

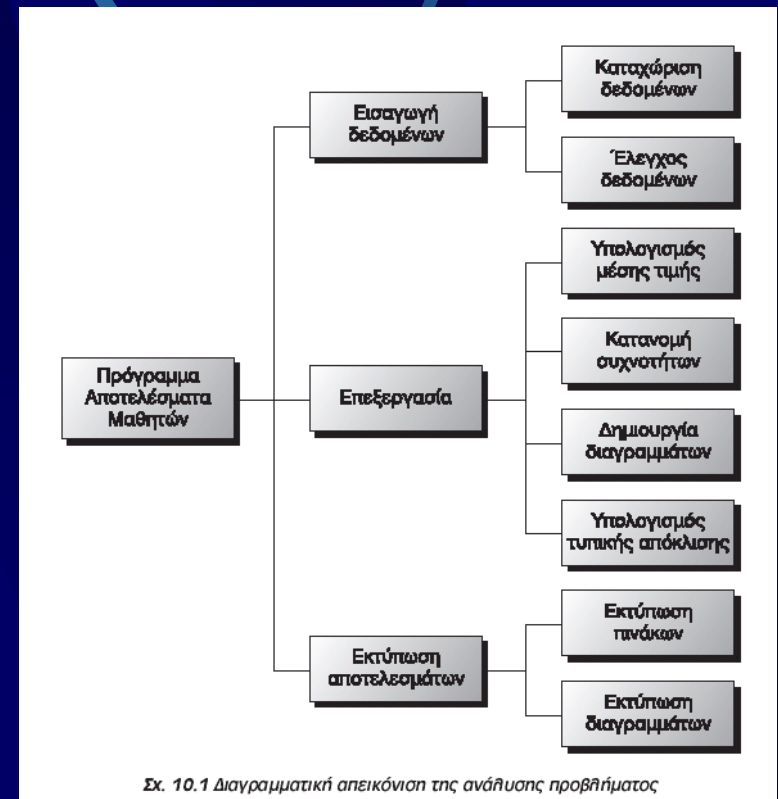
Κεφάλαιο 10 – Υποπρογράμματα

10.1 Τμηματικός προγραμματισμός

- Τμηματικός προγραμματισμός ονομάζεται η τεχνική σχεδίασης και ανάπτυξης των προγραμμάτων ως ένα σύνολο από απλούστερα τμήματα προγραμμάτων.
- Όταν ένα τμήμα προγράμματος επιτελεί ένα αυτόνομο έργο και έχει γραφεί χωριστά από το υπόλοιπο πρόγραμμα, τότε αναφερόμαστε σε **υποπρόγραμμα** - (subprogram).

10.2 Χαρακτηριστικά των υποπρογραμμάτων

- Κάθε υποπρόγραμμα έχει μόνο μία είσοδο (ενεργοποίησης) και μία μόνο έξοδο (απενεργοποίησης)
- Κάθε υποπρόγραμμα πρέπει να είναι ανεξάρτητο από τα άλλα (στο σχεδιασμό, την ανάπτυξη και τη συντήρηση).
- Κάθε υποπρόγραμμα πρέπει να μην είναι πολύ μεγάλο και να εκτελεί μία μόνο λειτουργία → κατανοητό και ελεγχόμενο.



Εκ. 10.1 Διαγραμματική απεικόνιση της ανάπτυξης προβλήματος

10.3 Πλεονεκτήματα του τμηματικού προγραμματισμού

- Διευκολύνει την ανάπτυξη του αλγορίθμου και του αντίστοιχου προγράμματος
- Διευκολύνει την κατανόηση και διόρθωση του προγράμματος
- Απαιτεί λιγότερο χρόνο και προσπάθεια στη συγγραφή του προγράμματος (κλήση υποπρογραμμάτων όπου χρειάζεται και όχι επανεγγραφή του κώδικα)
- Επεκτείνει τις δυνατότητες των γλωσσών προγραμματισμού (βιβλιοθήκες)

10.4 Παράμετροι

- **Παράμετρος** είναι μία μεταβλητή που επιτρέπει το πέρασμα της τιμής της από ένα τμήμα προγράμματος σε ένα άλλο
- Κάθε υποπρόγραμμα για να ενεργοποιηθεί καλείται, όπως λέγεται, από ένα άλλο υποπρόγραμμα ή το αρχικό πρόγραμμα, το οποίο ονομάζεται κύριο πρόγραμμα

10.5 Διαδικασίες και συναρτήσεις

- Η **συνάρτηση** είναι ένας τύπος υποπρογράμματος που υπολογίζει και επιστρέφει μόνο μία τιμή με το όνομά της (όπως οι μαθηματικές συναρτήσεις).
- Η **διαδικασία** είναι ένας τύπος υποπρογράμματος που μπορεί να εκτελεί όλες τις λειτουργίες ενός προγράμματος
- Οι συναρτήσεις εκτελούνται απλά με την εμφάνιση του ονόματος τους σε οποιαδήποτε έκφραση, ενώ για να εκτελεστούν οι διαδικασίες χρησιμοποιείται η ειδική εντολή ΚΑΛΕΣΕ και το όνομα της διαδικασίας

10.5.1 Ορισμός και κλήση συναρτήσεων

Κάθε συνάρτηση έχει την ακόλουθη δομή:

```
ΣΥΝΑΡΤΗΣΗ όνομα (λίστα παραμέτρων) : τύπος συνάρτησης
Τμήμα δηλώσεων
ΑΡΧΗ
    ....
    όνομα <- έκφραση
    ...
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```

10.5.2 Ορισμός και κλήση διαδικασιών

Κάθε διαδικασία έχει την ακόλουθη δομή:

```
ΔΙΑΔΙΚΑΣΙΑ Όνομα (λίστα παραμέτρων)
Τμήμα δηλώσεων
ΑΡΧΗ
    εντολές
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

Δυνατά σημεία κλήσης μιας συνάρτησης (οπουδήποτε μπαίνει μία έκφραση):

- στο δεξί μέρος μιας εκχώρησης π.χ. $x \leftarrow f(y)$
- στην εντολή Γράψε π.χ. Γράψε $f(x)$
- σε μία λογική συνθήκη π.χ. Αν $f(y) \geq 0$ τότε
- στη δομή επίλεξε π.χ. Επίλεξε $f(y)$
- στη λίστα παραμέτρων κλήσης συνάρτησης π.χ. $T_P(f(y))$
- σαν δείκτης αναφοράς σε στοιχείο πίνακα (για ακέραιες συναρτήσεις) π.χ. Γράψε $A[f(y)]$

- Η γενική μορφή της εντολής **ΚΑΛΕΣΕ** είναι:

ΚΑΛΕΣΕ όνομα – διαδικασίας (λίστα – παραμέτρων)

- Λειτουργία: Η εκτέλεση του προγράμματος διακόπτεται και εκτελούνται οι εντολές της διαδικασίας που καλείται. Μετά το τέλος της διαδικασίας η εκτέλεση του προγράμματος συνεχίζεται από την εντολή που ακολουθεί. Η λίστα των παραμέτρων ορίζει τις τιμές που περνούν στη διαδικασία και τις τιμές που αυτή επιστρέφει. Η λίστα παραμέτρων δεν είναι υποχρεωτική.

Να γραφεί πρόγραμμα, το οποίο υπολογίζει το εμβαδό του κύκλου από την ακτίνα του.

ΠΡΟΓΡΑΜΜΑ Παράδειγμα_2

ΜΕΤΑΒΛΗΤΕΣ

ΠΡΑΓΜΑΤΙΚΕΣ : R, E

ΑΡΧΗ

ΚΑΛΕΣΕ Είσοδος_δεδομένων (R)

E <- Εμβαδό_κύκλου (R)

ΚΑΛΕΣΕ Εκτύπωση (E)

ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ

ΔΙΑΔΙΚΑΣΙΑ Είσοδος_δεδομένων (Αριθμός)

ΜΕΤΑΒΛΗΤΕΣ

ΠΡΑΓΜΑΤΙΚΕΣ : Αριθμός

ΑΡΧΗ

ΑΡΧΗ_ΕΠΑΝΑΛΗΨΗΣ

ΓΡΑΨΕ 'Δώσε την ακτίνα'

ΔΙΑΒΑΣΕ Αριθμός

ΜΕΧΡΙΣ_ΟΤΟΥ Αριθμός>0

ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ

ΣΥΝΑΡΤΗΣΗ Εμβαδό_κύκλου (R) : **ΠΡΑΓΜΑΤΙΚΗ**

ΣΤΑΘΕΡΕΣ

Π=3.14

ΜΕΤΑΒΛΗΤΕΣ

ΠΡΑΓΜΑΤΙΚΕΣ: R

ΑΡΧΗ

Εμβαδό_κύκλου <- Π*R^2

ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ

ΔΙΑΔΙΚΑΣΙΑ Εκτύπωση (Αποτέλεσμα)

ΜΕΤΑΒΛΗΤΕΣ

ΠΡΑΓΜΑΤΙΚΕΣ : Αποτέλεσμα

ΑΡΧΗ

ΓΡΑΨΕ 'Το εμβαδό του κύκλου είναι :', Αποτέλεσμα

ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ

ΠΡΟΓΡΑΜΜΑ Ανταλλαγή

ΜΕΤΑΒΛΗΤΕΣ

ΑΚΕΡΑΙΕΣ: α, β, ι, κ, temp

ΑΡΧΗ

ΓΡΑΨΕ 'Δώσε τιμές για α και β'

ΔΙΑΒΑΣΕ α, β

temp <-- α

α <-- β

β <-- temp

ΓΡΑΨΕ 'α=', α, 'β=', β

ΓΡΑΨΕ 'Δώσε τιμές για ι και κ'

ΔΙΑΒΑΣΕ ι, κ

temp <-- ι

ι <-- κ

κ <-- temp

ΓΡΑΨΕ 'ι=', ι, 'κ=', κ

ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ Ανταλλαγή

Τμήμα προγράμματος που μπορεί να εκτελεστεί αυτοτελώς.

