ΔΙΑΔΙΚΑΣΙΕΣ ΚΑΙ ΣΥΝΑΡΤΗΣΕΙΣ

Υπάρχουν δύο ειδών υποπρογράμματα, **οι διαδικασίες** και **οι συναρτήσεις**.

Το είδος κάθε υποπρογράμματος καθορίζεται από το είδος της λειτουργίας που καλείται να επιτελέσει.

Η διαδικασία είναι ένας τύπος υποπρογράμματος που μπορεί να εκτελεί όλες τις λειτουργίες ενός προγράμματος.

Εκτελείται όπως οι εντολές ενέργειας, μπορούν να εισάγουν δεδομένα, να εκτελέσουν υπολογισμούς, να μεταβάλλουν τις τιμές των μεταβλητών και να τυπώσουν αποτελέσματα, να καλέσουν άλλη διαδικασία.

Η συνάρτηση είναι ένας τύπος υποπρογράμματος που υπολογίζει και επιστρέφει μόνο μία τιμή με το όνομά της (όπως οι μαθηματικές συναρτήσεις).

Οι διαφορές από τις διαδικασίες είναι:

-Μπορούν να συμμετέχουν στο σχηματισμό εκφράσεων.

-Δε μπορούν να εκτελεστούν σαν εντολές ενέργειας (δεν μπορούν να διαβάσουν, να εμφανίσουν, να καλέσουν διαδικασία).

-Υπολογίζουν και επιστρέφουν μόνο μια τιμή (αριθμητική - ακέραια ή πραγματική, χαρακτήρα ή λογική).

ΣΥΝΑΡΤΗΣΗ

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

Το όνομα της συνάρτησης μπορεί να είναι οποιαδήποτε αποδεκτή λέξη της ΓΛΩΣΣΑΣ.
Η λίστα παραμέτρων είναι μια λίστα από μεταβλητές των οποίων οι τιμές περνάνε στη συνάρτηση κατά την κλήση της.
Στις εντολές στο εσωτερικό της συνάρτησης θα πρέπει οπωσδήποτε να υπάρχει μια εντολή εκχώρησης τιμής στο όνομα της συνάρτησης.

