Ανάπτυξη Εφαρμογών σε Προγραμματιστικό Περιβάλλον

Επιμέλεια : Δρεμούσης Παντελής

Κεφάλαια 2,7,8

1. **Τι είναι αλγόριθμος;**

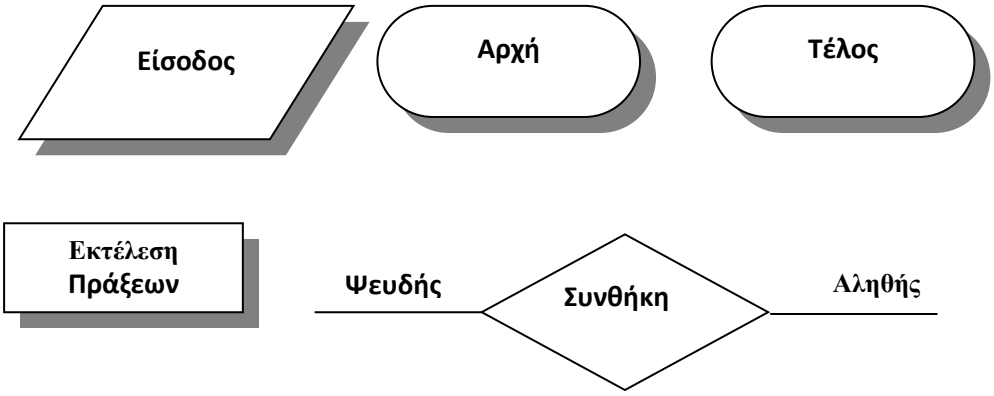
   Μια πεπερασμένη σειρά ενεργειών , αυστηρά καθορισμένων και εκτελέσιμων σε πεπερασμένο χρόνο, που στοχεύουν στην επίλυση ενός προβλήματος.

2. **Ποια κριτήρια πρέπει να ικανοποιεί ο αλγόριθμος;**
   a. Είσοδος
   b. Έξοδος
   c. Περατότητα (πεπερασμένα βήματα)
   d. Αποτελεσματικότητα (κάθε εντολή απλή και εκτελέσιμη)
   e. Καθοριστικότητα (απόλυτα καθορισμένες εντολές)
3. **Τρόποι αναπαράστασης Αλγορίθμων**
   a. Ελεύθερο κείμενο (πιθανή παραβίαση αποτελεσματικότητας)
   b. Φυσική γλώσσα (πιθανή παραβίασηκαθοριστικότητας)
   c. Διαγραμματικές Τεχνικές (όχι πρακτικά για μεγάλους αλγορίθμους)
   d. Κωδικοποίηση (Ψευδογλώσσα η γλώσσα προγραμματισμού)
4. **Τι είναι το διάγραμμα ροής;**

   Ένα σύνολο γεωμετρικών σχημάτων που το καθένα δηλώνει μια ενέργεια και βέλη που δηλώνουν την σειρά εκτέλεσης

<u>Σύμβολα:</u>

- **Έλλειψη**, που δηλώνει την αρχή και το τέλος του κάθε αλγορίθμου.
- **Πλάγιο παραλληλόγραμμο**, που δηλώνει είσοδο ή έξοδο στοιχείων.
- **Ορθογώνιο παραλληλόγραμμο**, που δηλώνει την εκτέλεση μίας ή περισσοτέρων πράξεων και γενικότερα την επεξεργασία.
- **Ρόμβος**, που δηλώνει μία συνθήκη η αλήθεια ή όχι της οποίας μας οδηγεί σε δύο ή περισσότερες εξόδους.



5. **Τι είναι οι σταθερέςκαι σε τι χρησιμεύουν;**

   Είναι προκαθορισμένες αμετάβλητες τιμές που αντιστοιχούνται με ένα όνομα (π.χ. Π=3.14), επίσης σταθερές οναμάζονται και οι αμετάβλητες τιμές που χρησιμοποιούνται μέσα σε μια έκφραση (π.χ. Εμ← 3.14 * R ^ 2)

   Αλλάζω την τιμή μιας σταθεράς πολύ πιο εύκολα στην αρχή ενός προγράμματος και όχι κατά την διάρκεια του προγράμματος

6. **Τι είναι μεταβλητές;**

   Ένα δεδομένο – μια ποσότητα που συμβολίζεται με ένα όνομα και περιέχει μια τιμή. Η τιμή του μπορεί να αλλάξει.