Μάλιστα επαναλαμβάνεται δύο φορές σε διαφορετικά σημεία στο πρόγραμμα.

ΠΡΟΓΡΑΜΜΑ Ανταλλαγή

ΜΕΤΑΒΛΗΤΕΣ

ΑΚΕΡΑΙΕΣ: α, β, ι, κ

ΑΡΧΗ

ΔΙΑΒΑΣΕ α, β

ΚΑΛΕΣΕ Εναλλαγή_τιμών(α,β)

ΓΡΑΨΕ α, β

ΔΙΑΒΑΣΕ ι, κ

ΚΑΛΕΣΕ Εναλλαγή_τιμών(ι,κ)

ΓΡΑΨΕ ι, κ

ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ Ανταλλαγή

Κύριο πρόγραμμα, που εκτελεί δύο φορές κατά σειρά τις εξής λειτουργίες:

- Διαβάζει σε δύο μεταβλητές, δύο ακέραιους αριθμούς.
- Εναλλάσσει τις τιμές μεταξύ των μεταβλητών (διαδοχική κλήση διαδικασίας).
- Εμφανίζει το περιεχόμενο των δύο μεταβλητών.

ΔΙΑΔΙΚΑΣΙΑ Εναλλαγή_τιμών(κ, λ)

ΜΕΤΑΒΛΗΤΕΣ

ΑΚΕΡΑΙΕΣ: κ, λ, temp

ΑΡΧΗ

temp <- κ

κ <- λ

λ <- temp

ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ Εναλλαγή_τιμών

Διαδικασία που εναλλάσσει τις τιμές δύο μεταβλητών (κ,λ) μέσω της βοηθητικής μεταβλητής temp.

10.5.3 Πραγματικές και τυπικές παράμετροι

Παράδειγμα:

Να γραφεί μια διαδικασία η οποία δέχεται στην είσοδο δύο τιμές και υπολογίζει και επιστρέφει το άθροισμα και τη διαφορά τους.

Οι μεταβλητές A, B, Διαφ1, Αθρ1, A,B, Διαφ2, Αθρ2 είναι μεταβλητές του προγράμματος Παράδειγμα_3 και αποτελούν τις **πραγματικές** παραμέτρους, ενώ οι μεταβλητές X,Y, Διαφορά, Άθροισμα είναι μεταβλητές της διαδικασίας Πράξεις, και ονομάζονται **τυπικές** παράμετροι.

Όλες οι μεταβλητές ισχύουν **τοπικά** μόνο για το τμήμα του προγράμματος στο οποίο έχουν δηλωθεί!

Μερικές γλώσσες προγραμματισμού ονομάζουν ορίσματα τις τυπικές παραμέτρους και απλά παραμέτρους τις πραγματικές



Η λίστα των τυπικών παραμέτρων (formal parameter list) καθορίζει τις παραμέτρους στη δήλωση του υποπρογράμματος.

Η λίστα των πραγματικών παραμέτρων (actual parameter list) καθορίζει τις παραμέτρους στην κλήση του υποπρογράμματος.

ΠΡΟΓΡΑΜΜΑ Παράδειγμα_3

...

A<-5

B<-7

ΚΑΛΕΣΕ Πράξεις (A, B, Διαφ1, Αθρ1)

...

A<-9

B<-6

ΚΑΛΕΣΕ Πράξεις (A, B, Διαφ2, Αθρ2)

...

ΔΙΑΔΙΚΑΣΙΑ Πράξεις (X, Y, Διαφορά, Άθροισμα)

ΜΕΤΑΒΛΗΤΕΣ

ΠΡΑΓΜΑΤΙΚΕΣ : X, Y, Διαφορά, Άθροισμα

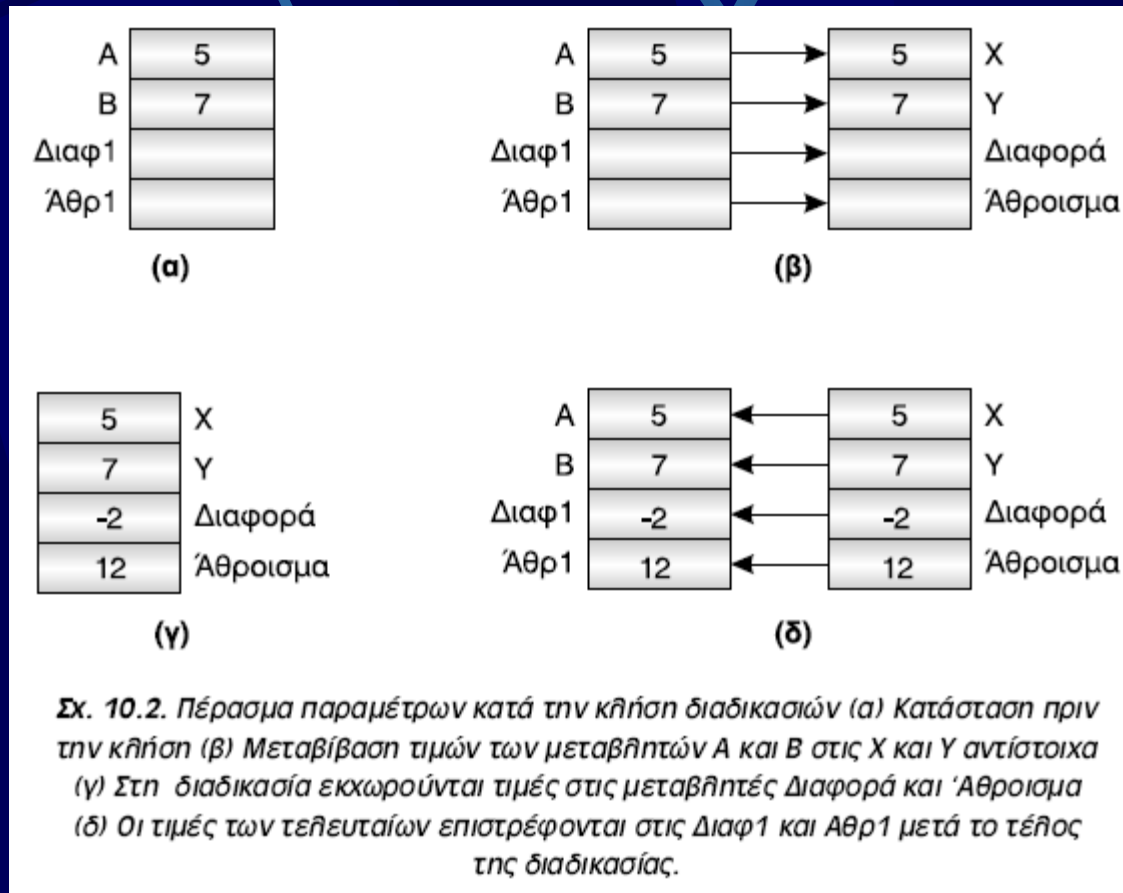
ΑΡΧΗ

Διαφορά <- X-Y

Άθροισμα <- X+Y

ΤΕΛΟΣ ΔΙΑΔΙΚΑΣΙΑΣ

10.5.3 Πραγματικές και τυπικές παράμετροι

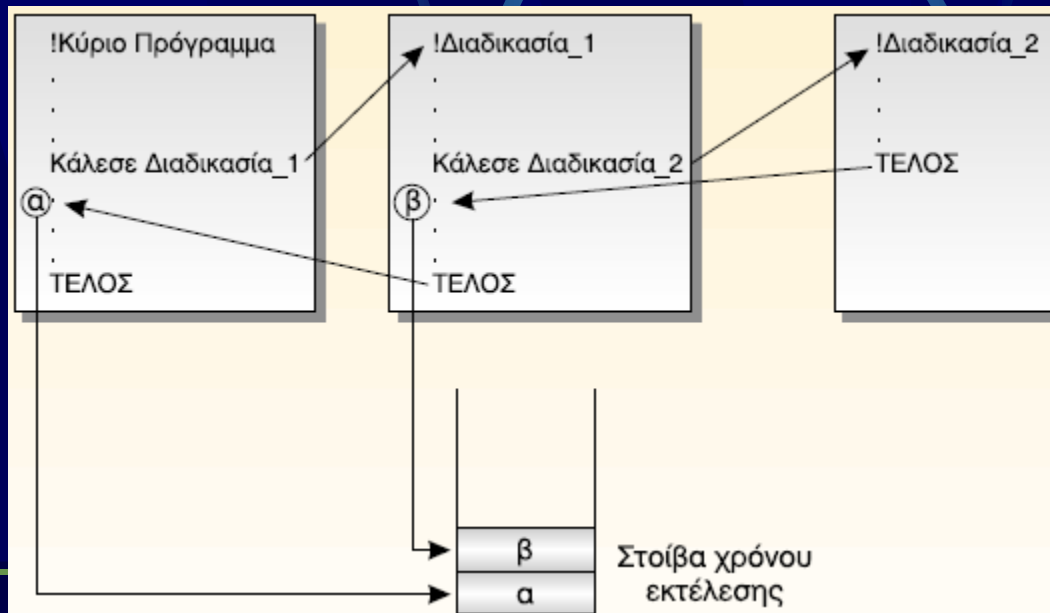


10.5.3 Πραγματικές και τυπικές παράμετροι

Η χρήση στοίβας στην κλήση υποπρογραμμάτων

Όταν μία διαδικασία ή συνάρτηση καλείται από το κύριο πρόγραμμα, τότε η αμέσως επόμενη διεύθυνση του κύριου προγράμματος, που ονομάζεται **διεύθυνση επιστροφής** (return address), αποθηκεύεται από το μεταφραστή σε μία στοίβα που ονομάζεται **στοίβα χρόνου εκτέλεσης** (execution time stack). Μετά την εκτέλεση της διαδικασίας ή της συνάρτησης η διεύθυνση επιστροφής απωθείται από τη στοίβα και έτσι ο έλεγχος του προγράμματος μεταφέρεται και πάλι στο κύριο πρόγραμμα. Η τεχνική αυτή εφαρμόζεται και γενικότερα, δηλαδή οποτεδήποτε μία διαδικασία ή συνάρτηση καλεί μία διαδικασία ή συνάρτηση. Για παράδειγμα, έστω ότι μία διαδικασία *a* καλεί τη διαδικασία *b*, που με τη σειρά της καλεί τη διαδικασία *c* κ.ο.κ. Στην περίπτωση αυτή οι διευθύνσεις επιστροφής εμφανίζονται στη στοίβα με σειρά *c*, *b*, *a*. Μετά την εκτέλεση κάθε διαδικασίας, η διεύθυνση επιστροφής απωθείται από τη στοίβα και ο έλεγχος μεταβιβάζεται στη διεύθυνση αυτή. Το παράδειγμα αυτό δείχνει μία από τις πολλές χρησιμότητες της LIFO ιδιότητας της στοίβας.

