Γ Λυκείου

Ανάπτυξη εφαρμογών σε προγραμματιστικό περιβάλλον

ΥΠΟΠΡΟΓΡΑΜΜΑΤΑ

Περιεχόμενα

Κατηφόρης Παναγιώτης
2019-2020

Τα **υποπρογράμματα** αποτελούν τμήματα (ενότητες) αυτόνομων προγραμμάτων και υλοποιούν το λεγόμενο τμηματικό προγραμματισμό.

10.1 Τμηματικός Προγραμματισμός

Τμηματικός προγραμματισμός ονομάζεται μία τεχνική σχεδίασης και ανάπτυξης προγραμμάτων ως ένα σύνολο από απλούστερα τμήματα προγραμμάτων.

Πρακτικά ο τμηματικός προγραμματισμός σημαίνει το '**σπάσιμο**' ενός προγράμματος (ουσιαστικά ενός αλγορίθμου που μπορεί να είναι μεγάλος και περίπλοκος), σε επιμέρους απλούστερους υποαλγορίθμους ή υποπρογράμματα.

Δηλαδή ένα **υποπρόγραμμα** είναι ένα αυτόνομο τμήμα προγράμματος και συνήθως γράφεται ξεχωριστά από το υπόλοιπο πρόγραμμα. Π.χ. αν έχουμε ένα μεγάλο και περίπλοκο πρόγραμμα, μπορούμε να το αναλύσουμε σε πολλά υποπρογράμματα και να αναθέσουμε κάθε υποπρόγραμμα σε άλλο προγραμματιστή. Έτσι μία ογκώδης εργασία μπορεί να διαμοιραστεί σε πολλούς προγραμματιστές. Το αρχικό πρόγραμμα (κύριο πρόγραμμα) μέσω ειδικής εντολής καλεί κάθε υποπρόγραμμα για να το ενεργοποιήσει. Ασφαλώς είναι δυνατό ένα υποπρόγραμμα να καλέσει άλλο υποπρόγραμμα, κ.ο.κ.

10.2 Χαρακτηριστικά (ιδιότητες) των υποπρογραμμάτων:

1. Κάθε υποπρόγραμμα έχει μόνο μία είσοδο και μία έξοδο. Δηλαδή, ενεργοποιείται με την είσοδο σε αυτό που γίνεται πάντα από την αρχή του, εκτελεί κάποιες ενέργειες και απενεργοποιείται με την έξοδο που γίνεται πάντα από το τέλος του.
2. Κάθε υποπρόγραμμα πρέπει να είναι ανεξάρτητο από τα άλλα. Αυτό σημαίνει ότι μπορεί να σχεδιαστεί, αναπτυχθεί και ελεγχθεί αυτόνομα και ανεξάρτητα από άλλα υποπρογράμματα. Έτσι, διευκολύνεται η ταχεία ανάπτυξη ενός μεγάλου προγράμματος το οποίο χρησιμοποιεί τα υποπρογράμματα.
   Πάντως στην πράξη, η *απόλυτη ανεξαρτησία είναι δύσκολο να επιτευχθεί*.
3. Κάθε υποπρόγραμμα πρέπει να εκτελεί μία μόνο λειτουργία και να μην είναι πολύ μεγάλο. Έτσι, διευκολύνεται η κατανόηση και ο έλεγχός του. Αν εκτελεί περισσότερες λειτουργίες τότε μάλλον πρέπει να διασπαστεί σε ακόμη μικρότερα υποπρογράμματα.

10.3 Πλεονεκτήματα του τμηματικού προγραμματισμού :

