Κεφάλαιο 1

Τι είναι πρόβλημα

Με τον όρο πρόβλημα εννοείται μια κατάσταση η οποία χρήζει αντιμετώπισης, απαιτεί λύση, η δε λύση της δεν είναι γνωστή, ούτε προφανής.

Παράγοντες κατανόησης προβλήματος

Η κατανόηση ενός προβλήματος αποτελεί συνάρτηση δύο παραγόντων, της σωστής διατύπωσης εκ μέρους του δημιουργού του και της αντίστοιχα σωστής ερμηνείας από τη μεριά εκείνου που καλείται να το αντιμετωπίσει.

Τι είναι δεδομένο

Με τον όρο δεδομένο δηλώνεται οποιοδήποτε στοιχείο μπορεί να γίνει αντιληπτό από έναν τουλάχιστον παρατηρητή με μία από τις πέντε αισθήσεις του.

Τι είναι πληροφορία

Με τον όρο πληροφορία αναφέρεται οποιοδήποτε γνωσιακό στοιχείο προέρχεται από επεξεργασία δεδομένων.

Τι είναι η επεξεργασία δεδομένων

Ο όρος επεξεργασία δεδομένων δηλώνει εκείνη τη διαδικασία κατά την οποία ένας "μηχανισμός" δέχεται δεδομένα, τα επεξεργάζεται σύμφωνα με έναν προκαθορισμένο τρόπο και αποδίδει πληροφορίες.

Τι λέμε δομή προβλήματος

Με τον όρο δομή προβλήματος αναφερόμαστε στα συστατικά του μέρη, στα επιμέρους τμήματα που το αποτελούν καθώς επίσης και στον τρόπο που αυτά τα μέρη συνδέονται μεταξύ τους.

Διαγραμματική αναπαράσταση προβλήματος

Για τη γραφική απεικόνιση της δομής ενός προβλήματος χρησιμοποιείται συχνότατα η διαγραμματική αναπαράσταση. Σύμφωνα με αυτή:

- το αρχικό πρόβλημα αναπαρίσταται από ένα ορθογώνιο παραλληλόγραμμο,
- κάθε ένα από τα απλούστερα προβλήματα στα οποία αναλύεται ένα οποιοδήποτε πρόβλημα, αναπαρίσταται επίσης από ένα ορθογώνιο παραλληλόγραμμο,
- τα παραλληλόγραμμα που αντιστοιχούν στα απλούστερα προβλήματα στα οποία αναλύεται ένα οποιοδήποτε πρόβλημα, σχηματίζονται ένα επίπεδο χαμηλότερα. Έτσι σε κάθε κατώτερο επίπεδο, δημιουργείται η γραφική αναπαράσταση των προβλημάτων στα οποία αναλύονται τα προβλήματα του αμέσως ψηλότερου επιπέδου.

Προϋποθέσεις σωστής επίλυσης προβλήματος

Η σωστή επίλυση ενός προβλήματος προϋποθέτει τον επακριβή προσδιορισμό των δεδομένων που παρέχει το πρόβλημα. Απαιτεί επίσης την λεπτομερειακή καταγραφή των ζητούμενων που αναμένονται σαν αποτελέσματα της επίλυσης του προβλήματος

Ποια είναι τα στάδια αντιμετώπισης ενός προβλήματος

- **κατανόηση**, όπου απαιτείται η σωστή και πλήρης αποσαφήνιση των δεδομένων και των ζητούμενων του προβλήματος
- **ανάλυση**, όπου το αρχικό πρόβλημα διασπάται σε άλλα επί μέρους απλούστερα προβλήματα
- **επίλυση**, όπου υλοποιείται η λύση του προβλήματος, μέσω της λύσης των επιμέρους προβλημάτων.

Με ποια κριτήρια κατηγοριοποιούμε τα προβλήματα

1. Με τη δυνατότητα επίλυσής τους
2. Με το βαθμό δόμησης των λύσεών τους
3. Με το είδος της επίλυσης που επιζητούν