Η στοίβα χρόνου εκτέλεσης περιέχει τόσα στοιχεία όσα και τα υποπρογράμματα που έχουν κληθεί αλυσιδωτά



Στοιβά χρόνου εκτέλεσης

Πρόγραμμα Υποπρογράμματα

	Εντολή	Στοιβά χρόνου εκτέλεσης
...		
Αρχή		
100: εντολή1	100	
101: εντολή2	101	
102: Κάλυψε Δ1(...)	102	103
103: εντολή3	200	103
104: Κάλυψε Δ2(...)	201	202
105: $x \leftarrow \Sigma(\dots)$		103
106: εντολή4	400	202
107: ΤέλοςΠρογράμματος		103
...	401	202
Διαδικασία Δ1(...)		103
...	402	403
Αρχή		202
200: εντολή1		103
201: Κάλυψε Δ3(...)	500	403
202: εντολή2		202
203: ΤέλοςΔιαδικασίας		103
...	501	403
Διαδικασία Δ2(...)		202
...		103
Αρχή	502	403
300: εντολή1		202
301: εντολή2		103
302: ΤέλοςΔιαδικασίας	503	202
...		103
Διαδικασία Δ3(...)	404	103
...	202	103
Αρχή	203	
400: εντολή1	103	
401: εντολή2	104	105
402: Κάλυψε Δ4(...)	300	105
403: εντολή3	301	105
404: ΤέλοςΔιαδικασίας	302	
...	105	105
Διαδικασία Δ4(...)	600	105
...	601	105
Αρχή	602	
500: εντολή1	105	
501: εντολή2	106	
502: εντολή3	107	
503: ΤέλοςΔιαδικασίας		
...		
Συνάρτηση $\Sigma(\dots)$:...		
...		
Αρχή		
600: εντολή1		
601: $\Sigma \leftarrow \dots$		
602: ΤέλοςΣυνάρτησης		

Οι λίστες των παραμέτρων πρέπει να ακολουθούν τους εξής **κανόνες**:

- Ο αριθμός των πραγματικών και των τυπικών παραμέτρων πρέπει να είναι ίδιος.
- Κάθε πραγματική παράμετρος αντιστοιχεί στην τυπική παράμετρο που βρίσκεται στην αντίστοιχη θέση. Για παράδειγμα η πρώτη της λίστας των τυπικών παραμέτρων στην πρώτη της λίστας των πραγματικών παραμέτρων κοκ.
- Η τυπική παράμετρος και η αντίστοιχη της πραγματική πρέπει να είναι του ίδιου τύπου.

Όταν ένα υποπρόγραμμα πρέπει να:

- επιστρέψει παραπάνω από μία τιμές
- αλλάξει τις τιμές των πραγματικών παραμέτρων
- εκτελέσει λειτουργίες εισαγωγής (Διάβασε) ή εξαγωγής (Γράψε)

τότε επιλέγουμε να το υλοποιήσουμε με **ΔΙΑΔΙΚΑΣΙΑ** μιας και μία **ΣΥΝΑΡΤΗΣΗ**: επιστρέφει μία τιμή, δεν αλλάζει τις τιμές των πραγματικών παραμέτρων, δεν χρησιμοποιείται για λειτουργίες εισαγωγής / εξαγωγής και δεν μπορεί να καλέσει διαδικασία.

Η πραγματική μεταβλητή μιας διαδικασίας **ΔΕΝ** μπορεί να είναι σταθερά ή έκφραση.

10.6 Εμβέλεια μεταβλητών-σταθερών

Κάθε κύριο πρόγραμμα όπως και κάθε υποπρόγραμμα περιλαμβάνει τις δικές του μεταβλητές και σταθερές. Οι μεταβλητές αυτές στη ΓΛΩΣΣΑ είναι γνωστές στο αντίστοιχο υποπρόγραμμα που δηλώνονται και μόνο σε αυτό. Είναι δηλ. **τοπικές** στο συγκεκριμένο τμήμα προγράμματος. Ο μόνος τρόπος για να περάσει μία τιμή από ένα υποπρόγραμμα σε ένα άλλο ή από το κυρίως πρόγραμμα είναι δια μέσου των **παραμέτρων**.

Αφού όλες οι μεταβλητές είναι τοπικές, το ίδιο όνομα μεταβλητής μπορεί να εμφανίζεται σε διαφορετικά τμήματα προγράμματος, χωρίς να αντιστοιχεί στην ίδια μεταβλητή.

Ότι ισχύει για την εμβέλεια των μεταβλητών ισχύει και για τις συμβολικές σταθερές.

Εμβέλεια (scope) των μεταβλητών: το τμήμα του προγράμματος που ισχύουν οι μεταβλητές.



10.6 Εμβέλεια μεταβλητών-σταθερών

ΕΙΔΗ ΕΜΒΕΛΕΙΑΣ

α) Απεριόριστη εμβέλεια

όλες οι μεταβλητές και όλες οι σταθερές είναι γνωστές σε οποιοδήποτε τμήμα του προγράμματος και λέγονται **καθολικές** (global)

Μειονεκτήματα:

1. καταστρατηγεί την αρχή της αυτονομίας των υποπρογραμμάτων
2. ο καθένας που γράφει κάποιο υποπρόγραμμα πρέπει να γνωρίζει όλες τις καθολικές μεταβλητές

β) Περιορισμένη εμβέλεια

Όλες οι μεταβλητές που χρησιμοποιούνται σε ένα τμήμα προγράμματος, πρέπει να δηλώνονται σε αυτό το τμήμα και λέγονται **τοπικές** (local), ισχύουν δηλαδή για το υποπρόγραμμα στο οποίο δηλώθηκαν. Στη **ΓΛΩΣΣΑ** έχουμε περιορισμένη εμβέλεια. **Πλεονεκτήματα:** η απόλυτη αυτονομία όλων των υποπρογραμμάτων και η δυνατότητα να χρησιμοποιείται οποιοδήποτε όνομα.

γ) Μερικώς περιορισμένη εμβέλεια.

Άλλες μεταβλητές είναι **τοπικές** και άλλες **καθολικές**. Προσφέρει μερικά πλεονεκτήματα στον πεπειραμένο προγραμματιστή, αλλά για τον αρχάριο περιπλέκει το πρόγραμμα.

Πίνακας παρακολούθησης τιμών μεταβλητών

```

ΠΡΟΓΡΑΜΜΑ ΠΤ1
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: Α, Β, Γ
ΑΡΧΗ
  Α ← 3
  Β ← 13
  Γ ← 2
  ΓΡΑΨΕ Α, Β, Γ
  ΚΑΛΕΣΕ Διαδ (Β, Γ)
  ΓΡΑΨΕ Α, Β, Γ
  ΚΑΛΕΣΕ Διαδ (Γ, Α)
  ΓΡΑΨΕ Α, Β, Γ
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
! =====
ΔΙΑΔΙΚΑΣΙΑ Διαδ (α1, α2)
  ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ: α1, α2
  ΑΡΧΗ
    α1 ← α1 DIV 2
    α2 ← α2 ^ 3
  ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
    
```

A	B	Γ	Οθόνη
3	13	2	3, 13, 2
3	6	8	3, 6, 8
27	6	4	27, 6, 4

(B)	(Γ)
α1	α2
13	2
6	8

(Γ)	(Α)
α1	α2
8	3
4	27

Πίνακας παρακολούθησης τιμών μεταβλητών

```

ΠΡΟΓΡΑΜΜΑ ΠΤ3
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: Α, Β, Γ
ΑΡΧΗ
  Α ← 27
  Β ← 2
  Γ ← 13
  ΚΑΛΕΣΕ Διαδ (Α, Β, Γ)
  ΓΡΑΨΕ Α, Β, Γ
  ΚΑΛΕΣΕ Διαδ (Γ, Α, Β)
  ΓΡΑΨΕ Α, Β, Γ
  ΚΑΛΕΣΕ Διαδ (Β, Γ, Α)
  ΓΡΑΨΕ Α, Β, Γ
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
! =====
ΔΙΑΔΙΚΑΣΙΑ Διαδ (α1, α2, α3)
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: α1, α2, α3
ΑΡΧΗ
  α3 ← α1 + α2 - α3
  α2 ← α2 - α1
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
    
```

A	B	Γ	Οθόνη
27	2	13	
27	-25	16	27, -25, 16
11	68	16	11, 68, 16
73	68	-52	73, 68, -52