7. **Τι είναι δεσμευμένες λέξεις;**

   Είναι οι εντολές και οι λέξεις που χρησιμοποιεί μια γλώσσα προγραμματισμού (π.χ. ΔΙΑΒΑΣΕ, ΓΡΑΨΕ , ΑΝ ,ΓΙΑ,ΟΣΟ,ΚΑΙ, Η,….)

8. **Ποιοι είναι οι τύποι των δεδομένων – μεταβλητών;**
   a. ΑΚΕΡΑΙΕΣ
   b. ΠΡΑΓΜΑΤΙΚΕΣ
   c. ΧΑΡΑΚΤΗΡΕΣ
   d. ΛΟΓΙΚΕΣ

9. **Ποιους κανόνες πρέπει να πληροί ένας όνομα;**

   Όσον αφορά την ονομασία του αλγορίθμου ισχύουν κάποιοι κανόνες που πρέπει να τηρούνται αυστηρά. Συγκεκριμένα το όνομα του αλγορίθμου πρέπει:

   - Να μην περιέχει κενά.
   - Να μην ξεκινάει με αριθμό.(μπορεί να έχει αριθμό μέσα στην ονομασία)
   - Να περιέχει αλφαριθμητικούς χαρακτήρες και όχι διάφορα ειδικά σύμβολα όπως <>,.+- $ #
   - Να μην είναι δεσμευμένη λέξη

Εξαίρεση στον κανόνα χρήσης συμβόλων, αποτελεί το σύμβολο _ (η κάτω παύλα) το οποίο μπορεί να χρησιμοποιηθεί μέσα στην ονομασία των μεταβλητών προκειμένου να διαχωρίζονται οι λέξεις μεταξύ τους.

Ειδικότερα το σύμβολο ! έχει έναν διαφορετικό ρόλο. Αν μια γραμμή εντολών ξεκινάει με το σύμβολο ! τότε η γραμμή αυτή αγνοείται και δεν εκτελείται. Έτσι μπορούμε να χρησιμοποιήσουμε το σύμβολο ! για εισαγωγή σχολίων στον κώδικά μας προκειμένου ο αναγνώστης να κατανοήσει καλύτερα τον κώδικά μας

10. **Τι είναι οι Τελεστές; Και ποιες κατηγορίες;**
    Είναι τα σύμβολα των πράξεων
    a. Αριθμητικοί : +,-,*,/,^,DIV,MOD
    b. Λογικοί : ΚΑΙ , Η , ΟΧΙ
    c. Συγκριτικοί : <, >, <=, >= ,= , <>

11. **Τι είναι οι Τελεσταίοι;**
    Οι σταθερές και οι μεταβλητές που μαχζί με τους τελεστές δημιουργούν μια έκφραση
12. **Προτεραιότητα πράξεων:**
    Για τους αριθμητικούς : 1. ^ , 2. * , / , DIV , MOD , 3. + , -
    Αν έχουν ίδια προτεραιότητα γίνονται από αριστερά προς τα δεξιά. Οι πράξεις μέσα στις παρενθέσεις προηγούνται.
    Για λογικούς : 1. ΟΧΙ , 2. ΚΑΙ , 3.Η
13. **Ποιες είναι οι στοιχειώδεις λογικές δομές εντολών;**
    a. Δομή ακολουθίας (ΔΙΑΒΑΣΕ, ΓΡΑΨΕ…)
    b. ΔΟΜΗ ΕΠΙΛΟΓΗΣ (ΑΝ, ΑΛΛΙΩΣ_ΑΝ…)
    c. ΔΟΜΗ ΕΠΑΝΑΛΗΨΗΣ (ΓΙΑ ,ΟΣΟ ,ΜΕΧΡΙ…)
14. **Τι είναι εμφωλευμένες δομές;**
    Ο συνδυασμός δυο η περισσοτέρων δομών επιλογής η και επανάληψης , όπου η μια περιέχεται μέσα στην άλλη.
15. **Τι είναι οι λογικές πράξεις;**
    Οι πράξεις που γίνονται σε μια σύνθετη λογική έκφραση με τους λογικούς τελεστές Η (διάζευξη) ,ΚΑΙ (σύζευξη), ΟΧΙ (άρνηση)

