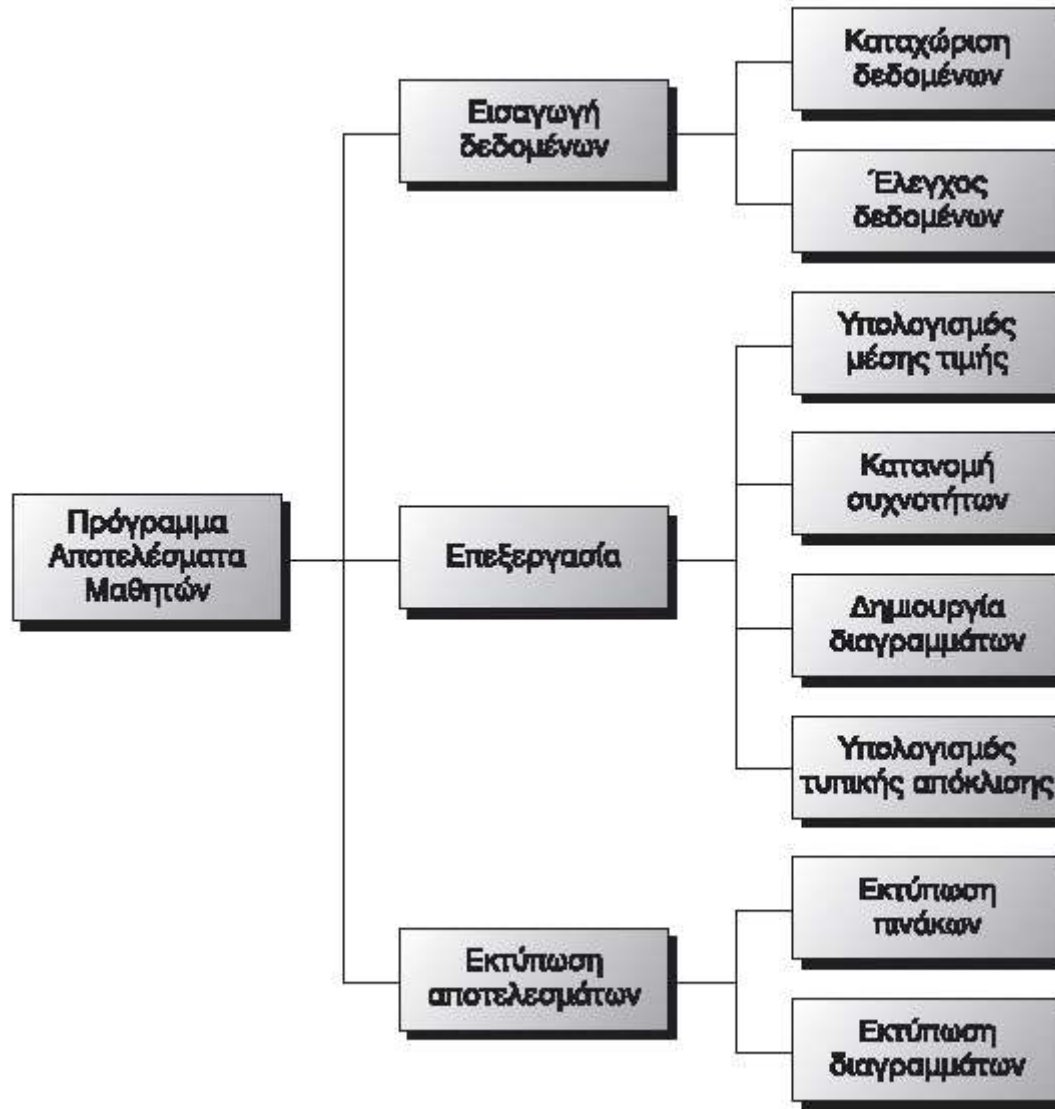


Κεφάλαιο 10 – Υποπρογράμματα

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).



Σχ. 10.1 Διαγραμματική απεικόνιση της ανάλυσης προβλήματος

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.

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

10.5.2 Ορισμός και κλήση διαδικασιών

- 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.