(Α)	(Β)	(Γ)
α1	α2	α3
27	2	13
27	-25	16

(Γ)	(Α)	(Β)
α1	α2	α3
16	27	-25
16	11	68

(Β)	(Γ)	(Α)
α1	α2	α3
68	16	11
68	-52	73

Πίνακας παρακολούθησης τιμών μεταβλητών

ΠΡΟΓΡΑΜΜΑ ΠΤ5

ΜΕΤΑΒΛΗΤΕΣ

ΑΚΕΡΑΙΕΣ: α, β, αποτ

ΑΡΧΗ

$\alpha \leftarrow 2$

$\beta \leftarrow 13$

$\text{αποτ} \leftarrow \text{Υπ}(\alpha, \beta)$

Γράψε αποτ

$\alpha \leftarrow 5 * \alpha + \text{αποτ}$

$\beta \leftarrow \text{A_T}(\text{αποτ} - 20)$

$\text{αποτ} \leftarrow \text{Υπ}(\beta, \alpha) - 3$

Γράψε α, β, αποτ

$\text{αποτ} \leftarrow \text{Υπ}(\alpha, \beta)$

Γράψε αποτ

ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ

! =====

Συνάρτηση Υπ(x, y): Ακέραια

ΜΕΤΑΒΛΗΤΕΣ

ΑΚΕΡΑΙΕΣ: x, y, π

ΑΡΧΗ

$x \leftarrow x + 1$

$y \leftarrow y - 1$

$\pi \leftarrow (x + y) \text{ div } x$

$\text{Υπ} \leftarrow 2 - \pi$

ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ

α	β	αποτ	Οθόνη
2	13	-	
2	13	-3	-3
7			
	23		
7	23	-2	7, 23, -2
		-1	-1

(α)	(β)	π	Υπ
x	y		
2	13	-	
3	12	5	-3

(β)	(α)	π	Υπ
x	y		
23	7	-	
24	6	1	1

(α)	(β)	π	Υπ
x	y		
7	23	-	
8	22	3	-1

Προγράμματα με υποπρογράμματα

Σύμφωνα με το νέο κώδικα οδικής κυκλοφορίας τα πρόστιμα που δίνονται για 3 είδη παραβάσεων, είναι: «κινητό» → 180 €, «κράνος» → 250 €, «ζώνη» → 300 €. Να γραφεί πρόγραμμα που δέχεται επαναληπτικά το είδος της παράβασης ('KIN' / 'KP' / 'Z') και σταματάει μόλις συμπληρωθούν 100 παραβάσεις ή συνολικό ποσό προστίμων που είναι τουλάχιστον 20000 €. Σε κάθε επανάληψη, να υπολογίζει με υποπρόγραμμα (το οποίο και να γράψετε) το ποσό του προστίμου και να το εμφανίζει. Να εμφανίζει το συνολικό αριθμό των παραβάσεων και το συνολικό ποσό των προστίμων. Να υλοποιήσετε υποπρόγραμμα που δέχεται το είδος της παράβασης και επιστρέφει το αντίστοιχο πρόστιμο.

Πρόγραμμα ΚΟΚ

Μεταβλητές

Ακέραιες: ΑΠ, ΣΠ, πρ

Χαρακτήρες: είδος

Αρχή

ΑΠ ← 0

ΣΠ ← 0

ΑρχήΕπανάληψης

Διάβασε είδος

πρ ← Πρόστιμο(είδος)

Γράψε πρ

ΑΠ ← ΑΠ + 1

ΣΠ ← ΣΠ + πρ

ΜέχριςΌτου ΑΠ = 100 Η ΣΠ >= 20000

Γράψε ΑΠ, ΣΠ

Τέλος_Προγράμματος

Συνάρτηση Πρόστιμο(είδος): Ακέραια

Μεταβλητές

Ακέραιες: π

Χαρακτήρες: είδος

Αρχή

Αν είδος = 'KIN' τότε

π ← 180

Αλλιώς Αν είδος = 'KP' τότε

π ← 250

Αλλιώς

π ← 300

ΤέλοςΑν

Πρόστιμο ← π

ΤέλοςΣυνάρτησης



Προγράμματα με υποπρογράμματα

Να γραφεί η Διαδικασία Εισαγωγή (Π, θ, ον) η οποία διαβάζει μία τιμή τύπου χαρακτήρα με έλεγχο εγκυρότητας ώστε να ανήκει στον Π[20] χαρακτήρων. Να επιστρέφει την τιμή (ον) και τη θέση θ(1-20) που εντοπίστηκε. Να γραφεί κύριο πρόγραμμα που διαβάζει τους Π[20] και Χ[20, 20] με τα ονόματα 20 πόλεων και τις χιλιομετρικές τους αποστάσεις και με τη βοήθεια της παραπάνω διαδικασίας διαβάζει δύο έγκυρα ονόματα πόλεων. Να εμφανίζει τη μεταξύ τους απόσταση.

Πρόγραμμα Απόσταση

...

Κάλεσε Εισαγωγή (Π, θ1, ον1)

Κάλεσε Εισαγωγή (Π, θ2, ον2)

Γράψε 'Η απόσταση των ', ον1, ' και ', ον2, ' είναι: ', Χ[θ1, θ2]

Διαδικασία Εισαγωγή (Π, θ, ον)

Μεταβλητές

Χαρακτήρες: Π[20], ον

Ακέραιες: θ, i

Λογική: βρ

Αρχή

ΑρχήΕπανάληψης

Διάβασε ον

i ← 1

βρ ← Ψευδής

Όσο i ≤ 20 ΚΑΙ βρ = Ψευδής επανάλαβε

Αν Π[i] = ον) τότε

βρ ← Αληθής

θ ← i

Αλλιώς

i ← i + 1

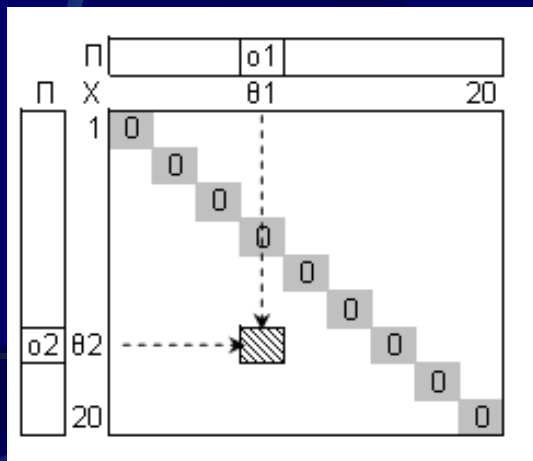
ΤέλοςΑν

ΤέλοςΕπανάληψης

ΜέχριςΌτου βρ = Αληθής

ΤέλοςΔιαδικασίας

	Bombay	Chennai	Delhi	Frankfurt	Gurgaon	Hanoi	Jakarta	Mumbai	New Delhi	Osaka	Paris	Seoul	Singapore	Tokyo	Yokohama
Bombay	321	483	520	531	466	899	1092	451	117	29	808	971	749	245	
Chennai	321	381	408	225	554	578	771	330	417	350	487	650	420	338	
Delhi	483	381	235	545	454	217	410	219	799	713	281	289	276	430	
Frankfurt	520	408	235	189	146	542	640	506	485	949	345	579	294	294	
Gurgaon	531	255	265	189	325	733	349	524	296	360	554	728	403	115	
Hanoi	466	554	424	146	335	334	534	360	631	695	219	393	148	440	
Jakarta	899	578	237	542	721	396	280	445	1016	920	189	159	248	838	
Mumbai	1092	771	410	650	849	514	280	578	1145	1121	307	321	366	574	
New Delhi	451	330	332	506	524	360	445	578	747	680	271	426	212	629	
Osaka	117	417	739	485	296	621	1636	1345	747	113	850	1024	779	230	
Paris	29	350	711	549	360	695	920	1121	680	113	837	1000	778	244	
Singapore	808	487	231	345	554	219	189	307	271	620	837	166	71	659	
Tokyo	971	650	281	579	728	393	159	731	426	1024	1000	166	345	873	
Yokohama	749	420	234	294	403	148	248	336	212	779	778	71	245	599	



Μετατροπή: Συνάρτηση – Διαδικασία – Χωρίς υποπρόγραμμα

Γενικά:

Πρόγραμμα Π1 ... $\alpha \leftarrow \Sigma(x_1, x_2, \dots, x_n)$... ΤέλοςΠρογράμματος	\Longrightarrow	Πρόγραμμα Π2 ... Κάλεσε $\Delta(x_1, x_2, \dots, x_n, y)$ $\alpha \leftarrow y$... ΤέλοςΠρόγραμματος
Συνάρτηση $\Sigma(x_1, x_2, \dots, x_n)$: Τύπος ... $\Sigma \leftarrow$ αποτέλεσμα ... ΤέλοςΣυνάρτησης	\Longrightarrow	Διαδικασία $\Delta(x_1, x_2, \dots, x_n, x_{n+1})$... $x_{n+1} \leftarrow$ αποτέλεσμα ... ΤέλοςΔιαδικασίας

π.χ.

