Για i από 2 μέχρι 8
   Για j από 8 μέχρι i με_βήμα -1
        Αν ΕΛΑΧΙΣΤΟ[j-1] < ΕΛΑΧΙΣΤΟ[j] τότε
                temp ← ΕΛΑΧΙΣΤΟ[j-1]
                ΕΛΑΧΙΣΤΟ[j-1]← ΕΛΑΧΙΣΤΟ[j]
                ΕΛΑΧΙΣΤΟ[j] ← temp
                temp2 ← ΔΙΕΥΘΥΝΤΗΣ[j-1]
                ΔΙΕΥΘΥΝΤΗΣ[j-1]← ΔΙΕΥΘΥΝΤΗΣ[j]
                ΔΙΕΥΘΥΝΤΗΣ[j] ← temp
        Τέλος_αν
   Τέλος_επανάληψης
Τέλος_επανάληψης

Εμφάνισε "Κατάσταση διευθυντών υποκαταστημάτων και ελάχιστων ετήσιων εισπράξεων"
Για i από 1 μέχρι 8
   Εμφάνισε ΔΙΕΥΘΥΝΤΗΣ[i], ΕΛΑΧΙΣΤΟ [j]
Τέλος_επανάληψης

Τέλος δισδιάστατος
```

Υποπρογράμματα

Τα υποπρογράμματα υλοποιούν τον τμηματικό προγραμματισμό. Είναι αυτόνομα κομμάτια αλγορίθμου που γράφονται μία φορά και μπορούν να χρησιμοποιηθούν μία ή περισσότερες φορές.

Για να εκτελεσθεί ο αλγόριθμος ενός υποπρογράμματος, πρέπει να το καλέσουμε (το υποπρόγραμμα) μέσα από το κύριο πρόγραμμα. Τα υποπρογράμματα ανταλλάσσουν δεδομένα με το κύριο πρόγραμμα μέσω των παραμέτρων.

Τα υποπρογράμματα χωρίζονται σε διαδικασίες και συναρτήσεις. Το επόμενο παράδειγμα είναι με διαδικασία.

Να γίνει πρόγραμμα που να

- *διαβάζει 10 βαθμούς μαθητών*
- *να καλεί μία διαδικασία που να υπολογίζει και να επιστρέφει το μέγιστο βαθμό*
- *να εμφανίζει το μέγιστο βαθμό*

Πρώτο μας βήμα είναι η δημιουργία του κύριου προγράμματος. Σε αυτή τη φάση είναι πολύ σημαντικό να ορίσουμε ποια δεδομένα θα ανταλλάσσονται μεταξύ του κύριου προγράμματος και του υποπρογράμματος. Στη συγκεκριμένη περίπτωση προκειμένου το υποπρόγραμμα να επιστρέφει το μέγιστο βαθμό, θα πρέπει να δέχεται έναν πίνακα με βαθμούς.

```
ΠΡΟΓΡΑΜΜΑ ΜΕΓΙΣΤΟ
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: Ι
    ΠΡΑΓΜΑΤΙΚΕΣ: ΒΑΘΜΟΣ[10], ΜΑΧ

ΑΡΧΗ
    ΓΙΑ Ι ΑΠΟ 1 ΜΕΧΡΙ 10
        ΓΡΑΨΕ 'Δώσε το βαθμό του μαθητή'
        ΔΙΑΒΑΣΕ ΒΑΘΜΟΣ[Ι]
    ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ

    ΚΑΛΕΣΕ ΒΡΕΣ_ΜΕΓΙΣΤΟ(ΒΑΘΜΟΣ, ΜΑΧ)

    ΓΡΑΨΕ 'Ο μέγιστος βαθμός είναι ',ΜΑΧ
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

Αφού έχουμε διαβάσει τους βαθμούς των μαθητών στον πίνακα **ΒΑΘΜΟΣ** καλούμε τη διαδικασία **ΒΡΕΣ_ΜΕΓΙΣΤΟ** με παραμέτρους τον πίνακα