16. Λογικές Πράξεις

p	q	p ΚΑΙ q	p Η q	ΟΧΙ p
Αληθής	Αληθής	Αληθής	Αληθής	Ψευδής
Αληθής	Ψευδής	Ψευδής	Αληθής	Ψευδής
Ψευδής	Αληθής	Ψευδής	Αληθής	Αληθής
Ψευδής	Ψευδής	Ψευδής	Ψευδής	Αληθής

17. Τι είναι βρόχος;

Είναι η δομή επανάληψης – οι εντολές που επαναλαμβάνονται

18. Ποιες είναι οι δομές επανάληψης και ποιες οι διαφορές τους;

a. ΟΣΟ : σύνθετη συνθήκη (Αγνωστο / Γνωστό πλήθος επαναλήψεων)
b. Μεχρις : σύνθετη συνθήκη (Αγνωστο / Γνωστό πλήθος επαναλήψεων)
c. ΓΙΑ : απλή συνθήκη (Γνωστό πλήθος επαναλήψεων)

19. Μετατροπές

Από ΓΙΑ μπορώ να μετατρέψω στις άλλες 2 δομές ΟΣΟ και ΜΕΧΡΙΣ

Από ΟΣΟ και ΜΕΧΡΙΣ μπορώ να μετατρέψω σε ΓΙΑ μόνο για γνωστό πλήθος επαναλήψεων

Προσοχή στις περιπτώσεις με άγνωστο πλήθος από ΟΣΟ σε ΜΕΧΡΙΣ να γίνεται χρήση ΑΝ για επιπλέον έλεγχο και επίσης από ΜΕΧΡΙΣ σε ΟΣΟ και επειδή οι εντολές στην ΜΕΧΡΙΣ εκτελούνται τουλάχιστον μια φορά πριν ελεγχθούν οι συνθήκες θα πρέπει να εκτελούνται οι εντολές μια φορά εκτός της ΟΣΟ

Κεφάλαια 3,9

20. Τι είναιΔομή Δεδομένων (DataStructure);

Είναι ένα σύνολο αποθηκευμένων δεδομένων που μπορούν και υφίστανται επεξεργασία από ένα σύνολο λειτουργιών. Κάθε δομή δεδομένων αποτελείται από κόμβους.

21. Πράξεις επί των δομών δεδομένων

a. Προσπέλαση (access): Πρόσβαση σε κόμβο με σκοπό να εξεταστεί ή να τροποποιηθεί το περιεχόμενό του.
b. Εισαγωγή (insertion): Προσθήκη νέων κόμβων σε υπάρχουσα δομή.
c. Διαγραφή (deletion): Αφαιρείται ένας κόμβος από τη δομή. Αντίστροφο της εισαγωγής.
d. Αναζήτηση (searching): Προσπέλαση κόμβων με σκοπό να εντοπιστούν ένας ή περισσότεροι που έχουν μια δεδομένη ιδιότητα.

e. Ταξινόμηση (sorting): οι κόμβοι διατάσσονται σε αύξουσα ή φθίνουσα σειρά.
f. Αντιγραφή (copying): όλοι οι κόμβοι ή μερικοί από αυτούς αντιγράφονται σε μια άλλη δομή.
g. Συγχώνευση (merging): Δύο ή περισσότερες δομές συνενώνονται σε μία ενιαία δομή.
h. Διαχωρισμός (separation): Αντίστροφη της συγχώνευσης.

22. Κατηγορίες Δομών Δεδομένων

Στατικές:

- Γνωρίζουμε τον χώρο μνήμης που καταλαμβάνουν.
- Το ακριβές μέγεθος της κύριας μνήμης καθορίζεται κατά τη στιγμή του προγραμματισμού του και κατά συνέπεια κατά τη στιγμή της μετάφρασής τους και όχι κατά τη στιγμή εκτέλεσης του προγράμματος.
- Αποθηκεύονται σε συνεχόμενες θέσεις μνήμης. Πχ Πίνακες

Δυναμικές:

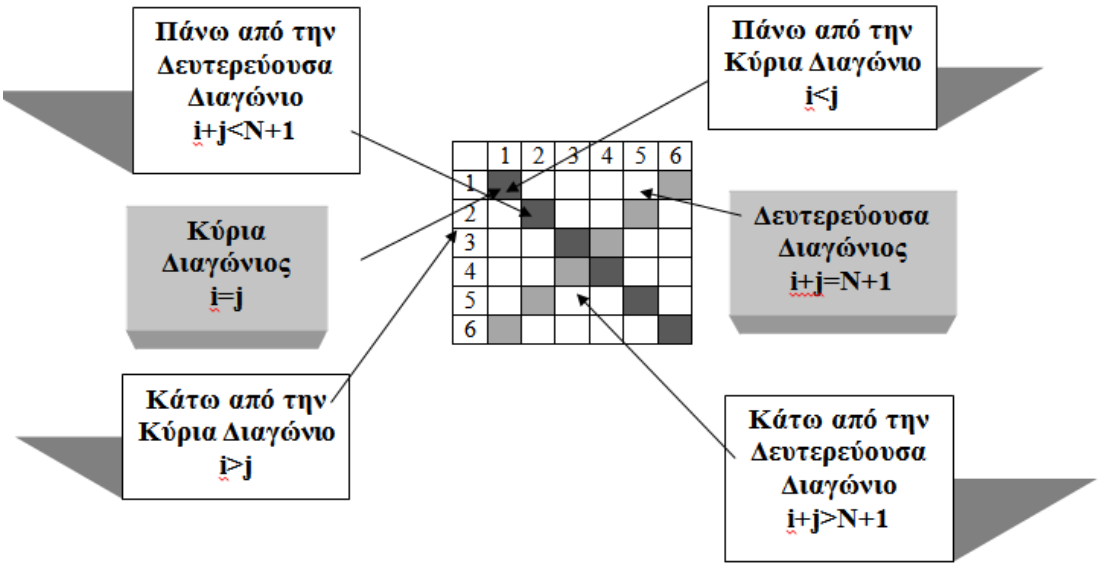
- Δεν είναι γνωστός ο χώρος μνήμης που καταλαμβάνουν..
- Στηρίζονται στην τεχνική της λεγόμενης δυναμικής παραχώρησης μνήμης (dynamicmemoryallocation). Τους παραχωρείται μνήμη ανάλογα με τις ανάγκες τους
- ΔΕΝ αποθηκεύονται σε συνεχόμενες θέσεις μνήμης.
- Ο αριθμός των κόμβων τους μεγαλώνει ή μικραίνει καθώς στη δομή εισάγονται νέα δεδομένα ή διαγράφονται κάποια άλλα αντίστοιχα. Πχ Ουρές, Λίστες.

23. Τι είναι Πίνακας και ποια τα χαρακτηριστικά του;

Πίνακας είναι ένα σύνολο αντικειμένων ίδιου τύπου, τα οποία αναφέρονται με ένα κοινό όνομα. Κάθε ένα από τα αντικείμενα που απαρτίζουν τον πίνακα λέγεται στοιχείο του πίνακα. Η αναφορά σε ατομικά στοιχεία του πίνακα γίνεται με το όνομα του πίνακα ακολουθούμενο από ένα δείκτη.

- Καταλαμβάνει διαδοχικές θέσεις στη μνήμη.
- Το πλήθος τους είναι προκαθορισμένο και δεσμεύονται κατά την μεταγλώττιση του προγράμματος και όχι κατά την εκτέλεσή του.
- Κάθε θέση μνήμης είναι και ένα στοιχείο του πίνακα και προσδιορίζεται από την τιμή ενός δείκτη.
- Πρακτικά ο πίνακας είναι μία στατική δομή δεδομένων που μας επιτρέπει να καταχωρίσουμε κάτω από το όνομα μιας μόνο μεταβλητής πολλά στοιχεία ίδιου τύπου.
- Τα δεδομένα ενός πίνακα είναι πάντα του ιδίου τύπου.

24. Ιδιότητες Τετραγωνικού Πίνακα



25. Είδη Ταξινόμησης Πίνακα;

a. Μέθοδος Φυσαλίδας (Bubble sort)
b. Μέθοδος επιλογής (Selection Sort)

26. Είδη Αναζήτησης Στοιχείου σε Πίνακα;

27.

a. Πολλαπλή Αναζήτηση
b. Σειριακή Αναζήτηση (άμεσο τερματισμό της επανάληψης μόλις βρεθεί το στοιχείο)
c. Δυαδική Αναζήτηση (σε ταξινομημένο πίνακα)

28. Μειονεκτήματα Πινάκων

a. Οι πίνακες απαιτούν μνήμη.
b. Περιορίζουν τις δυνατότητες του προγράμματος επειδή είναι περιορισμένης και προκαθορισμένης χωρητικότητας.