Πρόγραμμα Με_Συνάρτηση	Πρόγραμμα Με_Διαδικασία	Πρόγραμμα Χωρίς_Υποπρόγραμμα
... Διάβασε A, B $\Gamma \leftarrow 0$ Όσο $(B \neq 0)$ επανάλαβε $\Gamma \leftarrow \Gamma + \text{Συν}(A, B)$ $A \leftarrow 2 * A$ $B \leftarrow B \text{ div } 2$ ΤέλοςΕπανάληψης Γράψε Γ	... Διάβασε A, B $\Gamma \leftarrow 0$ Όσο $(B \neq 0)$ επανάλαβε Κάλεσε Διαδ(A, B, Ω) $\Gamma \leftarrow \Gamma + \Omega$ $A \leftarrow 2 * A$ $B \leftarrow B \text{ div } 2$ ΤέλοςΕπανάληψης Γράψε Γ	... Διάβασε A, B $\Gamma \leftarrow 0$ Όσο $(B \neq 0)$ επανάλαβε Αν $(B \text{ mod } 2 \neq 0)$ τότε $\zeta \leftarrow A$ Αλλιώς $\zeta \leftarrow 0$ ΤέλοςΑν $\Omega \leftarrow \zeta$ $\Gamma \leftarrow \Gamma + \Omega$ $A \leftarrow 2 * A$ $B \leftarrow B \text{ div } 2$ ΤέλοςΕπανάληψης Γράψε Γ
Συνάρτηση $\text{Συν}(x, \psi)$: Ακέραια ... Αν $(\psi \text{ mod } 2 \neq 0)$ τότε $\zeta \leftarrow x$ Αλλιώς $\zeta \leftarrow 0$ ΤέλοςΑν $\text{Συν} \leftarrow \zeta$	Διαδικασία Διαδ(x, ψ, ω) ... Αν $(\psi \text{ mod } 2 \neq 0)$ τότε $\zeta \leftarrow x$ Αλλιώς $\zeta \leftarrow 0$ ΤέλοςΑν $\omega \leftarrow \zeta$	

Μετατροπή: Συνάρτηση – Διαδικασία – Χωρίς υποπρόγραμμα

π.χ. περίπτωση συνάρτησης που αλλάζει τις τιμές των τυπικών παραμέτρων → αντίγραφα τυπικών παραμέτρων

```

ΣΥΝΑΡΤΗΣΗ Τιμή (A, B): ΛΟΓΙΚΗ
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: A, B
ΑΡΧΗ
  A ← A+8
  B ← B ^3
  AN (A + B) mod 2 = 0 ΤΟΤΕ
    Τιμή ← ΑΛΗΘΗΣ
  ΑΛΛΙΩΣ
    Τιμή ← ΨΕΥΔΗΣ
  ΤΕΛΟΣ_ΑΝ
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
    
```

```

ΔΙΑΔΙΚΑΣΙΑ Τιμή (A, B, Γ)
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: A, B, A2, B2
  ΛΟΓΙΚΕΣ: Γ
    
```

```

ΑΡΧΗ
  A2 ← A
  B2 ← B
  → A2 ← A2 + 8
  B2 ← B2 ^3
  AN (A2 + B2) mod 2 = 0 ΤΟΤΕ
    Γ ← ΑΛΗΘΗΣ
  ΑΛΛΙΩΣ
    Γ ← ΨΕΥΔΗΣ
  ΤΕΛΟΣ_ΑΝ
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
    
```

```

a ← f1(x) + f2(y)
⇔ με διαδικασίες:
ΚΑΛΕΣΕ D1(x, a1)
ΚΑΛΕΣΕ D2(y, a2)
a ← a1 + a2
    
```

Διάφορα

1. Αν η ακέραια συνάρτηση f δέχεται σαν παράμετρο μία ακέραια τιμή τότε η εντολή: Γράψε $f(f(2))$ είναι συντακτικά σωστή (Σωστό/Λάθος);. Ισχύει το ίδιο για λογική συνάρτηση f ;
2. Έστω η συνάρτηση Sum η οποία δέχεται 2 πραγματικούς και επιστρέφει το άθροισμά τους. Συμπληρώστε την παρακάτω εντολή (χωρίς χρήση των τελεστών +, -) ώστε να εκχωρείται στη μεταβλητή x το άθροισμα των πραγματικών μεταβλητών a, b, c :

$$x \leftarrow \dots\dots\dots$$
3. Έστω η συνάρτηση $\max(\alpha, \beta)$: Πραγματική η οποία επιστρέφει τη μεγαλύτερη τιμή των πραγματικών α, β . Συμπληρώστε τις παρακάτω εντολές ώστε να εκχωρείται στη μεταβλητή \min η μικρότερη τιμή των α, β .

$$\min \leftarrow \dots - \max(\alpha, \beta) \qquad \min \leftarrow -\max(\dots, \dots)$$

Συμπληρώστε το κενό ώστε να εκχωρείται στη μεταβλητή $\max4$ η μεγαλύτερη τιμή των $\alpha, \beta, \gamma, \delta$: $\max4 \leftarrow \dots\dots$

Μετατροπή Διαδικασίας σε Συνάρτηση/Συναρτήσεις

π.χ. Να μετατραπούν τα παρακάτω με Συναρτήσεις:

Πρόγραμμα Άσκηση
Μεταβλητές
Ακέραιες: A[100], i, min, max
Αρχή
για i από 1 μέχρι 100
Διάβασε A[i]
ΤέλοςΕπανάληψης
Κάλεσε MinMax(A, min, max)
Γράψε max-min
ΤέλοςΠρογράμματος

Διαδικασία MinMax(A, min, max)
Μεταβλητές
Ακέραιες: A[100], i, min, max
Αρχή
min \leftarrow A[1]
για i από 2 μέχρι 100
Αν A[i] < min τότε
min \leftarrow A[i]
ΤέλοςΑν
ΤέλοςΕπανάληψης
max \leftarrow A[1]
για i από 2 μέχρι 100
Αν A[i] > max τότε
max \leftarrow A[i]
ΤέλοςΑν
ΤέλοςΕπανάληψης
ΤέλοςΔιαδικασίας



Πρόγραμμα Άσκηση
Μεταβλητές
Ακέραιες: A[100], i, min, max
Αρχή
για i από 1 μέχρι 100
Διάβασε A[i]
ΤέλοςΕπανάληψης
min \leftarrow Minimum(A)
max \leftarrow Maximum(A)
Γράψε max-min
ΤέλοςΠρογράμματος

Συνάρτηση Minimum(A):Ακέραιοι
Μεταβλητές
Ακέραιες: A[100], i, min
Αρχή
min \leftarrow A[1]
για i από 2 μέχρι 100
Αν A[i] < min τότε
min \leftarrow A[i]
ΤέλοςΑν
ΤέλοςΕπανάληψης
Minimum \leftarrow min
ΤέλοςΣυνάρτησης

Συνάρτηση Maximum(A):Ακέραιοι
Μεταβλητές
Ακέραιες: A[100], i, min
Αρχή
max \leftarrow A[1]
για i από 2 μέχρι 100
Αν A[i] > max τότε
max \leftarrow A[i]
ΤέλοςΑν
ΤέλοςΕπανάληψης
Maximum \leftarrow max
ΤέλοςΣυνάρτησης

Πίνακας παρακολούθησης τιμών μεταβλητών

```

1  ΠΡΟΓΡΑΜΜΑ  kef10_Alysidwth_klhsh
2  ΜΕΤΑΒΛΗΤΕΣ
3      ΑΚΕΡΑΙΕΣ: x, y, z, A[3], i
4  ΑΡΧΗ
5      x <-- 1
6      y <-- 2
7      z <-- 3
8  ΓΙΑ i ΑΠΟ 1 ΜΕΧΡΙ 3
9      Α[i] <-- i+3
10  ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
11  ΓΡΑΨΕ x, y, z, A[1], A[2], A[3]
12  ΚΑΛΕΣΕ D1(y, z, x)
13  ΓΡΑΨΕ x, y, z
14  ΚΑΛΕΣΕ D1(A[2], A[1], A[3])
15  ΓΡΑΨΕ A[1], A[2], A[3]
16  ΚΑΛΕΣΕ D2(A[2], z, A[1])
17  ΓΡΑΨΕ x, y, z, A[1], A[2], A[3]
18  ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
19  ! -----
20  ΔΙΑΔΙΚΑΣΙΑ D1(p, q, r)
21  ΜΕΤΑΒΛΗΤΕΣ
22      ΑΚΕΡΑΙΕΣ: p, q, r
23  ΑΡΧΗ
24      p <-- p + 2
25      q <-- q - 2
26      r <-- r * 7
27      ΚΑΛΕΣΕ D2(r, p, q)
28  ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
29  ! -----
30  ΔΙΑΔΙΚΑΣΙΑ D2(a, b, c)
31  ΜΕΤΑΒΛΗΤΕΣ
32      ΑΚΕΡΑΙΕΣ: a, b, c
33  ΑΡΧΗ
34      a <-- a div 3
35      b <-- b mod 9
36      c <-- c^2
37  ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ

```

x	y	z	A[1]	A[2]	A[3]	Οθόνη
1	2	3	4	5	6	1,2,3,4,5,6
2	4	1				2,4,1
			4	7	14	4,7,14
			1	16	2	2,4,1,16,2,14

D1

y	z	x
p	q	r
2	3	1
4	1	7
4	1	2

D2

r	p	q
a	b	c
7	4	1
2	4	1

D1

A[2]	A[1]	A[3]
p	q	r
5	4	6
7	2	42
7	4	14

D2

r	p	q
a	b	c
42	7	2
14	7	4

D2

A[2]	z	A[1]
a	b	c
7	1	4
2	1	16

1	2	3	4	5	6
2	4	1			
4	7	14			
2	4	1	16	2	14

```

1 ΠΡΟΓΡΑΜΜΑ kefl0_pinakas_parametros
2 ΜΕΤΑΒΛΗΤΕΣ
3   ΑΚΕΡΑΙΕΣ: A[10], i
4 ΑΡΧΗ
5   ΚΑΛΕΣΕ ReadArray(A)
6   ΚΑΛΕΣΕ PrintArrayInverse(A)
7   ΓΡΑΨΕ 'Άθροισμα = ', SumArray(A)
8 ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
9 ! -----
10 ΔΙΑΔΙΚΑΣΙΑ ReadArray(A)
11 ΜΕΤΑΒΛΗΤΕΣ
12   ΑΚΕΡΑΙΕΣ: A[10], i
13 ΑΡΧΗ
14   ΓΙΑ i ΑΠΟ 1 ΜΕΧΡΙ 10
15     ΔΙΑΒΑΣΕ A[i]
16     ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
17 ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
18 ! -----
19 ΔΙΑΔΙΚΑΣΙΑ PrintArrayInverse(A)
20 ΜΕΤΑΒΛΗΤΕΣ
21   ΑΚΕΡΑΙΕΣ: A[10], i
22 ΑΡΧΗ
23   ΓΙΑ i ΑΠΟ 10 ΜΕΧΡΙ 1 ΜΕ ΒΗΜΑ -1
24     ΓΡΑΨΕ A[i]
25     ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
26 ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
27 ! -----
28 ΣΥΝΑΡΤΗΣΗ SumArray(A): ΑΚΕΡΑΙΑ
29 ΜΕΤΑΒΛΗΤΕΣ
30   ΑΚΕΡΑΙΕΣ: A[10], i, s
31 ΑΡΧΗ
32   s <-- 0
33   ΓΙΑ i ΑΠΟ 1 ΜΕΧΡΙ 10
34     s <-- s + A[i]
35 ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
36   SumArray <-- s
37 ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ

```

Πίνακες ως παράμετροι

Πρόγραμμα που με κατάλληλα υποπρόγραμματα: α) διαβάζει ακέραιο A[10] β) τον εμφανίζει ανάποδα γ) υπολογίζει το άθροισμα των στοιχείων του

Συνάρτηση που παίρνει ως παραμέτρους 2 αλφαριθμητικά και τα συγκρίνει.