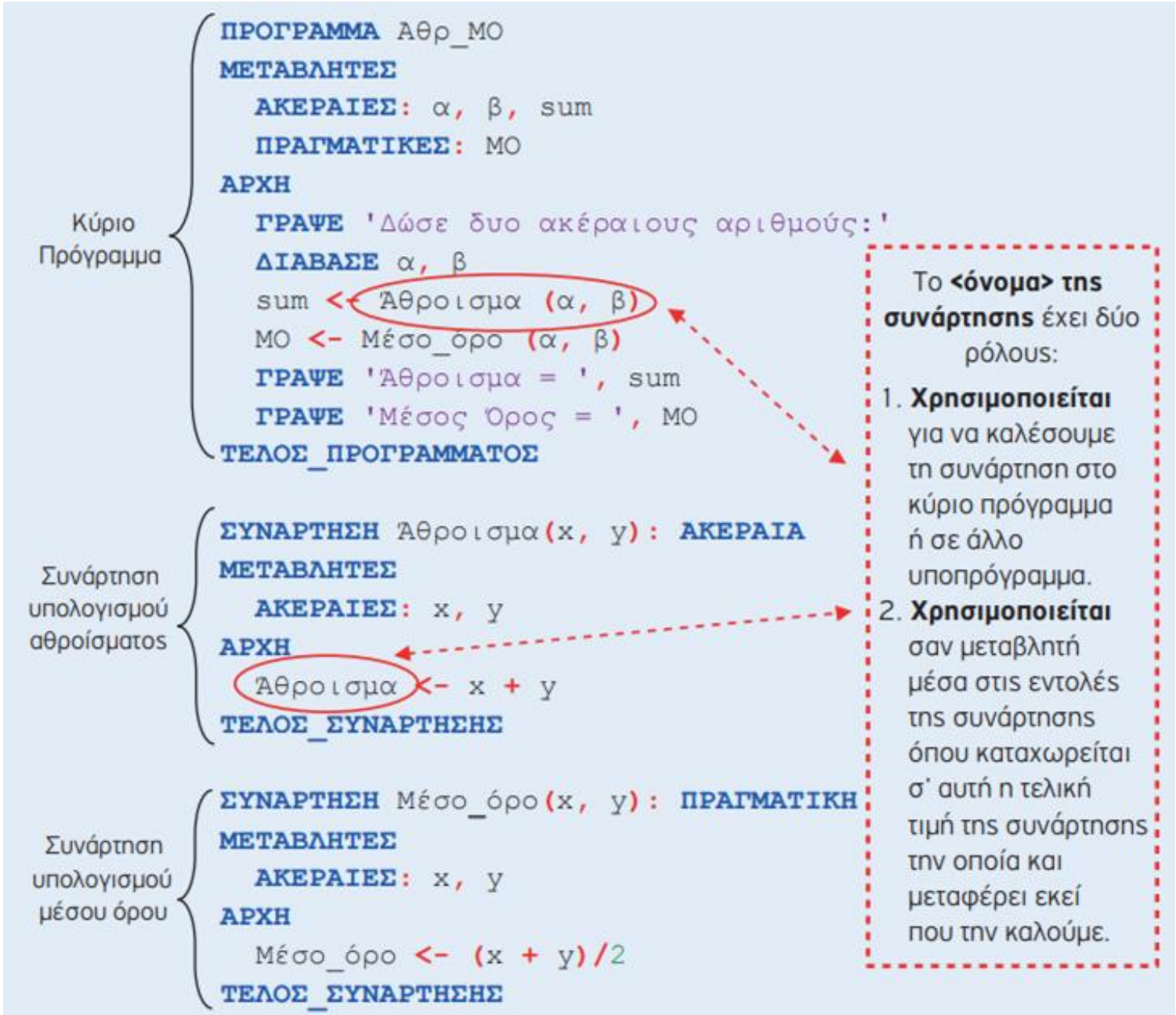
```
ΠΡΟΓΡΑΜΜΑ Π2
ΜΕΤΑΒΛΗΤΕΣ
ΠΡΑΓΜΑΤΙΚΕΣ : R, E
ΑΡΧΗ
   …
   E ← Εμβαδό_κύκλου(R)
   …
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ Π2
```

```
ΣΥΝΑΡΤΗΣΗ Εμβαδό_κύκλου(ακτ) : ΠΡΑΓΜΑΤΙΚΗ
ΣΤΑΘΕΡΕΣ
Π=3.14
ΜΕΤΑΒΛΗΤΕΣ
ΠΡΑΓΜΑΤΙΚΕΣ: ακτ
ΑΡΧΗ
   Εμβαδό_κύκλου ← Π*ακτ^2
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```

Στο παραπάνω παράδειγμα:
1)Εκτελούνται οι εντολές στο κυρίως πρόγραμμα μέχρι που φτάνουμε στο σημείο όπου καλείται η συνάρτηση και η τιμή που θα επιστραφεί θα εκχωρηθεί στην μεταβλητή Ε.
2)Στη συνάρτηση περνάει και αντιστοιχίζεται η παράμετρος R του κυρίως προγράμματος στην παράμετρο ακτ του υποπρογράμματος.
3)Πηγαίνουμε στη συνάρτηση και εκτελούνται οι εντολές, δηλαδή εκχωρείται στο όνομα της συνάρτησης η τιμή Π*ακτ^2.
4)Το αποτέλεσμα της παραπάνω έκφρασης περνάει στο κυρίως πρόγραμμα και καταχωρείται στη μεταβλητή Ε.
5)Εκτελείται κανονικά το υπόλοιπο πρόγραμμα συνεχίζοντας με την επόμενη εντολή.

Όλες οι μεταβλητές είναι γνωστές, έχουν ισχύ όπως λέγεται, μόνο για το τμήμα προγράμματος στο οποίο έχουν δηλωθεί, ισχύουν δηλαδή τοπικά για το συγκεκριμένο υποπρόγραμμα ή κυρίως πρόγραμμα.

Να γραφεί πρόγραμμα που να διαβάζει δύο ακέραιους αριθμούς και να τυπώνει το άθροισμα τους και τον μέσο όρο τους. Ο υπολογισμός του αθροίσματος και του μέσου όρου γίνεται από δύο συναρτήσεις.



Πίνακας τιμών σε συνάρτηση