Η χρήση τους πάντως είναι απαραίτητη αν τα δεδομένα που εισάγονται σε ένα πρόγραμμα πρέπει να διατηρηθούν μέχρι το τέλος της εκτέλεσης.

Κεφάλαιο 6

29. Φυσικές – Τεχνητές γλώσσες και Διαφορές

Φυσική γλώσσα είναι η γλώσσα που χρησιμοποιούν οι άνθρωποι για την μεταξύ τους επικοινωνία. Η επιστήμη που τις μελετά είναι η γλωσσολογία.

Τεχνητές γλώσσες είναι όλες οι γλώσσες προγραμματισμού. Ακολουθούν τις βασικές αρχές της γλωσσολογίας και αναπτύχθηκαν για να μπορεί ο προγραμματιστής να δίνει εντολές που πρέπει να εκτελέσει ο Υπολογιστής.

Κύρια διαφορά: Οι τεχνητές γλώσσες διακρίνονται από στασιμότητα και κατασκευάζονται για έναν ειδικό σκοπό. Μπορούν όμως να αλλάζουν σε επίπεδο διαλέκτου ή επίπεδο επέκτασης. Η φυσικές γλώσσες αλλάζουν συνεχώς με την προσθήκη νέων λέξεων και κανόνων γραμματικής και σύνταξης.

30. Τεχνικές Σχεδίασης

Σχεδίαση είναι η ανάπτυξη κανόνων και μεθοδολογιών και τεχνικών προγραμματισμού με σκοπό:

- ✓ Τη δημιουργία απλών και κομψών προγραμμάτων.
- ✓ Την εύκολη γραφή και κατανόησή τους.

Ιεραρχική σχεδίαση: διάσπαση λειτουργιών σε άλλες μικρότερες, ακολουθώντας μία πορεία «από επάνω προς τα κάτω». Χρησιμοποιεί την τεχνική της συνεχούς διαίρεσης του προβλήματος σε υποπροβλήματα.

Τμηματικός προγραμματισμός: Είναι η υλοποίηση της ιεραρχικής σχεδίασης (υποπροβλήματα-ενότητες-modules). Μετά την ανάλυση του προβλήματος σε υποπροβλήματα κάθε υποπρόβλημα αποτελεί ανεξάρτητη ενότητα (module) που γράφεται ξεχωριστά από το υπόλοιπα τμήματα του προγράμματος.
Πλεονεκτήματα:

- ✓ Διευκόλυνση δημιουργίας προγράμματος.
- ✓ Μειώνει τα λάθη.
- ✓ Εύκολη παρακολούθηση, κατανόηση, συντήρηση από τρίτους.

Δομημένος προγραμματισμός: Η μεθοδολογία που έχει επικρατήσει απόλυτα σήμερα και σχεδόν όλες οι σύγχρονες γλώσσες προγραμματισμού την υποστηρίζουν.

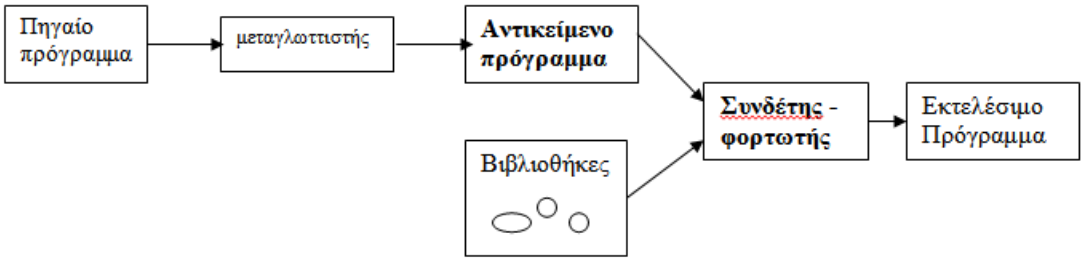
- Ο δομημένος προγραμματισμός στηρίζεται στη χρήση τριών και μόνον στοιχειωδών λογικών δομών:
  - ✓ Δομή ακολουθίας.
  - ✓ Δομή επιλογής.
  - ✓ Δομή επανάληψης.