Συνάρτηση Compare(x, y): ...

Μεταβλητές

Χαρακτήρες: x, y

Αρχή

Αν $x < y$ τότε

... ← ...

Αλλιώς Αν $x > y$ τότε

... ← ...

Αλλιώς

... ← ...

Τέλος Αν

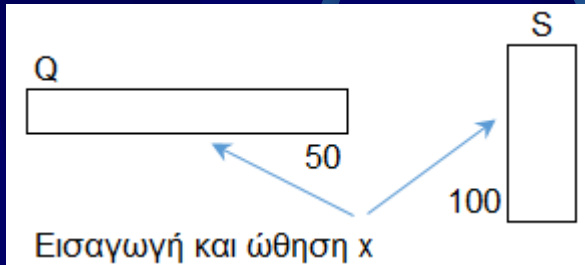
Τέλος Συνάρτησης

Να γράφει υποπρόγραμμα το οποίο...

- "διαβάζει/δέχεται σαν είσοδο" (Διάβασε: διαδικασία)
- "γράφει/εμφανίζει" (Γράψε: διαδικασία)
- "δέχεται" (παράμετρος: διαδικασία ή συνάρτηση)
- "επιστρέφει" (παράμετρος: διαδικασία ή συνάρτηση για μία τιμή)

Στοίβα και Ουρά - συνδυασμοί

Έστω ακέραια στοίβα $S[100]$ με δείκτη top και ακέραια ουρά $Q[50]$ με δείκτες f και r . Να γίνουν διαδικασίες που επιτελούν τα παρακάτω ζευγάρια πράξεων (εφόσον γίνονται και οι δύο). Να επιστρέφουν μέσω λογικής μεταβλητής Αληθής/Ψευδής αναλόγως εάν έγιναν και οι δύο πράξεις ή όχι.



Διαδικασία ΕισαγωγήΩθηση($Q, f, r, S, top, x, done$)

Μεταβλητές

Ακέραιες: $Q[50], f, r, S[100], top, x$

Λογικές: $done$

Αρχή

Αν $r=50$ Η $top=100$ τότε

$done \leftarrow$ Ψευδής

Αλλιώς

Αν $f=0$ ΚΑΙ $r=0$ τότε

$f \leftarrow 1$

$r \leftarrow 1$

Αλλιώς

$r \leftarrow r + 1$

ΤέλοςΑν

$Q[r] \leftarrow x$

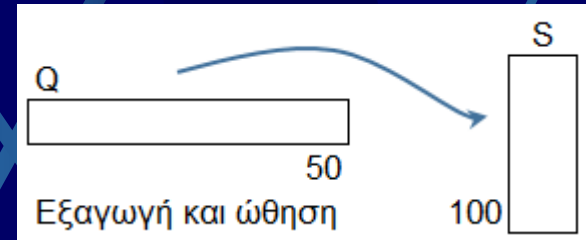
$top \leftarrow top + 1$

$S[top] \leftarrow x$

$done \leftarrow$ Αληθής

ΤέλοςΑν

ΤέλοςΔιαδικασίας



Διαδικασία ΕξαγωγήΩθηση($Q, f, r, S, top, done$)

Μεταβλητές

Ακέραιες: $Q[50], f, r, S[100], top$

Λογικές: $done$

Αρχή

Αν $f=0$ ΚΑΙ $r=0$ Η $top=100$ τότε

$done \leftarrow$ Ψευδής

Αλλιώς

$x \leftarrow Q[f]$

Αν $f=r$ τότε

$f \leftarrow 0$

$r \leftarrow 0$

Αλλιώς

$f \leftarrow f + 1$

ΤέλοςΑν

$top \leftarrow top + 1$

$S[top] \leftarrow x$

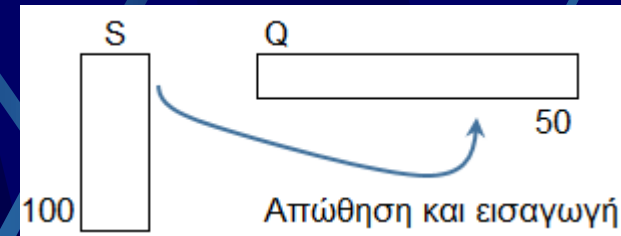
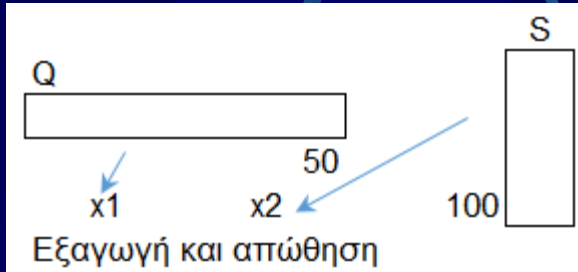
$done \leftarrow$ Αληθής

ΤέλοςΑν

ΤέλοςΔιαδικασίας

Στοίβα και Ουρά - συνδυασμοί

Έστω ακέραια στοίβα $S[100]$ με δείκτη top και ακέραια ουρά $Q[50]$ με δείκτες f και r . Να γίνουν διαδικασίες που επιτελούν τα παρακάτω ζευγάρια πράξεων (εφόσον γίνονται και οι δύο). Να επιστρέφουν μέσω λογικής μεταβλητής Αληθής/Ψευδής αναλόγως εάν έγιναν και οι δύο πράξεις ή όχι.



Διαδικασία ΕξαγωγήΑπόθεση($Q, f, r, x1, S, top, x2, done$)

Μεταβλητές

Ακέραιες: $Q[50], f, r, S[100], top, x1, x2$

Λογικές: $done$

Αρχή

Αν $f=0$ ΚΑΙ $r=0$ Η $top=0$ τότε

$done \leftarrow$ Ψευδής

Αλλιώς

$x1 \leftarrow Q[f]$

Αν $f=r$ τότε

$f \leftarrow 0$

$r \leftarrow 0$

Αλλιώς

$f \leftarrow f + 1$

ΤέλοςΑν

$x2 \leftarrow S[top]$

$top \leftarrow top - 1$

$done \leftarrow$ Αληθής

ΤέλοςΑν

ΤέλοςΔιαδικασίας

Διαδικασία ΑπόθεσηΕισαγωγή($Q, f, r, S, top, done$)

Μεταβλητές

Ακέραιες: $Q[50], f, r, S[100], top, x$

Λογικές: $done$

Αρχή

Αν $top=0$ Η $r=50$ τότε

$done \leftarrow$ Ψευδής

Αλλιώς

$x \leftarrow S[top]$

$top \leftarrow top - 1$

Αν $f=0$ ΚΑΙ $r=0$ τότε

$f \leftarrow 1$

$r \leftarrow 1$

Αλλιώς

$r \leftarrow r + 1$

ΤέλοςΑν

$Q[r] \leftarrow x$

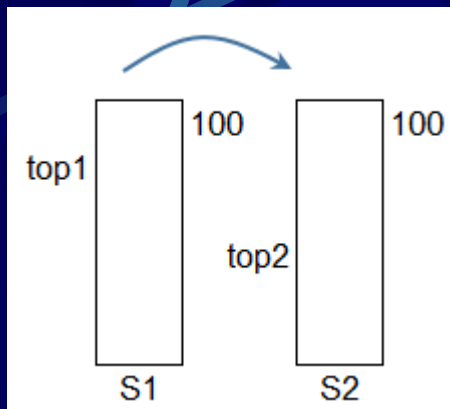
$done \leftarrow$ Αληθής

ΤέλοςΑν

ΤέλοςΔιαδικασίας

Στοιίβα και Ουρά - συνδυασμοί

Έστω δύο ακέραιες στοιίβες $S1[100]$ και $S2[100]$ με δείκτες $top1$ και $top2$ αντίστοιχα. Να γραφεί διαδικασία απωθεί ένα στοιχείο από την $S1$ και την ωθεί στην $S2$ (εφόσον γίνονται και τα δύο). Να επιστρέφει μέσω λογικής μεταβλητής Αληθής/Ψευδής αναλόγως εάν έγιναν και οι δύο πράξεις ή όχι.



Διαδικασία Απώθηση/Ωθηση($S1, top1, S2, top2, done$)

Μεταβλητές

Ακέραιες: $S1[100], top1, S2[100], top2$

Λογικές: $done$

Αρχή

Αν $top1=0$ Η $top2=100$ τότε

$done \leftarrow$ Ψευδής

Αλλιώς

$done \leftarrow$ Αληθής

$top2 \leftarrow top2 + 1$

$S2[top2] \leftarrow S1[top1]$

$top1 \leftarrow top1 - 1$

ΤέλοςΑν

ΤέλοςΔιαδικασίας

Έστω δύο ακέραιες ουρές $Q1[100]$ και $Q2[100]$ με δείκτες $front1, rear1$ και $front2, rear2$ αντίστοιχα. Να γραφεί διαδικασία εξαγάγει ένα στοιχείο από την $Q1$ και την εισάγει στην $Q2$ (εφόσον γίνονται και τα δύο). Να επιστρέφει μέσω λογικής μεταβλητής Αληθής/Ψευδής αναλόγως εάν έγιναν και οι δύο πράξεις ή όχι.