1. Διευκολύνει την ανάπτυξη του αλγορίθμου και το αντίστοιχου προγράμματος. Αντί να προσπαθούμε να επιλύσουμε μεμιάς το συνολικό πρόβλημα είναι προτιμότερο να το σπάσουμε σε απλούστερα υποπροβλήματα για τα οποία θα γράψουμε αντίστοιχους αλγόριθμους και συνεπώς τμήματα προγραμμάτων και κατόπιν να προβούμε στη σύνθεσή τους η οποία λύνει και το συνολικό πρόβλημα.
2. Διευκολύνει την κατανόηση και διόρθωση του προγράμματος. Είναι πολύ ευκολότερο από κάποιον να κατανοήσει, να ελέγξει και να τροποποιήσει το μικρότερο υποπρόγραμμα
3. Απαιτεί λιγότερο χρόνο και κόπο στη συγγραφή του προγράμματος. Αν η ίδια λειτουργία χρειάζεται σε διαφορετικά σημεία του συνολικού προγράμματος τότε είναι προτιμότερο να γραφτεί ένα υποπρόγραμμα που την υλοποιεί. Κατόπιν, θα μπορούμε να *καλούμε* το υποπρόγραμμα στο αντίστοιχο σημείο. Έτσι, διευκολύνεται η ταχεία ανάπτυξη του συνολικού προγράμματος μειώνοντας το μέγεθός του και τις πιθανότητες λαθών.
4. Επεκτείνει τις δυνατότητες των γλωσσών προγραμματισμού. Αν μία λειτουργία δεν υποστηρίζεται απευθείας από τη γλώσσα τότε φτιάχνουμε ένα υποπρόγραμμα που την υλοποιεί. Το υποπρόγραμμα τότε μπορεί να είναι διαθέσιμο σε όσα προγράμματα απαιτούν την λειτουργία. Π.χ. ένα υποπρόγραμμα που υπολογίζει το εμβαδόν ενός τριγώνου. Όσα προγράμματα χρειάζονται να υπολογίζουν εμβαδά τριγώνων μπορούν να καλούν αυτό το υποπρόγραμμα. Συγγράφοντας πολλά υποπρογράμματα μπορούμε να δημιουργήσουμε ολόκληρες *βιβλιοθήκες (Libraries)* και ουσιαστικά να επεκτείνουμε την ίδια τη γλώσσα προγραμματισμού για λειτουργίες που δεν τις υποστηρίζει απευθείας.

10.4 Παράμετροι

Τα υποπρογράμματα ενεργοποιούνται από κάποιο άλλο πρόγραμμα ή υποπρόγραμμα για να εκτελέσουν τη λειτουργία τους.

Η επικοινωνία μεταξύ του υποπρογράμματος και αυτού που το καλεί γίνεται διαμέσου κάποιων **ειδικών μεταβλητών** που ονομάζονται **παράμετροι.** Το καλούν πρόγραμμα «περνάει» (μεταβιβάζει) τιμές στο υποπρόγραμμα μέσω των παραμέτρων του. Ενδεχομένως, να επιστρέφει και τιμές στο καλούν μέσω αυτών.

Πρακτικά οι παράμετροι είναι απλές μεταβλητές που χρησιμοποιούνται για την επικοινωνία μεταξύ υποπρογραμμάτων ή μεταξύ του κυρίου προγράμματος και υποπρογραμμάτων. Δηλαδή η παράμετρος περνάει τιμές από ένα τμήμα προγράμματος σε άλλο.

Παράμετρος είναι μία μεταβλητή που επιτρέπει το πέρασμα τιμής από ένα τμήμα προγράμματος σε ένα άλλο.

10.5 Είδη υποπρογραμμάτων

Έχουμε δύο είδη υποπρογραμμάτων τις **διαδικασίες** και τις **συναρτήσεις.**

1. Διαδικασίες.

- **Ορισμός**.
  - — Μία διαδικασία είναι ένας τύπος υποπρογράμματος που μπορεί να εκτελέσει **οποιαδήποτε λειτουργία** (δηλ. μπορεί να διαβάσει, να υπολογίσει, να εμφανίσει).
  - — Μια διαδικασία δέχεται μηδέν, μία ή περισσότερες τιμές και επιστρέφει μηδέν, μία ή περισσότερες τιμές ως αποτελέσματα μέσω των παραμέτρων της.
  - — Η διαδικασία είναι η γενικότερη και ισχυρότερη μορφή υποπρογραμμάτων.
  - — Μπορούμε να τη θεωρήσουμε ως μικρό πρόγραμμα διότι είναι δυνατό να εκτελέσει οποιαδήποτε λειτουργία.
  - — Μια διαδικασία καλείται από το κυρίως πρόγραμμα ή από μια άλλη διαδικασία με την εντολή **ΚΑΛΕΣΕ.**