- Όλα τα προγράμματα μπορούν να γραφούν χρησιμοποιώντας μόνον αυτές τις τρεις δομές καθώς και συνδυασμό τους.
- Κάθε πρόγραμμα καθώς και κάθε ενότητα προγράμματος έχουν:
  - ✓ Μόνον μία είσοδο και μόνον μία έξοδο.
  - ✓

Πλεονεκτήματα Δομημένου Προγραμματισμού:

- Απλούστερα προγράμματα.
- Άμεση μεταφορά αλγορίθμων σε προγράμματα.
- Διευκόλυνση ανάλυσης προγράμματος (τμήματα).
- Διευκόλυνση ανάγνωσης και κατανόησης του προγράμματος από τρίτους.
- Περιορισμός λαθών.
- Ευκολότερη συντήρηση-διόρθωση.

31. Αναλύστε τα στάδια για την δημιουργία ενός εκτελέσιμου προγράμματος:



32. Τι είναι ο Μεταγλωττιστής (Compiler);

Δέχεται στην είσοδο ένα πρόγραμμα (Πηγαίο (source)) γραμμένο σε κάποια γλώσσα υψηλού επιπέδου και παράγει ισοδύναμο σε γλώσσα μηχανής ικανό να εκτελεστεί από τον υπολογιστή (Εκτελέσιμο (executable)). Αυτό εκτελείται σε οποιονδήποτε υπολογιστή και είναι ανεξάρτητο από το αρχικό πρόγραμμα.

33. Τι κάνει ο Διερμηνευτής (Interpreter);

Διαβάζει μία προς μία τις εντολές του αρχικού προγράμματος και για κάθε μια εκτελεί αμέσως μια ισοδύναμη ακολουθία εντολών σε γλώσσα μηχανής.

Πλεονέκτημα:

Άμεση εκτέλεση και συνεπώς άμεση διόρθωση λαθών.

Μειονέκτημα:

Αργή εκτέλεση προγράμματος

Παρατηρήσεις

a. Στα σύγχρονα προγραμματιστικά περιβάλλονται χρησιμοποιείται συνήθως Διερμηνευτής κατά τη φάση της δημιουργίας του προγράμματος και μεταγλωττιστής για την τελική έκδοση και εκμετάλλευση του προγράμματος.
b. Για την αρχική σύνταξη και τη διόρθωσή των προγραμμάτων χρησιμοποιείται ένα ειδικό πρόγραμμα συντάκτης(editor) που είναι ένας μικρός επεξεργαστής κειμένου

34. Λάθη κατά την εκτέλεση ενός προγράμματος :

- ✓ συντακτικά
- ✓ λογικά
- ✓ εκτέλεσης

Τα **λάθη εκτέλεσης** είναι αυτά που κάνουν την εμφάνιση τους κατά την εκτέλεση τουπρογράμματος, όπως για παράδειγμα η διαίρεση με το μηδέν, προκαλώντας τον απότομο τερματισμό του προγράμματος.

Τα **λογικά λάθη** είναι τα πιο επικίνδυνα απ' όλα γιατί δεν «φαίνονται». Είναι λάθη που οφείλονται στη σχεδίαση του αλγόριθμου. Αν για παράδειγμα για τον υπολογισμοί μέσου όρου 5 αριθμών διαιρείς το άθροισμα τους με το 6, το πρόγραμμα θα . εκτελεστεί κανονικά αλλά το αποτέλεσμα δεν θα είναι σωστό.

Τα **συντακτικά λάθη** είναι τα πιο ακίνδυνα και οφείλονται σε λάθος σύνταξη μιας εντολής, σε παράλειψη δήλωσης μιας μεταβλητής και σε άλλα τέτοιας μορφής λάθη, τα οποία πρέπει οπωσδήποτε να διορθωθούν για να παραχθεί το τελικό εκτελέσιμο πρόγραμμα.Τα συντακτικά λάθη θεωρούνται ακίνδυνα, γιατί ανιχνεύονται από τον μεταγλωττιστή ή το διερμηνευτή, οι οποίοι εμφανίζουν τα κατάλληλα μηνύματα λάθους για τη διόρθωση τους. Όταν διορθωθούν τα λάθη που ανιχνεύτηκαν, το διορθωμένο πρόγραμμα επαναϋποβάλεται για μεταγλώττιση και συνεχίζεται η ίδια διαδικασία μέχρι το πρόγραμμα να μην περιέχει κανένα συντακτικό λάθος.