Διαδικασία Εξαγωγή/Εισαγωγή($Q1, front1, rear1, Q2, front2, rear2, done$)

Μεταβλητές

Ακέραιες: $Q1[100], front1, rear1, Q2[100], front2, rear2, x$

Λογικές: $done$

Αρχή

Αν ($front1=0$ ΚΑΙ $rear1=0$) Η $rear2=100$ τότε

$done \leftarrow$ Ψευδής

Αλλιώς

$done \leftarrow$ Αληθής

$x \leftarrow Q1[front1]$

Αν $front1=rear1$ τότε

$front1 = 0$

$rear1 = 0$

Αλλιώς

$front1 = front1 + 1$

ΤέλοςΑν

Αν $front2=0$ ΚΑΙ $rear2=0$ τότε

$front2 \leftarrow 1$

$rear2 \leftarrow 1$

Αλλιώς

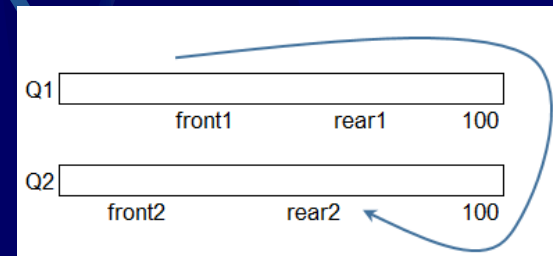
$rear2 \leftarrow rear2 + 1$

ΤέλοςΑν

$Q2[rear2] \leftarrow x$

ΤέλοςΑν

ΤέλοςΔιαδικασίας



Προγράμματα με υποπρογράμματα

Ένα παιχνίδι τένις, χωρίζεται σε πόντους (points), games και sets. Κερδίζει αυτός που θα πάρει τρία σετ. Ένα set αποτελείται από games (τουλάχιστον έξι), τα οποία με τη σειρά τους αποτελούνται από πόντους. Λέμε ότι ένας τενίστας πήρε το σετ, όταν έχει κερδίσει έξι ή επτά games, με διαφορά δύο games από τον αντίπαλό του (πχ 6-3, 6-4 ή 7-5). Αν το σετ είναι ισόπαλο 6-6 games, οι δύο μονομάχοι λύνουν τις διαφορές τους στο λεγόμενο tie break που αποτελείται από πόντους και για να το κερδίσει κάποιος θα πρέπει να φτάσει πρώτος στους 7 τουλάχιστον πόντους, με διαφορά δύο πόντων πχ 7-5. Αν στο tie break υπάρχει ισοπαλία 6-6, τότε αυτό κρίνεται στους οχτώ πόντους και πάει λέγοντας. Ένα game αποτελείται από σειρά πόντων. Για να κερδίσει ο παίκτης ένα game, πρέπει να πάρει τουλάχιστον 4 πόντους με ένα περιθώριο δύο ή περισσότερων πόντων απ'τον αντίπαλό του. Αν το game πάει στο 4-4 θα κερδίσει αυτός που θα προηγηθεί με δύο πόντους διαφορά. α) να γραφεί η διαδικασία Game η οποία προσομοιώνει ένα game διαβάζοντας επαναληπτικά τους πόντους των δύο παικτών και επιστρέφοντας τον νικητή 1/2 β) να γραφεί η διαδικασία Set η οποία προσομοιώνει ένα set καλώντας επαναληπτικά τη διαδικασία Game και αν χρειασθεί τους πόντους στο tie break. Να επιστρέφει τον νικητή 1/2 γ) κύριο πρόγραμμα το οποίο διαβάζει τα ονόματα των δύο παικτών και προσομοιώνει ένα παιχνίδι τένις καλώντας επαναληπτικά τη διαδικασία Set. Να εμφανίζει το όνομα του νικητή.



Πρόγραμμα Tennis

Μεταβλητές

Χαρακτήρες: on1, on2

Ακέραιες: s1, s2, winner

Αρχή

Διάβασε on1, on2

s1 <-- 0 s2 <-- 0

ΑρχήΕπανάληψης

Κάλεσε Set(winner)

Αν winner=1 τότε

s1 <-- s1 + 1

Αλλιώς

s2 <-- s2 + 1

ΤέλοςΑν

ΜέχριςΌτου s1=3 Η s2=3

Αν s1=3 τότε

Γράψε on1

Αλλιώς

Γράψε on2

ΤέλοςΑν

ΤέλοςΠρογράμματος

Διαδικασία Set(winner)

Μεταβλητές

Ακέραιες:

winner, w, g1, g2, min, max, p1, p2, p

Αρχή

g1 <-- 0 g2 <-- 0

ΑρχήΕπανάληψης

Κάλεσε Game(w)

Αν w=1 τότε

g1 <-- g1 + 1

Αλλιώς

g2 <-- g2 + 1

ΤέλοςΑν

max <-- g1

min <-- g2

Αν g2 > g1 τότε

max <-- g2

min <-- g1

ΤέλοςΑν

ΜέχριςΌτου max>=6 ΚΑΙ max-

min>=2 Η min=6 ΚΑΙ max=6

Αν g1 > g2 τότε

winner <-- 1

ΑλλιώςΑν g2 > g1 τότε

winner <-- 2

Αλλιώς ! tie break

p1 <-- 0 p2 <-- 0

ΑρχήΕπανάληψης

Διάβασε p

Αν p=1 τότε

p1 <-- p1 + 1

Αλλιώς

p2 <-- p2 + 1

ΤέλοςΑν

max <-- p1

min <-- p2

Αν p2 > p1 τότε

max <-- p2

min <-- p1

ΤέλοςΑν

ΜέχριςΌτου max>=7 ΚΑΙ max-min>=2

Αν p1 > p2 τότε

winner <-- 1

Αλλιώς

winner <-- 2

ΤέλοςΑν

ΤέλοςΔιαδικασίας



Να γραφεί Διαδικασία ισοδύναμη με την παρακάτω Συνάρτηση:

Διαδικασία Game(winner)

Μεταβλητές

Ακέραιες:

winner, p1, p2, p, min, max

Αρχή

p1 \leftarrow 0 p2 \leftarrow 0

ΑρχήΕπανάληψης

Διάβασε p

Αν p=1 τότε

p1 \leftarrow p1 + 1

Αλλιώς

p2 \leftarrow p2 + 1

ΤέλοςΑν

max \leftarrow p1

min \leftarrow p2

Αν p2 > p1 τότε

max \leftarrow p2

min \leftarrow p1

ΤέλοςΑν

ΜέχριςΌτου max \geq 4 ΚΑΙ

max-min \geq 2

Αν p1 > p2 τότε

winner \leftarrow 1

Αλλιώς

winner \leftarrow 2

ΤέλοςΑν

ΤέλοςΔιαδικασίας

Συνάρτηση MO(A):Πραγματική

Μεταβλητές

Ακέραιες: A[100], i, j, tmp, s

Αρχή

για i από 2 μέχρι 100

για j από 100 μέχρι i μεβήμα -1

Αν A[j-1] > A[j] τότε

tmp \leftarrow A[j-1]

A[j-1] \leftarrow A[j]

A[j] \leftarrow tmp

ΤέλοςΑν

ΤέλοςΕπανάληψης

ΤέλοςΕπανάληψης

s \leftarrow 0

για i από 91 μέχρι 100

s \leftarrow s + A[i]

ΤέλοςΕπανάληψης

MO \leftarrow s/10

ΤέλοςΣυνάρτησης

Διαδικασία Δ(A, MO)

Μεταβλητές

Ακέραιες: A[100], A2[100] i, j, tmp, s

Πραγματικές: MO

Αρχή

για i από 1 μέχρι 100

A2[i] \leftarrow A[i]

ΤέλοςΕπανάληψης

για i από 2 μέχρι 100

για j από 100 μέχρι i μεβήμα -1

Αν A2[j-1] > A2[j] τότε

tmp \leftarrow A2[j-1]

A2[j-1] \leftarrow A2[j]

A2[j] \leftarrow tmp

ΤέλοςΑν

ΤέλοςΕπανάληψης

ΤέλοςΕπανάληψης

s \leftarrow 0

για i από 91 μέχρι 100

s \leftarrow s + A2[i]

ΤέλοςΕπανάληψης

MO \leftarrow s/10

ΤέλοςΣυνάρτησης



Να γράψετε Συνάρτηση ισοδύναμη με την παρακάτω Διαδικασία:

```
Διαδικασία Δ(A, max)
Μεταβλητές
  Ακέραιες: A[100], max, i
Αρχή
  max <-- A[1]
  για i από 2 μέχρι 100
    Αν A[i] > max τότε
      max <--A[i]
  ΤέλοςΑν
  ΤέλοςΕπανάληψης
ΤέλοςΔιαδικασίας
```

```
Συνάρτηση Σ(A): Ακέραια
Μεταβλητές
  Ακέραιες: A[100], max, i
Αρχή
  max <-- A[1]
  για i από 2 μέχρι 100
    Αν A[i] > max τότε
      max <--A[i]
  ΤέλοςΑν
  ΤέλοςΕπανάληψης
  Σ <-- max
ΤέλοςΣυνάρτησης
```



Δίνονται οι συναρτήσεις μετατροπής: Celsius2Fahrenheit(c) και Fahrenheit2Kelvin(f)
Υλοποιήστε με τη βοήθεια αυτών τη συνάρτηση Celsius2Kelvin(c):Fahrenheit2Kelvin(Celsius2Fahrenheit(c)).....