- **Σύνταξη**:

```
ΔΙΑΔΙΚΑΣΙΑ όνομα (λίστα παραμέτρων)
Τμήμα δηλώσεων μεταβλητών
ΑΡΧΗ
… .
    εντολές
….
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

Εδώ γράφονται και οι **παράμετροι εισόδου** και οι **παράμετροι εξόδου.**

- **1ο παράδειγμα ορισμού διαδικασίας και πώς καλείται:**

```
ΔΙΑΔΙΚΑΣΙΑ Εμβαδόν_τριγώνου (βάση, ύψος, Εμβαδόν)
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ:  βάση, ύψος          ! οι παραμέτρους εισόδου.
    ΠΡΑΓΜΑΤΙΚΕΣ:  Εμβαδόν          ! παράμετρος για την επιστροφή του αποτελέσματος
ΑΡΧΗ    .
    Εμβαδόν ← (βάση * ύψος) / 2
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

```
ΠΡΟΓΡΑΜΜΑ Υπολογισμός_εμβαδών_διαφόρων_τριγώνων
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ:  Β,Υ
    ΠΡΑΓΜΑΤΙΚΕΣ:  Ε
ΑΡΧΗ
    Β ←12
    Υ ← 4
    ΚΑΛΑΕΣΕ Εμβαδόν_τριγώνου (Β,Υ,Ε)
    ΓΡΑΨΕ 'Το εμβαδόν είναι ' , Ε
    Β ← 18
    Υ ← 6
    ΚΑΛΕΣΕ Εμβαδόν_τριγώνου (Β,Υ,Ε)
    ΓΡΑΨΕ 'Το εμβαδόν είναι ' , Ε
Τέλος_προγράμματος
```

Εδώ καλείται ! Περνά τις τιμές των *Β,Υ* του κύριου προγράμματος στις αντίστοιχες παραμέτρους *βάση, ύψος* του υποπρογράμματος. Κατόπιν, η διαδικασία υπολογίζει το εμβαδόν και επιστρέφει την τιμή μέσω της παραμέτρου *Ε*.

Κι εδώ καλείται όπου περνά άλλες τιμές.

- **2ο Παράδειγμα διαδικασίας**

Να γραφτεί μία διαδικασία που εκτυπώνει ένα πλήθος άστρων (δηλαδή τον χαρακτήρα *) στην οθόνη. Το πλήθος των άστρων που θα τυπώνει θα περνά ως παράμετρο. Για παράδειγμα, αν περάσουμε τον αριθμό 5 να τυπώνει *****.

```
ΔΙΑΔΙΚΑΣΙΑ Εκτύπωση_άστρων (Ν)
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ:  Ν      !παράμετρος. Υποδηλώνει το πόσα άστρα θα τυπώσει.
    ΕΚΕΡΑΙΕΣ:  i       !τοπική μεταβλητή
ΑΡΧΗ
    ΓΙΑ i ΑΠΟ 1 ΜΕΧΡΙ Ν
      ΓΡΑΨΕ '*'
    ΤΕΛΟΣ_ΕΠΑΝΑΛΗΨΗΣ
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

Θα γράψουμε, τώρα, ένα πρόγραμμα που θα διαβάζει έναν αριθμό, που αφορά το πόσα άστρα θα τυπώσει, και κατόπιν θα καλεί τη διαδικασία που θα εκτελέσει τη λειτουργία εκτύπωσης.