```
ΠΡΟΓΡΑΜΜΑ Αθρ_ΜΟ
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: α, β, sum
  ΠΡΑΓΜΑΤΙΚΕΣ: ΜΟ
ΑΡΧΗ
  ΓΡΑΨΕ 'Δώσε δυο ακέραιους αριθμούς:'
  ΔΙΑΒΑΣΕ α, β
  sum <- Άθροισμα (α, β)
  ΜΟ <- Μέσο_όρο (α, β)
  ΓΡΑΨΕ 'Άθροισμα = ', sum
  ΓΡΑΨΕ 'Μέσος Όρος = ', ΜΟ
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ

ΣΥΝΑΡΤΗΣΗ Άθροισμα(x, y): ΑΚΕΡΑΙΑ
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: x, y
ΑΡΧΗ
  Άθροισμα <- x + y
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ

ΣΥΝΑΡΤΗΣΗ Μέσο_όρο(x, y): ΠΡΑΓΜΑΤΙΚΗ
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: x, y
ΑΡΧΗ
  Μέσο_όρο <- (x + y)/2
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```

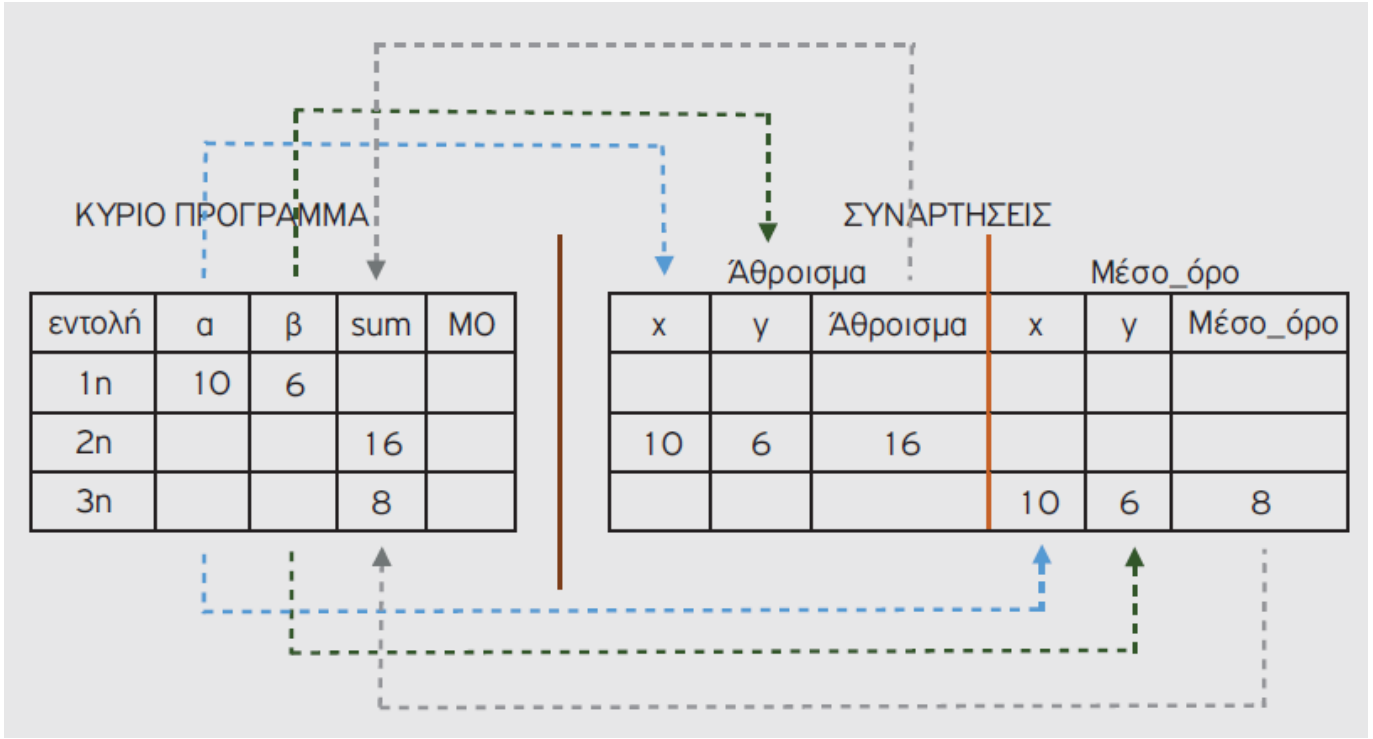
δεδομένα εισόδου: 10, 6

Πίνακας τιμών σε συνάρτηση

```
ΠΡΟΓΡΑΜΜΑ Αθρ_ΜΟ
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: α, β, sum
  ΠΡΑΓΜΑΤΙΚΕΣ: ΜΟ
ΑΡΧΗ
  ΓΡΑΨΕ 'Δώσε δυο ακέραιους αριθμούς:'
  ΔΙΑΒΑΣΕ α, β
  sum <- Άθροισμα (α, β)
  ΜΟ <- Μέσο_όρο (α, β)
  ΓΡΑΨΕ 'Άθροισμα = ', sum
  ΓΡΑΨΕ 'Μέσος Όρος = ', ΜΟ
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ

ΣΥΝΑΡΤΗΣΗ Άθροισμα(x, y): ΑΚΕΡΑΙΑ