Κεφάλαιο 10

35. Τι είναι 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

36. Ποιες οι ιδιότητες του τμηματικού προγραμματισμού;

a. Κάθε υποπρόγραμμα έχει μόνο μία είσοδο και μία έξοδο.
b. Κάθε υποπρόγραμμα πρέπει να είναι ανεξάρτητο από τα άλλα
c. Κάθε υποπρόγραμμα πρέπει να μην είναι πολύ μεγάλο

37. Ποια τα πλεονεκτήματα:

- 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
- Επεκτείνει τις δυνατότητες των γλωσσών προγραμματισμού

38. Τι ονομάζουμε παράμετρο;

Οι τιμές που περνούν από το ένα υποπρόγραμμα στο άλλο λέγονται παράμετροι

39. Τι ονομάζουμε Διαδικασία ;

Διαδικασία είναι ένας τύπος υποπρογράμματος που μπορεί να εκτελεί όλες τις λειτουργίες ενός προγράμματος Ας δούμε πως υλοποιείται το σενάριο με το μηχάνημα ανάληψης

40. Τι ονομάζουμε Συνάρτηση;

Συνάρτηση είναι ένας τύπος υποπρογράμματος που υπολογίζει και επιστρέφει μόνο μία τιμή με το όνομά της

41. Τυπικές και Πραγματικές Παράμετροι

Τα υποπρογράμματα ενεργοποιούνται από κάποιο άλλο πρόγραμμα ή υποπρόγραμμα για να εκτελέσουν συγκεκριμένες λειτουργίες. Κάθε υποπρόγραμμα για να ενεργοποιηθεί καλείται, όπως λέγεται, από ένα άλλο υποπρόγραμμα ή το αρχικό πρόγραμμα, το οποίο ονομάζεται κύριο πρόγραμμα. Το υποπρόγραμμα είναι αυτόνομο και ανεξάρτητο τμήμα προγράμματος, αλλά συχνά πρέπει να επικοινωνεί με το

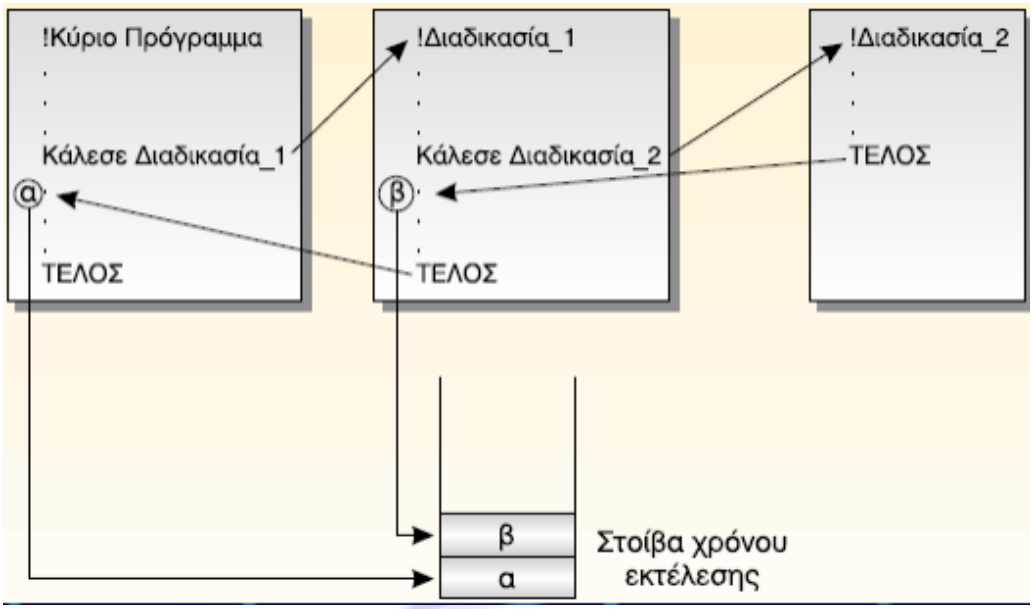
υπόλοιπο πρόγραμμα. Συνήθως δέχεται τιμές από το τμήμα προγράμματος που το καλεί και μετά την εκτέλεση επιστρέφει σε αυτό νέες τιμές, αποτελέσματα. Οι τιμές αυτές που περνούν από το ένα υποπρόγραμμα στο άλλο λέγονται παράμετροι. Οι παράμετροι λοιπόν είναι σαν τις κοινές μεταβλητές ενός προγράμματος