Παράδειγμα

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
```

```
ΣΥΝΑΡΤΗΣΗ Εμβαδό_κύκλου (R) : ΠΡΑΓΜΑΤΙΚΗ
ΣΤΑΘΕΡΕΣ
    Π=3.14
ΜΕΤΑΒΛΗΤΕΣ
    ΠΡΑΓΜΑΤΙΚΕΣ : R
ΑΡΧΗ
    Εμβαδό_κύκλου <- Π*R^2
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```

```
ΔΙΑΔΙΚΑΣΙΑ Είσοδος_δεδομένων (Αριθμός)
ΜΕΤΑΒΛΗΤΕΣ
    ΠΡΑΓΜΑΤΙΚΕΣ : Αριθμός
ΑΡΧΗ
    ΑΡΧΗ_ΕΠΑΝΑΛΗΨΗΣ
        ΓΡΑΨΕ 'Δώσε την ακτίνα'
        ΔΙΑΒΑΣΕ Αριθμός
    ΜΕΧΡΙΣ_ΟΤΟΥ Αριθμός>0
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

```
ΔΙΑΔΙΚΑΣΙΑ Εκτύπωση (Αποτέλεσμα)
ΜΕΤΑΒΛΗΤΕΣ
    ΠΡΑΓΜΑΤΙΚΕΣ : Αποτέλεσμα
ΑΡΧΗ
    ΓΡΑΨΕ 'Το εμβαδό του κύκλου είναι :', Αποτέλεσμα
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

Παράδειγμα

```
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.



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.

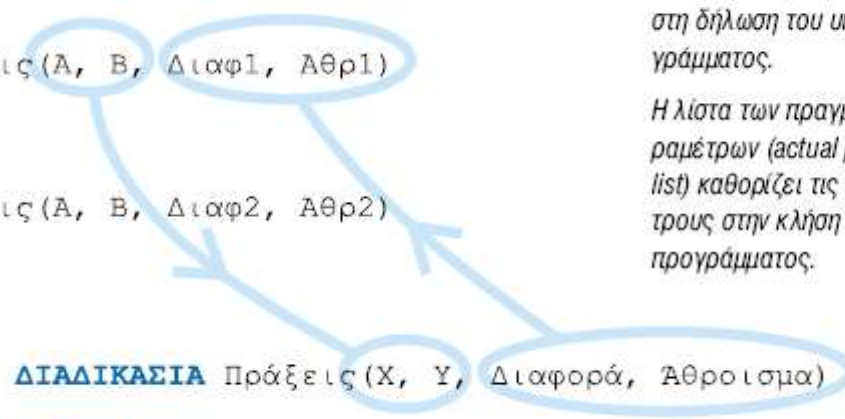
```
ΠΡΟΓΡΑΜΜΑ Παράδειγμα_3
...
A<-5
B<-7
ΚΑΛΕΣΕ Πράξεις(A, B, Διαφ1, Αθρ1)
...
A<-9
B<-6
ΚΑΛΕΣΕ Πράξεις(A, B, Διαφ2, Αθρ2)
...

ΔΙΑΔΙΚΑΣΙΑ Πράξεις(X, Y, Διαφορά, Αθροισμα)