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: x, y
ΑΡΧΗ
  Άθροισμα <- x + y
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ

ΣΥΝΑΡΤΗΣΗ Μέσο_όρο(x, y): ΠΡΑΓΜΑΤΙΚΗ
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: x, y
ΑΡΧΗ
  Μέσο_όρο <- (x + y)/2
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```



ΔΙΑΔΙΚΑΣΙΑ

ΔΙΑΔΙΚΑΣΙΑ Όνομα (λίστα παραμέτρων)

Τμήμα δηλώσεων

ΑΡΧΗ

εντολές

ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ

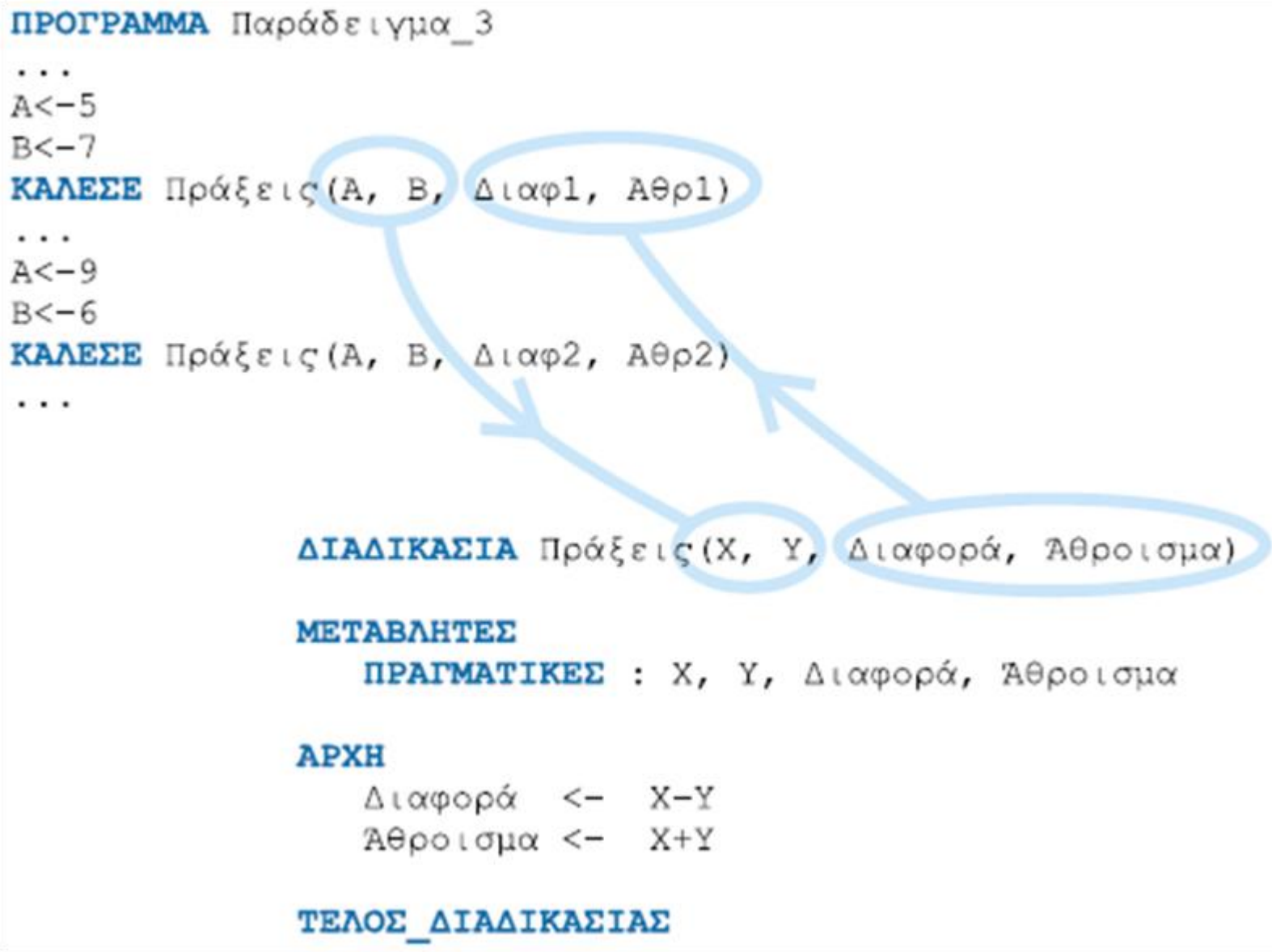
Το όνομα της διαδικασίας μπορεί να είναι οποιαδήποτε τιμή της ΓΛΩΣΣΑΣ.

Η λίστα παραμέτρων είναι μια λίστα από μεταβλητές, των οποίων οι τιμές περνάνε στη διαδικασία κατά την κλήση της. Μπορεί να έχει καμία, μία ή περισσότερους παραμέτρους. Όταν υπάρχουν πολλές παράμετροι, τότε άλλες είναι για να εισάγουν τιμές στη διαδικασία και άλλες να επιστρέψουν τιμές στο κύριο πρόγραμμα.

Για να καλέσουμε τη διαδικασία χρησιμοποιούμε την ειδική εντολή ΚΑΛΕΣΕ:

ΚΑΛΕΣΕ <όνομα_διαδικασίας>(λίστα παραμέτρων)

Μετά το τέλος της διαδικασίας επιστρέφουμε στο κύριο πρόγραμμα στο σημείο ακριβώς μετά την εντολή ΚΑΛΕΣΕ.