Ποιες οι κατηγορίες προβλημάτων με κριτήριο τη ΔΥΝΑΤΟΤΗΤΑ ΕΠΙΛΥΣΗΣ τους

- **Επιλύσιμα**, είναι εκείνα τα προβλήματα για τα οποία η λύση τους είναι ήδη γνωστή και έχει διατυπωθεί. Επιλύσιμα μπορεί επίσης να χαρακτηριστούν και προβλήματα, των οποίων η λύση δεν έχει ακόμα διατυπωθεί, αλλά ή συνάφειά τους με άλλα ήδη επιλυμένα μας επιτρέπει να θεωρούμε σαν βέβαιη τη δυνατότητα επίλυσης τους.
- **Ανοικτά**, ονομάζονται εκείνα τα προβλήματα για τα οποία η λύση τους δεν έχει μεν ακόμα βρεθεί, αλλά παράλληλα δεν έχει αποδειχθεί, ότι δεν επιδέχονται λύση. Σαν παράδειγμα ανοικτού προβλήματος μπορούμε να αναφέρουμε το πρόβλημα της ενοποίησης των τεσσάρων πεδίων δυνάμεων.
- **Άλυτα**, χαρακτηρίζονται εκείνα τα προβλήματα για τα οποία έχουμε φτάσει στην παραδοχή, ότι δεν επιδέχονται λύση. Τέτοιου είδους πρόβλημα είναι το γνωστό από τους αρχαίους ελληνικούς χρόνους πρόβλημα του τετραγωνισμού του κύκλου. Το πρόβλημα αυτό θεωρείται άλυτο, στην πραγματικότητα η λύση που επιδέχεται είναι προσεγγιστική.

Ποιες οι κατηγορίες προβλημάτων με κριτήριο το ΒΑΘΜΟ ΔΟΜΗΣΗΣ των λύσεών τους

- **Δομημένα**, χαρακτηρίζονται εκείνα τα προβλήματα των οποίων η επίλυση προέρχεται από μια αυτοματοποιημένη διαδικασία. Για παράδειγμα, η επίλυση της δευτεροβάθμιας εξίσωσης αποτελεί ένα δομημένο πρόβλημα, αφού ο τρόπος επίλυσης της εξίσωσης είναι γνωστός και αυτοματοποιημένος.
- **Ημιδομημένα**, ονομάζονται τα προβλήματα εκείνα των οποίων η λύση επιδιώκεται στα πλαίσια ενός εύρους πιθανών λύσεων, αφήνοντας στον ανθρώπινο παράγοντα περιθώρια επιλογής της. Σαν παράδειγμα ημιδομημένου προβλήματος μπορούμε να αναφέρουμε ένα πρόβλημα όπου ένας ταξιδιώτης αναζητά να επιλέξει το μεταφορικό μέσο μετακίνησής του από ένα μέρος σε κάποιο άλλο. Το πρόβλημα είναι ημιδομημένο, δεδομένου ότι η λύση που θα επιλεγεί, πρέπει να αναζητηθεί σε ένα σύνολο σαφώς προκαθορισμένο που συμπεριλαμβάνει όλα τα διαθέσιμα μεταφορικά μέσα.

- **Αδόμητα**, χαρακτηρίζονται τα προβλήματα εκείνα των οποίων οι λύσεις δεν μπορούν να δομηθούν ή δεν έχει διερευνηθεί σε βάθος η δυνατότητα δόμησής τους. Πρωτεύοντα ρόλο στην επίλυση αυτού του τύπου προβλημάτων κατέχει η ανθρώπινη διαίσθηση. Παράδειγμα αδόμητου προβλήματος είναι η επιλογή του τρόπου, του τόπου και του χρόνου ενός εφηβικού πάρτυ.

Ποιες οι κατηγορίες προβλημάτων με κριτήριο το ΕΙΔΟΣ ΕΠΙΛΥΣΗΣ τους

- **Απόφασης**, όπου η απόφαση που πρόκειται να ληφθεί σαν λύση του προβλήματος που τίθεται, απαντά σε ένα ερώτημα και πιθανόν αυτή η απάντηση να είναι ένα "Ναι" ή ένα "Όχι". Αυτό που θέλουμε να διαπιστώσουμε σε ένα πρόβλημα απόφασης είναι αν υπάρχει απάντηση που ικανοποιεί τα δεδομένα που θέτονται από το πρόβλημα.

  Παράδειγμα: Δίδεται ένας ακέραιος αριθμός Ν και το πρόβλημα που τίθεται είναι, αν ο αριθμός Ν είναι πρώτος.

- **Υπολογιστικά**, όπου το πρόβλημα που τίθεται απαιτεί τη διενέργεια υπολογισμών, για να μπορεί να δοθεί μία απάντηση στο πρόβλημα. Σε ένα υπολογιστικό πρόβλημα ζητάμε να βρούμε τη τιμή της απάντησης που ικανοποιεί τα δεδομένα που παρέχει το πρόβλημα.

  Παράδειγμα: Δίδεται ένας ακέραιος αριθμός Ν και ζητείται να βρεθεί πόσες διαφορετικές παραγοντοποιήσεις του Ν υπάρχουν.