ΒΑΘΜΟΣ και την μεταβλητή **ΜΑΧ**. Η διαδικασία δέχεται τα δεδομένα μέσω του πίνακα **ΒΑΘΜΟΣ** και επιστρέφει το μέγιστο βαθμό μέσω της μεταβλητής **ΜΑΧ.**
Το επόμενο βήμα είναι η δημιουργία της διαδικασίας. Είναι πολύ σημαντικό να χρησιμοποιούμε διαφορετικά ονόματα πινάκων και μεταβλητών ως παραμέτρους από αυτά που χρησιμοποιήσαμε στο κύριο πρόγραμμα.

```
ΔΙΑΔΙΚΑΣΙΑ ΒΡΕΣ_ΜΕΓΙΣΤΟ(Β,Μ)
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: Ι
    ΠΡΑΓΜΑΤΙΚΕΣ: Β[10], Μ
ΑΡΧΗ
    Μ←Β[1]
    ΓΙΑ Ι ΑΠΟ 2 ΜΕΧΡΙ 10
        ΑΝ Β[Ι] > Μ ΤΟΤΕ
            Μ ← Β[Ι]
        ΤΕΛΟΣ_ΑΝ
    ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

Ακόμα και στην περίπτωση που χρησιμοποιήσουμε το ίδιο όνομα παραμέτρου στο κύριο πρόγραμμα και στο υποπρόγραμμα που καλούμε, πρέπει να γνωρίζουμε ότι πρόκειται για δύο διαφορετικές μεταβλητές. Καλό είναι όμως να αποφεύγεται για να μη δημιουργείται σύγχυση.

Με το τέλος της διαδικασίας οι τιμές με τις οποίες κλήθηκε η διαδικασία (μέσω των παραμέτρων) **επιστρέφονται** στο κύριο πρόγραμμα. Οποιεσδήποτε αλλαγές στις τιμές των παραμέτρων μέσα στη διαδικασία θα ισχύσουν και για το κύριο πρόγραμμα. Στην προκειμένη περίπτωση αυτή που αλλάζει είναι η μεταβλητή Μ της οποίας η τιμή επιστρέφεται στην μεταβλητή ΜΑΧ στο κύριο πρόγραμμα.

Η υλοποίηση θα μπορούσε να γίνει και με μία συνάρτηση. Η βασική διαφορά μεταξύ της συνάρτησης και της διαδικασίας είναι ότι **στη συνάρτηση επιστρέφεται μόνο μία τιμή** ενώ **στη διαδικασία μπορούν να επιστραφούν περισσότερες από μία τιμές**. Ο τρόπος κλήσης της συνάρτησης είναι επίσης διαφορετικός. Το παραπάνω παράδειγμα με συνάρτηση υλοποιείται ως εξής:

```
ΠΡΟΓΡΑΜΜΑ ΜΕΓΙΣΤΟ
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: Ι
    ΠΡΑΓΜΑΤΙΚΕΣ: ΒΑΘΜΟΣ[10], ΜΑΧ

ΑΡΧΗ
    ΓΙΑ Ι ΑΠΟ 1 ΜΕΧΡΙ 10
        ΓΡΑΨΕ 'Δώσε το βαθμό του μαθητή'
        ΔΙΑΒΑΣΕ ΒΑΘΜΟΣ[Ι]
    ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ

    ΜΑΧ ← ΒΡΕΣ_ΜΕΓΙΣΤΟ(ΒΑΘΜΟΣ)
```

```
    ΓΡΑΨΕ 'Ο μέγιστος βαθμός είναι ',ΜΑΧ
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