```
ΠΡΟΓΡΑΜΜΑ Παράδειγμα_3
...
Α<-5
Β<-7
ΚΑΛΕΣΕ Πράξεις(Α, Β, Διαφ1, Αθρ1)
...
Α<-9
Β<-6
ΚΑΛΕΣΕ Πράξεις(Α, Β, Διαφ2, Αθρ2)
...

ΔΙΑΔΙΚΑΣΙΑ Πράξεις(Χ, Υ, Διαφορά, Άθροισμα)

ΜΕΤΑΒΛΗΤΕΣ
   ΠΡΑΓΜΑΤΙΚΕΣ : Χ, Υ, Διαφορά, Άθροισμα

ΑΡΧΗ
   Διαφορά   <-  Χ-Υ
   Άθροισμα  <-  Χ+Υ

ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

1)Εκτελείται το πρόγραμμα κατά το οποίο εκχωρούνται τιμές στο Α και στο Β.

2)Έπειτα γίνεται η κλήση της διαδικασίας με όνομα Πράξεις και παραμέτρους τα Α, Β, Διαφ1 και Αθρ1.

3)Κατά την κλήση περνάει η τιμή της Α στην αντίστοιχη Χ της διαδικασίας, και η τιμή της Β στην αντίστοιχη Υ της διαδικασίας. Οι παράμετροι Διαφ1 και Αθρ1 δίνονται «κενές» με σκοπό να πάρουν τιμές μετά την εκτέλεση της διαδικασίας.

4)Έχει διακοπεί η εκτέλεση στο κυρίως πρόγραμμα και εκτελούμε τις εντολές στο υποπρόγραμμα. Άρα παίρνουν τιμές οι παράμετροι Διαφορά και Άθροισμα.

5)Όταν φτάνουμε στο ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ επιστρέφονται οι τιμές στο κύριο πρόγραμμα (αντιστοιχίζονται).