Όταν μία παράμετρος γράφεται στο τμήμα δήλωσης του υποπρογράμματος χαρακτηρίζεται ως Τυπική Παράμετρος, ενώ όταν γράφεται στην εντολή κλήσης του υποπρογράμματος ονομάζεται Πραγματική Παράμετρος. Κατά την κλήση ενός υποπρογράμματος οι τιμές των πραγματικών παραμέτρων «περνάνε» στις τυπικές και μετά την ολοκλήρωση του, οι νέες τιμές που έχουν πάρει οι τυπικές παράμετροι «περνάνε» και πάλι στις πραγματικές παραμέτρους.
Είναι σαφές ότι:

- 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 δεδομένων (π.χ. να είναι και οι δύο πραγματικές, κλπ)
- Σε κάποιες γλώσσες προγραμματισμού οι τυπικές παράμετροι αναφέρονται σαν ορίσματα, ενώ οι πραγματικές απλώς σαν παράμετροι.

42. 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 (returnaddress), 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. Το παράδειγμα αυτό δείχνει μία από τις πολλές χρησιμότητες της LIFO ιδιότητας της στοίβας



43. Εμβέλεια Μεταβλητών –Σταθερών – Ορισμός – Κατηγορίες

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. 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 διαμέσου των παραμέτρων κατά το στάδιο της κλήσης του υποπρογράμματος και μετά το τέλος της εκτέλεσης του υποπρογράμματος. Ας δούμε τον παρακάτω σκελετό προγράμματος.

```
ΠΡΟΓΡΑΜΜΑ Αρχικό
ΜΕΤΑΒΛΗΤΕΣ
   ΠΡΑΓΜΑΤΙΚΕΣ : Α, Β, Γ
ΑΡΧΗ
...
ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
ΔΙΑΔΙΚΑΣΙΑ Πρώτη
ΜΕΤΑΒΛΗΤΕΣ
   ΠΡΑΓΜΑΤΙΚΕΣ : Δ, Ε, Ζ, Η
ΑΡΧΗ
...
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
ΔΙΑΔΙΚΑΣΙΑ Δεύτερη
ΜΕΤΑΒΛΗΤΕΣ
   ΑΚΕΡΑΙΕΣ : Γ, Θ, Ι
ΑΡΧΗ
...
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ
```

Οι μεταβλητές του προγράμματος Αρχικό με ονόματα Α, Β, Γ είναι γνωστές, ισχύουν μόνο για το πρόγραμμα. Έξω από το πρόγραμμα σε όλα τα υποπρογράμματα οι μεταβλητές αυτές δεν ισχύουν. Επίσης η διαδικασία Πρώτη έχει τις πραγματικές μεταβλητές Δ, Ε, Ζ, Η, οι οποίες ισχύουν μόνο για τη συγκεκριμένη διαδικασία και όχι για τα υπόλοιπα υποπρογράμματα ή το κύριο πρόγραμμα. Η διαδικασία Δεύτερη έχει και αυτή τις δικές της μεταβλητές Γ, Θ, Ι. Οι μεταβλητές αυτές ισχύουν μόνο για τη διαδικασία Δεύτερη. Η μεταβλητή με το όνομα Γ δεν έχει καμία σχέση με τη μεταβλητή Γ του κύριου προγράμματος, η μία είναι τύπου Ακεραίου και η άλλη τύπου Πραγματικού. Αφού όλες οι μεταβλητές είναι τοπικές, το ίδιο όνομα μεταβλητής μπορεί να εμφανίζεται σε διαφορετικά τμήματα προγράμματος, χωρίς να αντιστοιχεί στην ίδια μεταβλητή. Το ίδιο έγινε και με τη μεταβλητή R της ακτίνας του κύκλου στο παράδειγμα 2. Ό,τι ισχύει για τις μεταβλητές ισχύει και για τις σταθερές. 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.

Ορισμός:

Το τμήμα του προγράμματος που ισχύουν οι μεταβλητές λέγεται εμβέλεια (scope) 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.