```
ΠΡΟΓΡΑΜΜΑ Άστρα
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ:  Ν
ΑΡΧΗ
    ΔΙΑΒΑΣΕ Ν
    ΚΑΛΕΣΕ Εκτύπωση_άστρων (Ν)
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟς
```

Εδώ καλείται ! Περνά την τιμή του Ν στη διαδικασία η οποία και τυπώνει το ανάλογο πλήθος άστρων.
Εδώ, δεν υπάρχει επιστρεφόμενο αποτέλεσμα μέσω παραμέτρου.

2. Συναρτήσεις.

- **Ορισμός**

— Μία συνάρτηση είναι ένας τύπος υποπρογράμματος που υπολογίζει μία μόνο τιμή και την επιστρέφει στο πρόγραμμα ή στο υποπρόγραμμα που την καλεί μέσω του ονόματός της.

— Η συνάρτηση είναι η πιο περιορισμένη μορφή υποπρογραμμάτων.

— Μπορούμε να θεωρήσουμε την συνάρτηση ως σύντομο πρόγραμμα που υπολογίζει μόνο μία τιμή που φέρει το όνομά της. Η τιμή αυτή επιστρέφετε στο κυρίως πρόγραμμα ή στο υποπρόγραμμα που την κάλεσε. Τυπικό παράδειγμα είναι οι μαθηματικές συναρτήσεις

— Προσοχή:

> Στις συναρτήσεις δεν χρησιμοποιούμε εντολές εισόδου (ΔΙΑΒΑΣΕ) και εντολές εξόδου (ΓΡΑΨΕ, εμφάνισε ,τύπωσε)

> Μια συνάρτηση δεν μπορεί να καλέσει μια διαδικασία

- **Σύνταξη**:

ΣΥΝΑΡΤΗΣΗ όνομα (λίστα παραμέτρων) : τύπος συνάρτησης
Τμήμα δηλώσεων μεταβλητών
ΑΡΧΗ
….
όνομα ← έκφραση
…
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ

Ο τύπος της συνάρτησης είναι ο τύπος της επιστρεφόμενης τιμής

Εδώ γράφονται μόνο οι **παράμετροι εισόδου.**

- **1ο παράδειγμα ορισμού συνάρτησης και πώς καλείται:**

Να γραφτεί μία συνάρτηση που υπολογίζει το εμβαδόν ενός τριγώνου δοθέντος της βάσης και του ύψους.

ΣΥΝΑΡΤΗΣΗ Εμβαδόν_τριγώνου (βάση, ύψος) : Πραγματική
ΜΕΤΑΒΛΗΤΕΣ
ΑΚΕΡΑΙΕΣ: βάση, ύψος ! οι παράμετροι δηλώνονται εδώ.
ΑΡΧΗ
Εμβαδόν_τριγώνου ← (βάση * ύψος) / 2
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ

Πρόγραμμα Υπολογισμός_εμβαδών_διαφόρων_τριγώνων
ΜΕΤΑΒΛΗΤΕΣ
ΑΚΕΡΑΙΕΣ: Β,Υ
ΠΡΑΓΜΑΤΙΚΕΣ: Ε
ΑΡΧΗ
! Τα δεδομένα για το 1ο τρίγωνο
Β ←12
Υ ← 4
Ε ← Εμβαδόν_τριγώνου (Β,Υ)
ΓΡΑΨΕ'Το εμβαδόν είναι ', Ε
! Τα δεδομένα για το 2ο τρίγωνο
Β ← 18
Υ ← 6
Ε ← Εμβαδόν_τριγώνου (Β,Υ)
ΓΡΑΨΕ 'ο εμβαδόν είναι ', Ε
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟς

Εδώ καλείται ! Περνά τις τιμές των *Β,Υ* του κύριου προγράμματος στις αντίστοιχες παραμέτρους *βάση, ύψος* του υποπρογράμματος. Κατόπιν, η συνάρτηση υπολογίζει το εμβαδόν και επιστρέφει την τιμή μέσω του ονόματός της. Η τιμή αυτή εκχωρείται στη μεταβλητή Ε

Κι εδώ καλείται όπου περνά άλλες τιμές.

- **2ο παράδειγμα συνάρτησης:**

Να γραφτεί μία συνάρτηση που υπολογίζει το μέγιστο τριών ακεραίων.

ΣΥΝΑΡΤΗΣΗ Μέγιστο (Α, Β,Γ) : Ακεραία
ΜΕΤΑΒΛΗΤΕΣ
ΑΚΕΡΑΙΕΣ: Α, Β, Γ ! οι παράμετροι δηλώνονται εδώ.
ΑΚΕΡΑΙΕΣ: max ! βοηθητική μεταβλητή για το μέγιστο.
ΑΡΧΗ
ΑΝ Α > Β ΤΟΤΕ
max ← Α
ΑΛΛΙΩΣ
max ←Β
ΤΕΛΟΣ_ΑΝ
ΑΝ Γ > max ΤΟΤΕ
max ← Γ
ΤΕΛΟΣ_ΑΝ
Μέγιστο ←max
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ

```
ΠΡΟΓΡΑΜΜΑ Μέγιστο_3_Ακεραίων
ΜΕΤΑΒΛΗΤΕΣ
    ΑΚΕΡΑΙΕΣ:   X, Y, Z, max
ΑΡΧΗ
    ΓΡΑΨΕ ′ Δώσε 3 ακεραίους′
    ΔΙΑΒΑΣΕ X, Y, Z
    max ← Μέγιστο(X, Y, Z)
    ΓΡΑΨΕ ′Ο μεγαλύτερος είναι ο ′, max
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

Εδώ καλείται η συνάρτηση όπου περνά τις τιμές X,Y,Z του κυρίου προγράμματος.