6)Συνεχίζεται η εκτέλεση στο κυρίως πρόγραμμα εκτελώντας την εντολή που βρίσκεται αμέσως μετά την εντολή ΚΑΛΕΣΕ. Πλέον οι μεταβλητές Διαφ1 και Αθρ1 έχουν τιμή.

7)Στη συνέχεια γίνεται το ίδιο αλλά στο υποπρόγραμμα δίνονται διαφορετικές τιμές άρα επιστρέφονται διαφορετικά αποτελέσματα.

Να σχεδιασθεί κύριο πρόγραμμα και διαδικασία, έτσι ώστε όταν εκτελείται το κύριο πρόγραμμα (και με κλήση της αντίστοιχης διαδικασίας), θα ανταλλάσει τις τιμές μεταξύ δύο μεταβλητών, για δύο διαφορετικά ζεύγη τιμών που θα δίνονται από το πληκτρολόγιο.
Συγκεκριμένα: θα εισάγονται σε δύο μεταβλητές δύο τιμές από το πληκτρολόγιο και το πρόγραμμα θα ανταλλάσσει τις τιμές τους. Στη συνέχεια, θα εισάγονται για δύο άλλες μεταβλητές δύο νέες τιμές από το πληκτρολόγιο και το πρόγραμμα θα ανταλλάσσει πάλι τις τιμές τους.

```
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

ΔΙΑΔΙΚΑΣΙΑ Εναλλαγή_τιμών(κ, λ)
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: κ, λ,temp
ΑΡΧΗ
  temp <- κ
  κ  <- λ
  λ  <- temp
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ Εναλλαγή_τιμών
```

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

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

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

δες οδηγίες μελέτης σελ. 100-107

Πίνακας τιμών σε διαδικασία

```
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

ΔΙΑΔΙΚΑΣΙΑ Εναλλαγή_τιμών(κ, λ)
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: κ, λ,temp
ΑΡΧΗ
  temp <- κ
  κ  <- λ
  λ  <- temp
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ Εναλλαγή_τιμών
```