Να γραφεί υποπρόγραμμα που δέχεται μέσω πίνακα τις τιμές 10 αντιστάσεων (μη μηδενικές) και επιστρέφει τη συνολική αντίσταση αν συνδεθούν σε σειρά (σύνολο των αντιστάσεων) και παράλληλα (σύνολο των αντίστροφων τιμών των αντιστάσεων)

```
Διαδικασία ΣυνολικήΑντίσταση(R_array, R_seira, R_parallhla)
...
R_seira <-- 0
R_parallhla <-- 0
για i από 1 μέχρι 10
  R_seira <-- R_seira + R_array[i]
  R_parallhla <-- R_parallhla + 1/R_array[i]
ΤέλοςΕπανάληψης
```

Παραδείγματα υποπρογραμμάτων

Σε πραγματικό πίνακα A[100] εύρεση
min(x=0)/max(x=1)
Συνάρτηση MinMax(A, x): Πραγματική

```
...
m <-- A[1]
για i από 2 μέχρι 100
  Αν (x=0 ΚΑΙ A[i]<m) Η (x=1 ΚΑΙ A[i]>m)
  τότε
    m <-- A[i]
  ΤέλοςΑν
ΤέλοςΕπανάληψης
MinMax <-- m
ΤέλοςΣυνάρτησης
```

Σε πραγματικό πίνακα A[100] εύρεση
αθροίσματος άρτιων(x=0)/περιττών(x=1)
Συνάρτηση SumArtiwnPerittwn(A, x):
Πραγματική

```
...
sum <-- 0
για i από 1 μέχρι 100
  Αν (x=0 ΚΑΙ A[i] mod 2 = 0) Η (x=1 ΚΑΙ
A[i] mod 2 <> 0) τότε
    sum <-- sum + A[i]
  ΤέλοςΑν
ΤέλοςΕπανάληψης
SumArtiwnPerittwn <-- sum
ΤέλοςΣυνάρτησης
```

Έλεγχος εγκυρότητας για εισαγωγή πραγματικού x σε
διάστημα [a, b]

Διαδικασία ΕισαγωγήΜεΕγκυρότητα(x, a, b)

```
...
ΑρχήΕπανάληψης
  Διάβασε x
  ΜέχριςΌτου x >= a ΚΑΙ x <= b
ΤέλοςΔιαδικασίας
```

Σε πραγματικό πίνακα A[100] σειριακή αναζήτηση του x ξεκινώντας από
την αρχή(s=0)/τέλος(s=1)

Συνάρτηση Αναζήτηση(A, x, s): Ακέραια

```
...
Αν s = 0 τότε
  i <-- 1
Αλλιώς
  i <-- 100
ΤέλοςΑν
βρ <-- Ψευδής
θέση <-- -1
Όσο (s=0 ΚΑΙ i<=100 Η s=1 ΚΑΙ i>=1) ΚΑΙ βρ = Ψευδής επανάλαβε
  Αν A[i] = x τότε
    βρ <-- Αληθής
    θέση <-- i
  Αλλιώς
    Αν s = 0 τότε
      i <-- i + 1
    Αλλιώς
      i <-- i - 1
  ΤέλοςΑν
ΤέλοςΑν
ΤέλοςΕπανάληψης
Αναζήτηση <-- θέση
ΤέλοςΣυνάρτησης
```

Παραδείγματα υποπρογραμμάτων

Σε πραγματικό πίνακα A[100] ταξινόμηση κατά αύξουσα(x=0)/φθίνουσα(x=1) σειρά
Διαδικασία Ταξινόμηση(A, x)

```
...
για i από 2 μέχρι 100
  για j από 100 μέχρι i μεβήμα -1
    Αν (x=0 ΚΑΙ A[j-i]>A[j]) Η (x=1 ΚΑΙ A[j-i]<A[j]) τότε
      tmp <-- A[j-1]
      A[j-1] <-- A[j]
      A[j] <-- tmp
    ΤέλοςΑν
  ΤέλοςΕπανάληψης
ΤέλοςΔιαδικασίας
```

Σε πραγματικό πίνακα A[50, 100] επιστροφή μέσου όρου γραμμής k(x=0)/στήλης k(x=1)
Συνάρτηση MO(A, k, x): Πραγματική

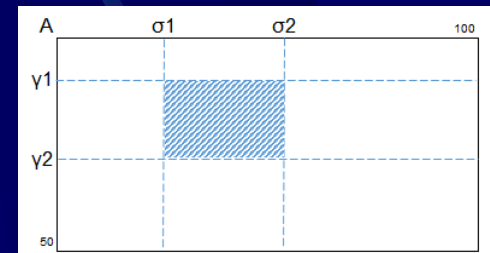
```
...
s <-- 0
Αν x = 0 τότε
  για j από 1 μέχρι 100
    s <-- s + A[k, j]
  ΤέλοςΕπανάληψης
  MO <-- s / 100
Αλλιώς
  για i από 1 μέχρι 50
    s <-- s + A[i, k]
  ΤέλοςΕπανάληψης
  MO <-- s / 50
ΤέλοςΑν
ΤέλοςΣυνάρτησης
```

Σε πραγματικό πίνακα A[50, 100] εμφάνιση κατά γραμμές(x=0)/στήλες(x=1)
Διαδικασία Εμφάνιση(A, x)

```
...
Αν x = 0 τότε
  για i από 1 μέχρι 50
    για j από 1 μέχρι 100
      Γράψε A[i, j]
    ΤέλοςΕπανάληψης
  ΤέλοςΕπανάληψης
Αλλιώς
  για j από 1 μέχρι 100
    για i από 1 μέχρι 50
      Γράψε A[i, j]
    ΤέλοςΕπανάληψης
  ΤέλοςΕπανάληψης
ΤέλοςΑν
ΤέλοςΔιαδικασίας
```

Σε πραγματικό πίνακα A[50, 100] επιστροφή μέσου όρου της υποπεριοχής γραμμών γ1-γ2 και στηλών σ1-σ2
Συνάρτηση MO(A, γ1, γ2, σ1, σ2): Πραγματική

```
...
s <-- 0
για i από γ1 μέχρι γ2
  για j από σ1 μέχρι σ2
    s <-- s + A[i, j]
  ΤέλοςΕπανάληψης
ΤέλοςΕπανάληψης
MO <-- s / ((γ2-γ1+1)*(σ2-σ1+1))
ΤέλοςΣυνάρτησης
```



Παραδείγματα υποπρογραμμάτων

Σε πραγματικό πίνακα $A[100, 100]$ επιστροφή του αθροίσματος των στοιχείων άνω($x=1$)/κάτω($x=2$) της κύριας διαγωνίου

Συνάρτηση $\text{SumD}(A, x)$: Πραγματική

```
...
s <-- 0
για i από 1 μέχρι 100
  για j από 1 μέχρι 100
    Αν (x=1 ΚΑΙ i<j) Η (x=2 ΚΑΙ i>j) τότε
      s <-- s + A[i, j]
    ΤέλοςΑν
  ΤέλοςΕπανάληψης
ΤέλοςΣυνάρτησης
```

Σε πραγματικό πίνακα $A[100, 100]$ επιστροφή του αθροίσματος της 1ης($d=1$)/2ης($d=2$) κύριας διαγωνίου

Συνάρτηση $\text{SumD}(A, d)$: Πραγματική

```
...
s <-- 0
για i από 1 μέχρι 100
  Αν d=1 τότε
    s <-- s + A[i, i]
  Αλλιώς
    s <-- s + A[i, 101-i]
  ΤέλοςΑν
ΤέλοςΕπανάληψης
SumD <-- s
ΤέλοςΣυνάρτησης
ή
Συνάρτηση SumD(A, d): Πραγματική
...
Αν d=1 τότε
  p <-- 1
  y <-- 0
Αλλιώς
  p <-- -1
  y <-- 101
ΤέλοςΑν
s <-- 0
για i από 1 μέχρι 100
  s <-- s + A[i, y + p*i]
ΤέλοςΕπανάληψης
SumD <-- s
ΤέλοςΣυνάρτησης
```

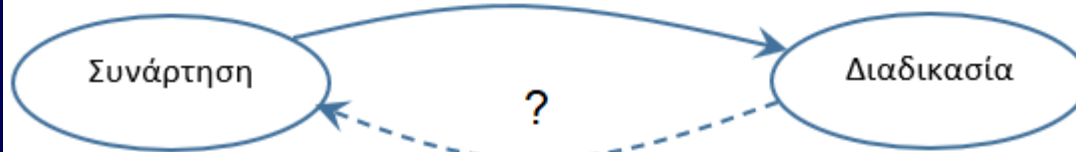


Συνάρτηση - Διαδικασία

	Επιστρεφόμενες τιμές			Διάβασε Γράψε	Αλλαγή πραγματικών παραμέτρων	Κλήση		Σταθερές ή εκφράσεις στις πραγματικές παραμέτρους	Προαιρετική λίστα παραμέτρων	Κλήση με το όνομά της	Τύπος στην επικεφαλίδα	Πολλαπλές κλήσεις σε 1 εντολή	Κατάλληλη σε κάθε περίπτωση	
	0	1	≥ 2			Συνάρτησης	Διαδικασίας							
Συνάρτηση	✗	✓	✗	✗	✗	✓	✗	✓	✗	✓	✓	✓	✗	Συνάρτηση
Διαδικασία	✓	✓	✓	✓	✓	✓	✓	✗	✓	✗	✗	✗	✓	Διαδικασία

Μετατροπές

+ 1 παράμετρος, αντίγραφα τυπικών παραμέτρων(,)



- 1 παράμετρος

Γράψε Συν(x)



Κάλεσε Διαδ(x, y)

Γράψε y