ΜΕΤΑΒΛΗΤΕΣ
  ΠΡΑΓΜΑΤΙΚΕΣ : X, Y, Διαφορά, Αθροισμα

ΑΡΧΗ
  Διαφορά <- X-Y
  Αθροισμα <- X+Y

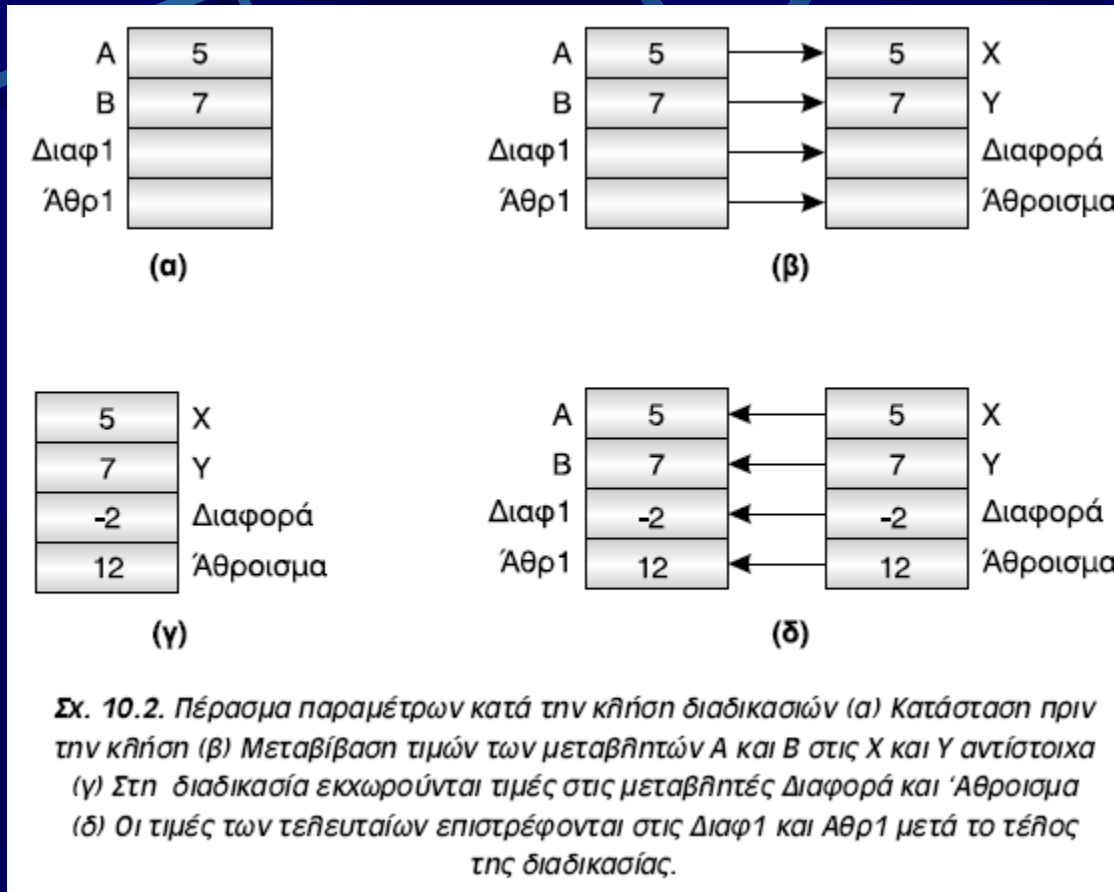
ΤΕΛΟΣ ΔΙΑΔΙΚΑΣΙΑΣ
```



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, Αθροισμα είναι μεταβλητές της διαδικασίας Πράξεις, και ονομάζονται **τυπικές** παράμετροι.

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

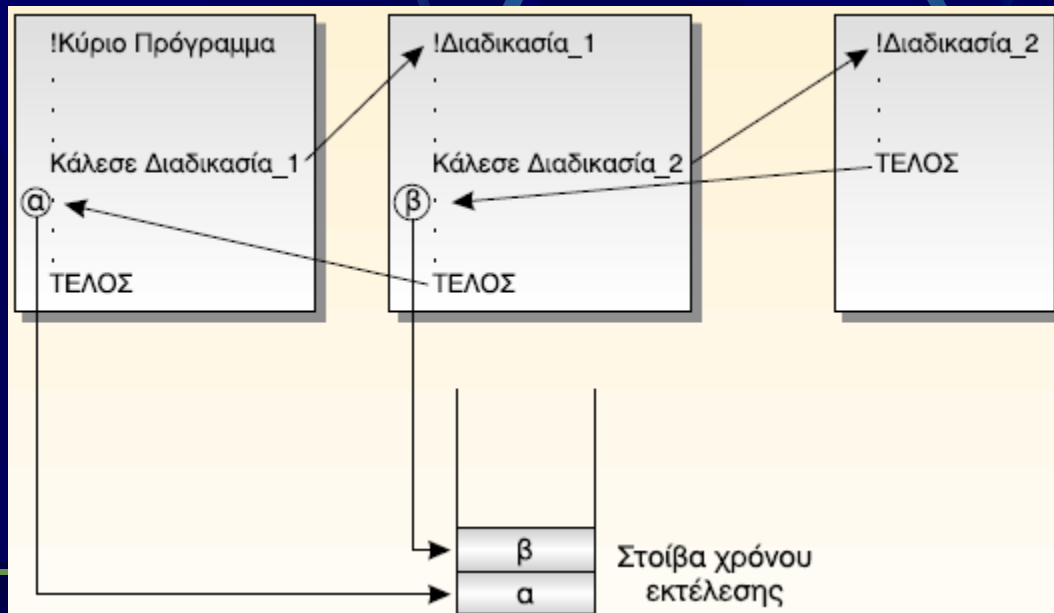
10.5.3 Πραγματικές και τυπικές παράμετροι



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* 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. Το παράδειγμα αυτό δείχνει μία από τις πολλές χρησιμότητες της LIFO ιδιότητας της στοίβας.



10.5.3 Πραγματικές και τυπικές παράμετροι

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 ιδίου τύπου.

Όταν ένα υποπρόγραμμα πρέπει να:

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

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 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

(A)	(B)	(Γ)
α1	α2	α3
27	2	13
27	-25	16

(Γ)	(A)	(B)
α1	α2	α3
16	27	-25
16	11	68

(B)	(Γ)	(A)
α1	α2	α3
68	16	11
68	-52	73

Πίνακας παρακολούθησης τιμών μεταβλητών

ΠΡΟΓΡΑΜΜΑ ΠΤ5

ΜΕΤΑΒΛΗΤΕΣ

ΑΚΕΡΑΙΕΣ: α, β, αποτ

ΑΡΧΗ

$\alpha \leftarrow 2$

$\beta \leftarrow 13$

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

Γράψε αποτ

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

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

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

Γράψε α, β, αποτ

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

Γράψε αποτ

ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ

! =====

Συνάρτηση $\Upsilon\pi(x, y)$: Ακέραιος

ΜΕΤΑΒΛΗΤΕΣ

ΑΚΕΡΑΙΕΣ: x, y, π

ΑΡΧΗ

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