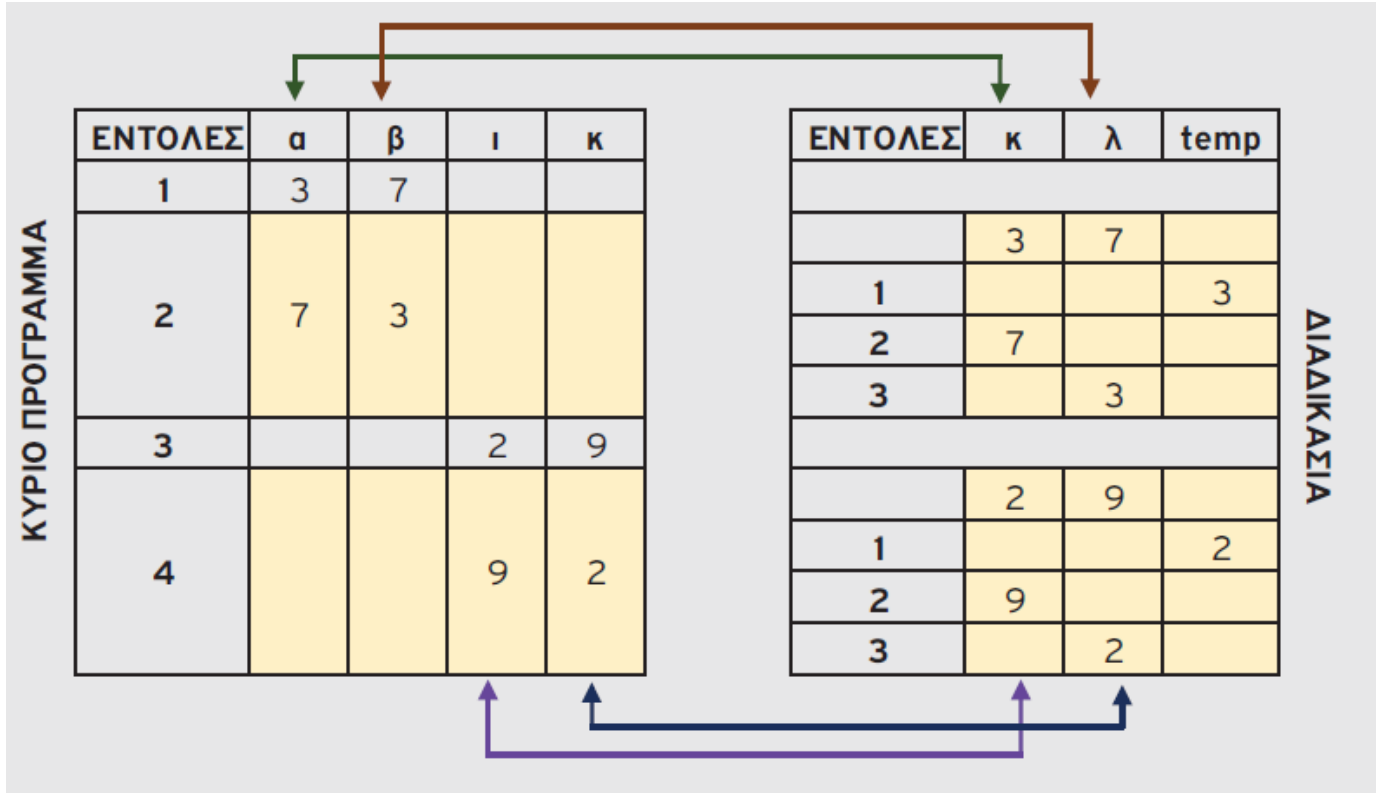
δεδομένα εισόδου: 3, 7, 2, 9

Πίνακας τιμών σε διαδικασία

```
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

ΔΙΑΔΙΚΑΣΙΑ Εναλλαγή_τιμών(κ, λ)
ΜΕΤΑΒΛΗΤΕΣ
  ΑΚΕΡΑΙΕΣ: κ, λ,temp
ΑΡΧΗ
  temp <- κ
  κ  <- λ
  λ  <- temp
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ Εναλλαγή_τιμών
```

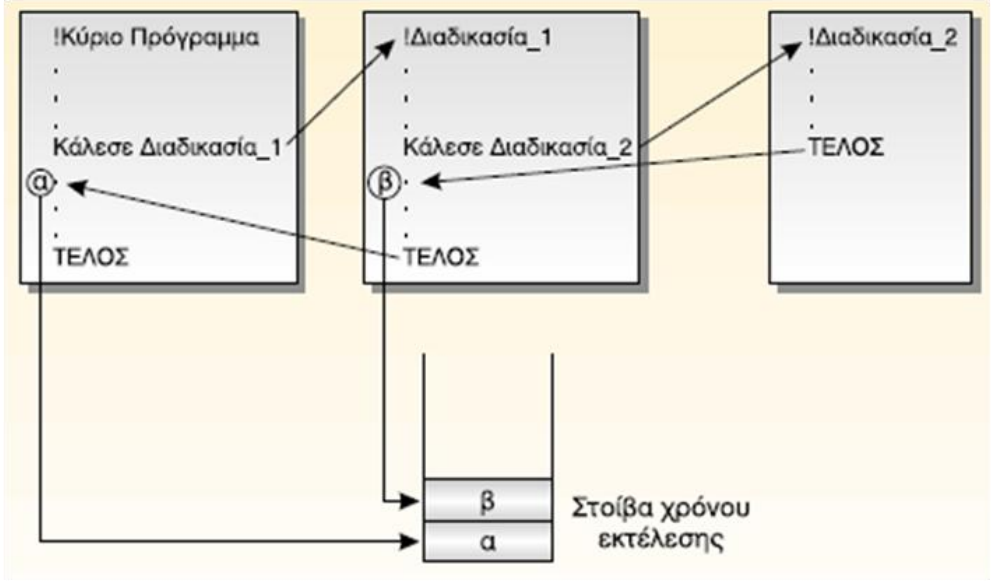


ΕΝΤΟΛΕΣ	α	β	ι	κ
1	3	7		
2	7	3		
3			2	9
4			9	2

ΕΝΤΟΛΕΣ	κ	λ	temp
	3	7	
1			3
2	7		
3		3	
	2	9	
1			2
2	9		
3		2	

Η χρήση στοίβας στη κλήση διαδικασιών

Η έννοια της στοίβας είναι πολύ χρήσιμη στο ίδιο το λογισμικό των γλωσσών προγραμματισμού. 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 (executiontimestack). 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 κοκ. 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.