- **Βελτιστοποίησης**, όπου το πρόβλημα που τίθεται επιζητά το βέλτιστο αποτέλεσμα για τα συγκεκριμένα δεδομένα που διαθέτει. Σε ένα πρόβλημα βελτιστοποίησης αναζητούμε την απάντηση που ικανοποιεί κατά τον καλύτερο τρόπο τα δεδομένα που παρέχει το πρόβλημα.

  Παράδειγμα: Δίδεται ένας ακέραιος αριθμός Ν και ζητείται ποια είναι η παραγοντοποίηση για το Ν με το μεγαλύτερο πλήθος παραγόντων.

Για ποιους λόγους ανατίθεται η επίλυση ενός προβλήματος σε υπολογιστή

- την πολυπλοκότητα των υπολογισμών,
- την επαναληπτικότητα των διαδικασιών,
- την ταχύτητα εκτέλεσης των πράξεων,
- το μεγάλο πλήθος των δεδομένων.

Ποιες λειτουργίες επιτελεί ένας υπολογιστής

- **πρόσθεση**, η οποία αποτελεί τη βασική αριθμητική πράξη, δεδομένου ότι και οι άλλες αριθμητικές πράξεις μπορούν να αντιμετωπιστούν, σαν διαδικασίες πρόσθεσης
- **σύγκριση**, η οποία συνιστά τη βασική λειτουργία για την επιτέλεση όλων των λογικών πράξεων,
- **μεταφορά δεδομένων**, λειτουργία που προηγείται και έπεται της επεξεργασίας δεδομένων.

Κεφάλαιο 2

Τι είναι αλγόριθμος

Αλγόριθμος είναι μια πεπερασμένη σειρά ενεργειών, αυστηρά καθορισμένων και εκτελέσιμων σε πεπερασμένο χρόνο, που στοχεύουν στην επίλυση ενός προβλήματος

Τα κριτήρια που κάθε αλγόριθμος πρέπει απαραίτητα να ικανοποιεί

- **Είσοδος**. Καμία, μία ή περισσότερες τιμές δεδομένων πρέπει να δίνονται ως είσοδοι στον αλγόριθμο. Η περίπτωση που δεν δίνονται τιμές δεδομένων εμφανίζεται, όταν ο αλγόριθμος δημιουργεί και επεξεργάζεται κάποιες πρωτογενείς τιμές με τη βοήθεια συναρτήσεων παραγωγής τυχαίων αριθμών ή με τη βοήθεια άλλων απλών εντολών.
- **Έξοδος**. Ο αλγόριθμος πρέπει να δημιουργεί τουλάχιστον μία τιμή δεδομένων ως αποτέλεσμα προς το χρήστη ή προς έναν άλλο αλγόριθμο.
- **Καθοριστικότητα**. Κάθε εντολή πρέπει να καθορίζεται χωρίς καμία αμφιβολία για τον τρόπο εκτέλεσής της. Λόγου χάριν, μία εντολή διαίρεσης πρέπει να θεωρεί και την περίπτωση, όπου ο διαιρέτης λαμβάνει μηδενική τιμή.
- **Περατότητα**. Ο αλγόριθμος να τελειώνει μετά από πεπερασμένα βήματα εκτέλεσης των εντολών του. Μία διαδικασία που δεν τελειώνει μετά από ένα συγκεκριμένο αριθμό βημάτων δεν αποτελεί αλγόριθμο, αλλά λέγεται απλά υπολογιστική διαδικασία.
- **Αποτελεσματικότητα**. Κάθε μεμονωμένη εντολή του αλγορίθμου να είναι απλή. Αυτό σημαίνει ότι μία εντολή δεν αρκεί να έχει ορισθεί, αλλά πρέπει να είναι και εκτελέσιμη.