Εδώ βλέπουμε ότι η συνάρτηση χρησιμοποιεί μία παράμετρο αντί για δύο. Αυτό συμβαίνει γιατί η **επιστροφή της τιμής στη συνάρτηση γίνεται με το όνομά της** και όχι μέσω παραμέτρου, όπως συνέβη στη διαδικασία.

```
ΣΥΝΑΡΤΗΣΗ ΒΡΕΣ_ΜΕΓΙΣΤΟ(Β): ΠΡΑΓΜΑΤΙΚΗ
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: Ι
    ΠΡΑΓΜΑΤΙΚΕΣ: Β[10], Μ
ΑΡΧΗ
    Μ←Β[1]
    ΓΙΑ Ι ΑΠΟ 2 ΜΕΧΡΙ 10
        ΑΝ Β[Ι] > Μ ΤΟΤΕ
            Μ ← Β[Ι]
        ΤΕΛΟΣ_ΑΝ
    ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
    ΒΡΕΣ_ΜΕΓΙΣΤΟ ← Μ
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```

Στον ορισμό της συνάρτησης πρέπει να αναφέρουμε τον τύπο των δεδομένων που επιστρέφεται από τη συνάρτηση. Στην συγκεκριμένη περίπτωση ορίσαμε τη συνάρτηση ως ΠΡΑΓΜΑΤΙΚΗ.

Πριν το τέλος της συνάρτησης επιστρέφεται η τιμή της με το όνομά της:

```
    ΒΡΕΣ_ΜΕΓΙΣΤΟ ← Μ
```

Στο παραπάνω παράδειγμα, θα μπορούσαμε να εισάγουμε τις τιμές μέσω μίας διαδικασίας. Σε αυτήν την περίπτωση το κύριο πρόγραμμα θα ήταν ως εξής:

```
ΠΡΟΓΡΑΜΜΑ ΜΕΓΙΣΤΟ
ΜΕΤΑΒΛΗΤΕΣ
    ΠΡΑΓΜΑΤΙΚΕΣ: ΒΑΘΜΟΣ[10], ΜΑΧ
ΑΡΧΗ
    ΚΑΛΕΣΕ ΕΙΣΑΓΩΓΗ_ΤΙΜΩΝ(ΒΑΘΜΟΣ)
    ΜΑΧ ← ΒΡΕΣ_ΜΕΓΙΣΤΟ(ΒΑΘΜΟΣ)
    ΓΡΑΨΕ 'Ο μέγιστος βαθμός είναι ',ΜΑΧ
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

Εδώ καλείται πρώτα η διαδικασία ΕΙΣΑΓΩΓΗ_ΤΙΜΩΝ με παράμετρο τον πίνακα ΒΑΘΜΟΣ. Ο πίνακας αυτός επιστρέφεται γεμάτος τιμές και έπειτα συνεχίζεται η ροή του προγράμματος όπως και πριν.
Αυτό βέβαια σημαίνει ότι τώρα πρέπει να ορίσουμε τη διαδικασία όπου θα γίνεται πλέον η εισαγωγή των τιμών.

```
ΔΙΑΔΙΚΑΣΙΑ ΕΙΣΑΓΩΓΗ_ΤΙΜΩΝ(Β)
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: Ι
    ΠΡΑΓΜΑΤΙΚΕΣ: Β[10]
ΑΡΧΗ
    ΓΙΑ Ι ΑΠΟ 1 ΜΕΧΡΙ 10
        ΓΡΑΨΕ 'Δώσε το βαθμό του μαθητή'
        ΔΙΑΒΑΣΕ Β[Ι]
    ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

Ας δούμε τώρα μία πιο ενδιαφέρουσα παραλλαγή του προηγούμενου παραδείγματος όπου θα δούμε πως «επαναχρησιμοποιείται» ο αλγόριθμος που βρίσκεται σε ένα υποπρόγραμμα. Στο επόμενο παράδειγμα η συνάρτηση θα κληθεί δύο φορές.

Να γίνει πρόγραμμα που να

- *καλεί μία διαδικασία που να διαβάζει 10 βαθμούς μαθητών*
- *να καλεί μία συνάρτηση που να υπολογίζει και να επιστρέφει το μέγιστο βαθμό*
- *να καλεί την ίδια συνάρτηση που να υπολογίζει και να επιστρέφει τον ελάχιστο βαθμό*
- *να εμφανίζει το μέγιστο και ελάχιστο βαθμό*

Η διαδικασία θα δέχεται έναν πίνακα 10 θέσεων στον οποίο θα διαβάζει τους βαθμούς 10 μαθητών

Η συνάρτηση θα δέχεται ως παραμέτρους έναν πίνακα 10 θέσεων με τους βαθμούς των μαθητών και μία δεύτερη παράμετρο, έναν ακέραιο που όταν η τιμή του είναι «1» θα γίνεται εύρεση μεγίστου, ενώ όταν η τιμή του είναι «2» θα γίνεται εύρεση ελαχίστου.