Εμβέλεια μεταβλητών-σταθερών

Οι μεταβλητές στη ΓΛΩΣΣΑ είναι γνωστές στο αντίστοιχο υποπρόγραμμα που δηλώνονται και μόνο σε αυτό. Όλες οι μεταβλητές (και οι σταθερές) είναι τοπικές στο συγκεκριμένο τμήμα προγράμματος.

Ο μόνος τρόπος για να περάσει μία τιμή από ένα υποπρόγραμμα σε ένα άλλο ή από το κυρίως πρόγραμμα σε ένα υποπρόγραμμα είναι δια μέσου των παραμέτρων κατά το στάδιο της κλήσης του υποπρογράμματος και μετά το τέλος της εκτέλεσης του υποπρογράμματος.

Μία μεταβλητή με το όνομα Γ σε ένα κύριο πρόγραμμα δεν έχει καμία σχέση με τη μεταβλητή Γ ενός υποπρογράμματος, η οποία μπορεί να είναι και διαφορετικού τύπου. Αφού όλες οι μεταβλητές είναι τοπικές, το ίδιο όνομα μεταβλητής μπορεί να εμφανίζεται σε διαφορετικά τμήματα προγράμματος, χωρίς να αντιστοιχεί στην ίδια μεταβλητή.

Πολλές γλώσσες προγραμματισμού επιτρέπουν τη χρήση των μεταβλητών και των σταθερών, όχι μόνο στο τμήμα προγράμματος που δηλώνονται, αλλά και σε άλλα ή ακόμη και σε όλα τα υπόλοιπα υποπρογράμματα. Αυτό που καθορίζει την περιοχή που ισχύουν οι μεταβλητές και οι σταθερές είναι η εμβέλεια των μεταβλητών της γλώσσας

Το τμήμα του προγράμματος που ισχύουν οι μεταβλητές λέγεται εμβέλεια μεταβλητών.

Απεριόριστη εμβέλεια

Σύμφωνα με αυτή την αρχή όλες οι μεταβλητές και όλες οι σταθερές είναι γνωστές και μπορούν να χρησιμοποιούνται σε οποιοδήποτε τμήμα του προγράμματος, άσχετα που δηλώθηκαν. Όλες οι μεταβλητές είναι καθολικές. Η απεριόριστη εμβέλεια καταστρατηγεί την αρχή της αυτονομίας των υποπρογραμμάτων, δημιουργεί πολλά προβλήματα και τελικά είναι αδύνατη για μεγάλα προγράμματα με πολλά υποπρογράμματα, αφού ο καθένας που γράφει κάποιο υποπρόγραμμα πρέπει να γνωρίζει τα ονόματα όλων των μεταβλητών που χρησιμοποιούνται στα υπόλοιπα υποπρογράμματα.

Περιορισμένη εμβέλεια

Η περιορισμένη εμβέλεια υποχρεώνει όλες τις μεταβλητές που χρησιμοποιούνται σε ένα τμήμα προγράμματος, να δηλώνονται σε αυτό το τμήμα. Όλες οι μεταβλητές είναι τοπικές, ισχύουν δηλαδή για το υποπρόγραμμα στο οποίο δηλώθηκαν. Στη ΓΛΩΣΣΑ έχουμε περιορισμένη εμβέλεια. Τα πλεονεκτήματα της περιορισμένης εμβέλειας είναι η απόλυτη αυτονομία όλων των υποπρογραμμάτων και η δυνατότητα να χρησιμοποιείται οποιοδήποτε όνομα, χωρίς να ενδιαφέρει αν το ίδιο χρησιμοποιείται σε άλλο υποπρόγραμμα.

Μερικώς περιορισμένη εμβέλεια

Σύμφωνα με αυτή την αρχή άλλες μεταβλητές είναι τοπικές και άλλες καθολικές. Κάθε γλώσσα προγραμματισμού έχει τους δικούς της κανόνες και μηχανισμούς για τον τρόπο και τις προϋποθέσεις που ορίζονται οι μεταβλητές ως τοπικές ή καθολικές. Η μερικώς περιορισμένη εμβέλεια προσφέρει μερικά πλεονεκτήματα στον πεπειραμένο προγραμματιστή, αλλά για τον αρχάριο περιπλέκει το πρόγραμμα δυσκολεύοντας την ανάπτυξή του.