Σημειώσεις:

1. Κάθε διαδικασία ή συνάρτηση μπορεί να καλείται από το κύριο πρόγραμμα, από άλλη διαδικασία.
2. Μια συνάρτηση μπορεί να καλέσει **μόνο** συνάρτηση.
3. Οι λίστες παραμέτρων στα υποπρογράμματα δεν είναι υποχρεωτικές, δηλαδή μπορεί να μην υπάρχουν παράμετροι για μεταβίβαση τιμών ή επιστροφή αποτελεσμάτων.
4. Οι παράμετροι
   στη συνάρτηση χρησιμοποιούνται για να περάσουμε τιμές από το πρόγραμμα (ή υποπρόγραμμα) προς την συνάρτηση και όχι το αντίθετο.
   στη διαδικασία χρησιμοποιούνται για να περάσουμε τιμές από το πρόγραμμα (ή υποπρόγραμμα) προς την διαδικασία αλλά και αντίστροφα.
5. Είναι προφανές ότι κάθε συνάρτηση μπορεί να γραφεί ως διαδικασία, ενώ το αντίστροφο δεν ισχύει πάντα.
6. Η συνάρτηση καλείται μέσω του ονόματός της
7. Η διαδικασία καλείται με την **εντολή Κάλεσε**

Πραγματικές και τυπικές παράμετροι

Οι παράμετροι που χρησιμοποιούμε για την επικοινωνία των υποπρογραμμάτων με το κυρίως πρόγραμμα διακρίνονται σε δύο κατηγορίες: στις **πραγματικές** παραμέτρους και στις **τυπικές** παραμέτρους.

Οι πραγματικές παράμετροι είναι αυτές που χρησιμοποιούμε όταν **καλούμε** το υποπρόγραμμα

Οι τυπικές παράμετροι είναι αυτές που χρησιμοποιούμε όταν **ορίζουμε** το υποπρόγραμμα.

Αυτό φαίνεται καλύτερα στο παρακάτω παράδειγμα.

```
ΠΡΟΓΡΑΜΜΑ παράδειγμα
ΜΕΤΑΒΛΗΤΕΣ
        ΠΡΑΓΜΑΤΙΚΕΣ:Α, Β, Διαφ, Αθρ, Γ
        ΠΡΑΓΜΑΤΙΚΕΣ: X, Z.
ΑΡΧΗ
        X ← 5
        Z ← 8
        A ← 2*X – 3
        B ← 3*Z + 1
        ΚΑΛΕΣΕ Πράξη1(Α, Β, Διαφ, Αθρ)
        …..
        Γ ← Πράξη2(Α, Β)
        ….
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

(Απλές) Μεταβλητές

Πραγματικές παράμετροι

Πραγματικές παράμετροι

```
ΔΙΑΔΙΚΑΣΙΑ Πράξη1(Χ, Υ, Δ, Α)
ΜΕΤΑΒΛΗΤΕΣ
        ΠΡΑΓΜΑΤΙΚΕΣ:Χ, Υ, Δ, Α
ΑΡΧΗ
        Δ ← Χ – Υ
        Α ← Χ + Υ
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ

ΣΥΝΑΡΤΗΣΗ Πράξη2(Χ, Υ) : ΠΡΑΓΜΑΤΙΚΗ
ΜΕΤΑΒΛΗΤΕΣ
        ΠΡΑΓΜΑΤΙΚΕΣ:Χ, Υ, Αποτ
ΑΡΧΗ
        Αποτ ← Χ * Υ
        Πράξη2 ← Αποτ
ΤΕΛΟΣ_ΣΥΝΑΡΤΗΣΗΣ
```

Τυπικές παράμετροι της διαδικασίας

Τυπικές παράμετροι της συνάρτησης

Κανόνες παραμέτρων:

1. Τα ονόματα των πραγματικών και τυπικών παραμέτρων μπορεί να είναι οποιαδήποτε.
2. Ο αριθμός των τυπικών και πραγματικών παραμέτρων πρέπει να είναι ο ίδιος.
3. Κάθε πραγματική παράμετρος αντιστοιχεί στην τυπική που βρίσκεται στην ίδια θέση.
4. Η τυπική παράμετρος και η αντίστοιχη πραγματική είναι ίδιου τύπου.
5. Η ισχύς όλων των μεταβλητών είναι τοπική δηλαδή έχουν ισχύ μόνο για το πρόγραμμα ή το υποπρόγραμμα στο οποίο έχουν δηλωθεί..

10.6 Εμβέλεια μεταβλητών και σταθερών

Κάθε κύριο πρόγραμμα και κάθε υποπρόγραμμα περιλαμβάνει τις δικές του μεταβλητές και συμβολικές σταθερές.

Μπορώ να χρησιμοποιήσω μια μεταβλητή, η οποία έχει δηλωθεί, στο κύριο πρόγραμμα, σε ένα υποπρόγραμμα, χωρίς να την δηλώσω στο υποπρόγραμμα;

Μπορώ να χρησιμοποιήσω μια σταθερά, την οποία έχω δηλώσει, σε κάποιο υποπρόγραμμα, στο κύριο πρόγραμμα, χωρίς να την δηλώσω σ΄αυτό;

Το τμήμα του προγράμματος στο οποίο ισχύουν οι μεταβλητές (και οι συμβολικές σταθερές) λέγεται **εμβέλεια** μεταβλητών (συμβολικών σταθερών)

Ειδή εμβέλειας

1. **ΑΠΕΡΙΟΡΙΣΤΗ ΕΜΒΕΛΕΙΑ**
   Σύμφωνα μ' αυτήν την αρχή **όλες** οι μεταβλητές και οι σταθερές είναι γνωστές και μπορούν να χρησιμοποιηθούν σε **οποιοδήποτε** σημείο του προγράμματος, άσχετα που δηλώθηκαν. Είναι δηλαδή **καθολικές.**
   Η απεριόριστη εμβέλεια καταστρατηγεί την αρχή της αυτονομίας (ανεξαρτησίας) των υποπρογραμμάτων.
2. **ΠΕΡΙΟΡΙΣΜΕΝΗ ΕΜΒΕΛΕΙΑ**
   Σύμφωνα μ' αυτήν την αρχή ο προγραμματιστής **υποχρεώνεται** να δηλώνει όλες τις μεταβλητές ή σταθερές που χρησιμοποιεί σε ένα υποπρόγραμμα, στο τμήμα δηλώσεων του υποπρογράμματος αυτού.
   Όλες οι μεταβλητές ή σταθερές είναι **τοπικές**, ισχύουν δηλαδή για το υποπρόγραμμα στο οποίο δηλώθηκαν.
   Στη γλώσσα προγραμματισμού **ΓΛΩΣΣΑ** έχουμε περιορισμένη εμβέλεια.
3. **ΜΕΡΙΚΩΣ ΠΕΡΙΟΡΙΣΜΕΝΗ ΕΜΒΕΛΕΙΑ**
   Σύμφωνα μ' αυτήν την αρχή άλλες μεταβλητές είναι τοπικές και άλλες καθολικές.

ΓΕΝΙΚΑ:
Κάθε γλώσσα προγραμματισμού έχει τους δικούς της κανόνες και μηχανισμούς για τον τρόπο και τις προϋποθέσεις που ορίζονται οι μεταβλητές και οι συμβολικές σταθερές ως τοπικές ή καθολικές

5.2.4 Εκσφαλμάτωση λογικών λαθών στα υποπρογράμματα

Κατά την εκσφαλμάτωση προγραμμάτων που χρησιμοποιούν υποπρογράμματα χρειάζεται να δίνεται προσοχή στον εντοπισμό λογικών λαθών που σχετίζονται με:

- την κλήση του υποπρογράμματος και το πέρασμα των παραμέτρων
- τα λοιπά λογικά λάθη που εμφανίζονται και στα προγράμματα.