```
ΠΡΟΓΡΑΜΜΑ ΜΕΓΙΣΤΟ_ΚΑΙ_ΕΛΑΧΙΣΤΟ
ΜΕΤΑΒΛΗΤΕΣ
    ΠΡΑΓΜΑΤΙΚΕΣ: ΒΑΘΜΟΣ[10], MAX, MIN
ΑΡΧΗ
    ΚΑΛΕΣΕ ΕΙΣΑΓΩΓΗ_ΤΙΜΩΝ(ΒΑΘΜΟΣ)
    MAX ← ΒΡΕΣ_ΜΕΓΙΣΤΟ_ΕΛΑΧΙΣΤΟ(ΒΑΘΜΟΣ,1)
    MIN ← ΒΡΕΣ_ΜΕΓΙΣΤΟ_ΕΛΑΧΙΣΤΟ (ΒΑΘΜΟΣ,2)
    ΓΡΑΨΕ 'Ο μέγιστος βαθμός είναι ',MAX
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

Εδώ βλέπουμε ότι ως παράμετροι πέρα από πίνακες ή μεταβλητές μπορεί να είναι ακόμα και σταθερές τιμές (1 ή 2).

```
ΔΙΑΔΙΚΑΣΙΑ ΕΙΣΑΓΩΓΗ_ΤΙΜΩΝ(Β)
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: Ι
    ΠΡΑΓΜΑΤΙΚΕΣ: Β[10]
ΑΡΧΗ
    ΓΙΑ Ι ΑΠΟ 1 ΜΕΧΡΙ 10
        ΓΡΑΨΕ 'Δώσε το βαθμό του μαθητή ', Ι
        ΔΙΑΒΑΣΕ Β[Ι]
    ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

```
ΣΥΝΑΡΤΗΣΗ ΒΡΕΣ_ΜΕΓΙΣΤΟ_ΕΛΑΧΙΣΤΟ(Β, ΕΙΔΟΣ): ΠΡΑΓΜΑΤΙΚΗ
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: Ι, ΕΙΔΟΣ
    ΠΡΑΓΜΑΤΙΚΕΣ: Β[10], Μ
ΑΡΧΗ
    Μ←Β[1]
    ΓΙΑ Ι ΑΠΟ 2 ΜΕΧΡΙ 10

        ΕΠΙΛΕΞΕ ΕΙΔΟΣ
            ΠΕΡΙΠΤΩΣΗ 1
                ΑΝ Β[Ι] > Μ ΤΟΤΕ
                    Μ ← Β[Ι]
                ΤΕΛΟΣ_ΑΝ
            ΠΕΡΙΠΤΩΣΗ 2
                ΑΝ Β[Ι] < Μ ΤΟΤΕ
                    Μ ← Β[Ι]
                ΤΕΛΟΣ_ΑΝ
        ΤΕΛΟΣ_ΕΠΙΛΟΓΩΝ
    ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ

    ΒΡΕΣ_ΜΕΓΙΣΤΟ_ΕΛΑΧΙΣΤΟ ← Μ
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```

Εναλλακτικά

```
ΑΝ ΕΙΔΟΣ = 1 ΤΟΤΕ
    ΑΝ Β[Ι] > Μ ΤΟΤΕ
        Μ ← Β[Ι]
    ΤΕΛΟΣ_ΑΝ
ΑΛΛΙΩΣ_ΑΝ ΕΙΔΟΣ = 2
    ΑΝ Β[Ι] < Μ ΤΟΤΕ
        Μ ← Β[Ι]
    ΤΕΛΟΣ_ΑΝ
ΤΕΛΟΣ_ΑΝ
```