$\Upsilon\pi \leftarrow 2 - \pi$

ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ

α	β	αποτ	Οθόνη
2	13	-	
2	13	-5	-5
5	25	-5	
5	25	-2	5, 25, -2
		-4	-4

(α)	(β)	π	Υπ
x	y	-	
2	13	-	
2	13	7	-5

(β)	(α)	π	Υπ
x	y	-	
25	5	-	
		1	1

(α)	(β)	π	Υπ
x	y	-	
5	25	-	
		6	-4

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 30000 €. 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.

Πρόγραμμα ΚΟΚ

Μεταβλητές

Ακέραιες: ΑΠ, ΣΠ, πρ

Χαρακτήρες: είδος

Αρχή

ΑΠ ← 0

ΣΠ ← 0

Όσο (ΑΠ < 100 ΚΑΙ ΣΠ < 30000) επανάλαβε

 Διάβασε είδος

 πρ ← Πρόστιμο(είδος)

 Γράψε πρ

 ΑΠ ← ΑΠ + 1

 ΣΠ ← ΣΠ + πρ

 ΤέλοςΕπανάληψης

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

Ένας κασκαντέρ χρησιμοποιεί για τις επιδείξεις του 3 τύπους οχημάτων: Αυτοκίνητο(A), Τρακτέρ(T) και Φορηγό(Φ) με αντίστοιχα μήκη 4m, 6m και 10m. Για να περάσει πάνω από παρατεταγμένα οχήματα, πρέπει το όχημά του να αναπτύξει ταχύτητα τουλάχιστον 3πλάσια από το συνολικό μήκος των οχημάτων. Να γραφεί πρόγραμμα το οποίο: α) διαβάζει το πλήθος των παρατεταγμένων οχημάτων, β) για κάθε όχημα διαβάζει με εγκυρότητα τον τύπο του και υπολογίζει με υποπρόγραμμα το μήκος του, γ) υπολογίζει και εμφανίζει την απαραίτητη ταχύτητα που πρέπει να αναπτύξει ο κασκαντέρ και δ) εάν αυτή ξεπερνάει τα 400 km/h να εμφανίζει το μήνυμα «Άλμα αδύνατο».

Πρόγραμμα ΚΟΚ

Μεταβλητές

Ακέραιες:

Χαρακτήρες:

Αρχή

Διάβασε N

$\Sigma \leftarrow 0$

για i από 1 μέχρι N

ΑρχήΕπανάληψης

Διάβασε τύπος

ΜέχριςΌτου (τύπος="A" Η τύπος="T" Η τύπος="Φ")

$\mu \leftarrow \text{Μήκος(τύπος)}$

$\Sigma \leftarrow \Sigma + \mu$

ΤέλοςΕπανάληψης

$u \leftarrow 3 * \Sigma$

Γράψε u, "km/h"

Αν ($u > 400$) τότε Γράψε "Άλμα αδύνατο"

Τέλος_Προγράμματος

Συνάρτηση Μήκος (τύπος): Ακέραια