Τρόποι αναπαράστασης αλγορίθμων

- **με ελεύθερο κείμενο**, που αποτελεί τον πιο ανεπεξέργαστο και αδόμητο τρόπο παρουσίασης αλγορίθμου. Έτσι εγκυμονεί τον κίνδυνο ότι μπορεί εύκολα να οδηγήσει σε μη εκτελέσιμη παρουσίαση παραβιάζοντας το τελευταίο χαρακτηριστικό των αλγορίθμων, δηλαδή την αποτελεσματικότητα.
- **με διαγραμματικές τεχνικές**, που συνιστούν ένα γραφικό τρόπο παρουσίασης του αλγορίθμου. Από τις διάφορες διαγραμματικές τεχνικές που έχουν επινοηθεί, η πιο παλιά και η πιο γνωστή ίσως, είναι το διάγραμμα ροής. Ωστόσο η χρήση διαγραμμάτων ροής για την παρουσίαση αλγορίθμων δεν αποτελεί την καλύτερη λύση, γι'αυτό και εμφανίζονται όλο και σπανιότερα στη βιβλιογραφία και στην πράξη.
- **με φυσική γλώσσα** κατά βήματα. Στην περίπτωση αυτή χρειάζεται προσοχή, γιατί μπορεί να παραβιασθεί το τρίτο βασικό χαρακτηριστικό ενός αλγορίθμου, όπως προσδιορίσθηκε προηγουμένως, δηλαδή το κριτήριο του καθορισμού.
- **με κωδικοποίηση**, δηλαδή με ένα πρόγραμμα που όταν εκτελεσθεί θα δώσει τα ίδια αποτελέσματα με τον αλγόριθμο.

Βασικές συνιστώσες αλγορίθμου

Είναι οι δομές ακολουθίας, επιλογής και επανάληψης.

Σύμβολα διαγραμμάτων ροής

Τα κυριότερα χρησιμοποιούμενα γεωμετρικά σχήματα είναι τα εξής:

- **έλλειψη** , που δηλώνει την αρχή και το τέλος του κάθε αλγορίθμου,
- **ρόμβος** , που δηλώνει μία ερώτηση με δύο ή περισσότερες εξόδους για απάντηση,
- **ορθογώνιο** , που δηλώνει την εκτέλεση μίας ή περισσότερων πράξεων, και
- **πλάγιο παραλληλόγραμμο**, που δηλώνει είσοδο ή έξοδο στοιχείων.

Τι είναι οι σταθερές

Με τον όρο **σταθερές** αναφερόμαστε σε προκαθορισμένες τιμές που παραμένουν αμετάβλητες σε όλη τη διάρκεια της εκτέλεσης ενός αλγορίθμου. Οι σταθερές διακρίνονται σε:

Αριθμητικές, στις οποίες χρησιμοποιούνται οι αριθμοί, το +, το - και η τελεία ως υποδιαστολή.
πχ. 123 +5 -1.25

Αλφαριθμητικές που σχηματίζονται από οποιουσδήποτε χαρακτήρες εντός εισαγωγικών.
πχ. "Τιμή" "Νούμερο 2" "+Χ-3"

Λογικές που είναι ακριβώς δύο, Αληθής και Ψευδής

Τι είναι η μεταβλητή

Μια **μεταβλητή** είναι ένα γλωσσικό αντικείμενο, που χρησιμοποιείται για να παραστήσει ένα στοιχείο δεδομένου. Στη μεταβλητή εκχωρείται μια τιμή, η οποία μπορεί να αλλάζει κατά τη διάρκεια εκτέλεσης του αλγορίθμου. Ανάλογα με το είδος της τιμής που μπορούν να λάβουν, οι μεταβλητές διακρίνονται σε αριθμητικές, αλφαριθμητικές και λογικές.

Τι είναι οι τελεστές, ποιες οι κατηγορίες τους και ποια η ιεραρχία τους

Πρόκειται για τα γνωστά σύμβολα που χρησιμοποιούνται στις διάφορες πράξεις. Οι **τελεστές** διακρίνονται σε:

αριθμητικούς: + - * / ^ DIV MOD

συγκριτικούς: = <> > >= < <=

λογικούς: ΟΧΙ (άρνηση) ΚΑΙ (σύζευξη) Ή (διάζευξη)

Η ιεραρχία τους είναι αυτή που φαίνεται παραπάνω, δηλαδή πρώτοι οι αριθμητικοί και τελευταίοι οι λογικοί.

Τι είναι οι εκφράσεις

Οι **εκφράσεις** διαμορφώνονται από τους τελεστέους, που είναι σταθερές και μεταβλητές και από τους τελεστές. Η διεργασία αποτίμησης μιας έκφρασης συνίσταται στην απόδοση τιμών στις μεταβλητές και στην εκτέλεση των πράξεων. Η τελική τιμή μιας έκφρασης εξαρτάται από την ιεραρχία των πράξεων και τη χρήση των παρενθέσεων. Μια έκφραση μπορεί να αποτελείται από μια μόνο μεταβλητή ή σταθερά μέχρι μια πολύπλοκη μαθηματική παράσταση.