Μεταβλητές

Ακέραιες: μ

Χαρακτήρες: τύπος

Αρχή

Αν (τύπος = "A") τότε

$\mu \leftarrow 4$

Αλλιώς Αν (τύπος = "T") τότε

$\mu \leftarrow 6$

Αλλιώς

$\mu \leftarrow 10$

ΤέλοςΑν

Μήκος $\leftarrow \mu$

ΤέλοςΣυνάρτησης

Μετατροπή: Συνάρτηση – Διαδικασία – Χωρίς υποπρόγραμμα

Γενικά:

Πρόγραμμα Π1 ... $\alpha \leftarrow \Sigma(x_1, x_2, \dots, x_n)$... Τέλος Προγράμματος	\longrightarrow	Πρόγραμμα Π2 ... Κάλεσε $\Delta(x_1, x_2, \dots, x_n, \alpha)$... Τέλος Προγράμματος
Συνάρτηση $\Sigma(x_1, x_2, \dots, x_n)$: Τύπος ... $\Sigma \leftarrow$ αποτέλεσμα ... Τέλος Συνάρτησης	\longrightarrow	Διαδικασία $\Delta(x_1, x_2, \dots, x_n, x_{n+1})$... $x_{n+1} \leftarrow$ αποτέλεσμα ... Τέλος Διαδικασίας

π.χ.

Πρόγραμμα Με_Συνάρτηση	Πρόγραμμα Με_Διαδικασία	Πρόγραμμα Χωρίς_Υποπρόγραμμα
... Διάβασε A, B $\Gamma \leftarrow 0$ Όσο (B <> 0) επανάλαβε $\Gamma \leftarrow \Gamma + \text{Συν}(A, B)$ $A \leftarrow 2 * A$ $B \leftarrow B \text{ div } 2$ Τέλος Επανάληψης Γράψε Γ	... Διάβασε A, B $\Gamma \leftarrow 0$ Όσο (B <> 0) επανάλαβε Κάλεσε Διαδ(A, B, Ω) $\Gamma \leftarrow \Gamma + \Omega$ $A \leftarrow 2 * A$ $B \leftarrow B \text{ div } 2$ Τέλος Επανάληψης Γράψε Γ	... Διάβασε A, B $\Gamma \leftarrow 0$ Όσο (B <> 0) επανάλαβε Αν (B mod 2 <> 0) τότε $\Omega \leftarrow A$ Αλλιώς $\Omega \leftarrow 0$ Τέλος Αν $\Gamma \leftarrow \Gamma + \Omega$ $A \leftarrow 2 * A$ $B \leftarrow B \text{ div } 2$ Τέλος Επανάληψης Γράψε Γ
Συνάρτηση Συν(χ, ψ): Ακέραια ... Αν (ψ mod 2 <> 0) τότε $\zeta \leftarrow \chi$ Αλλιώς $\zeta \leftarrow 0$ Τέλος Αν Συν $\leftarrow \zeta$	Διαδικασία Διαδ(χ, ψ, ζ) ... Αν (ψ mod 2 <> 0) τότε $\zeta \leftarrow \chi$ Αλλιώς $\zeta \leftarrow 0$ Τέλος Αν	

Τέλος Κεφαλαίου 10 - Υποπρογράμματα