Ποια η προτεραιότητα των πράξεων σε μία έκφραση

1. Ύψωση σε δύναμη
2. Πολλαπλασιασμός και διαίρεση
3. Πρόσθεση και αφαίρεση

Όταν η ιεραρχία είναι ίδια, τότε οι πράξεις εκτελούνται από τα αριστερά προς τα δεξιά. Η ιεραρχία μπορεί να αλλάξει με τη χρήση παρενθέσεων.

Τι είναι συνθήκη

Η **συνθήκη** είναι μία λογική έκφραση και μπορεί να πάρει τις τιμές **Αληθής** ή **Ψευδής**.

Πίνακας αληθείας

Πρόταση Α	Πρόταση Β	Α ή Β	Α και Β	όχι Α
Αληθής	Αληθής	Αληθής	Αληθής	Ψευδής
Αληθής	Ψευδής	Αληθής	Ψευδής	Ψευδής
Ψευδής	Αληθής	Αληθής	Ψευδής	Αληθής
Ψευδής	Ψευδής	Ψευδής	Ψευδής	Αληθής

Τι είναι βρόχος και πόσες φορές εκτελείται

Βρόχος είναι το τμήμα του αλγορίθμου που επαναλαμβάνεται. Στην **Όσο** και στη **Για** μπορεί να μην εκτελεστεί καμία φορά, ενώ στη **Μέχρις_Ότου** εκτελείται τουλάχιστον μία φορά.

Στην **Για** έχουμε προκαθορισμένο πλήθος επαναλήψεων, ενώ στην **Όσο** και τη **Μέχρις_ότου** όχι.

πλήθος επαναλήψεων = 1 + A_M((τ2 - τ1) / β)

Υπολογισμός επαναλήψεων της << Για i από τ1 μέχρι τ2 με_βήμα β >>

Ο τύπος "δουλεύει" στην περίπτωση που πραγματοποιείται μία τουλάχιστον επανάληψη.

Κανόνες στη χρήση εμφωλευμένων βρόχων

- Ο εσωτερικός βρόχος πρέπει να βρίσκεται ολόκληρος μέσα στον εξωτερικό. Ο βρόχος που ξεκινάει τελευταίος, πρέπει να ολοκληρώνεται πρώτος.
- Η είσοδος σε κάθε βρόχο υποχρεωτικά γίνεται από την αρχή του.
- Δεν μπορεί να χρησιμοποιηθεί η ίδια μεταβλητή ως μετρητής δύο ή περισσοτέρων βρόχων που ο ένας βρίσκεται στο εσωτερικό του άλλου.

Ολίσθηση

Η ολίσθηση προς τα αριστερά ισοδυναμεί με πολλαπλασιασμό επί δύο, ενώ η ολίσθηση προς τα δεξιά ισοδυναμεί με την ακέραια διαίρεση διά δύο.

Σχήματα Δομών Επιλογής

Απλή Επιλογή

Αν <συνθήκη> **τότε**
    <εντολή>
Τέλος_αν

Σύνθετη επιλογή

Αν <συνθήκη> **τότε**
    <διαδικασία_1>
αλλιώς
    <διαδικασία_2>
Τέλος_αν

Πολλαπλή επιλογή

Αν <συνθήκη_1> **τότε**
    <διαδικασία_1>
αλλιώς_αν <συνθήκη_2> **τότε**
    <διαδικασία_2>
.....
αλλιώς_αν <συνθήκη_ν> **τότε**
    <διαδικασία_ν>
αλλιώς
    <διαδικασία_αλλιώς>
Τέλος_αν

Σχήματα Δομών Επανάληψης

Επαναληπτικό σχήμα με έλεγχο επανάληψης στην αρχή

Όσο <συνθήκη> **επανάλαβε**
    Διαδικασία
Τέλος_επανάληψης

Επαναληπτικό σχήμα με έλεγχο επανάληψης στο τέλος

Αρχή_επανάληψης
    Διαδικασία
Μέχρις_ότου <συνθήκη>

Επαναληπτικό σχήμα ορισμένων φορών επανάληψης

Για μεταβλητή **από** τ1 **μέχρι** τ2 **με_βήμα** β
    Διαδικασία
Τέλος_επανάληψης

Τι ονομάζουμε εμφωλευμένα ΑΝ

Εμφωλευμένα ΑΝ ονομάζονται δύο ή περισσότερες εντολές της μορφής ΑΝ...ΤΟΤΕ...ΑΛΛΙΩΣ που περιέχονται η μία μέσα στην άλλη.

Πολλαπλασιασμός αλά Ρωσικά

Είναι ο τρόπος που χρησιμοποιείται από τους υπολογιστές προκειμένου να πολλαπλασιαστούν δύο ακέραιοι αριθμοί.

Ακολουθείται μία επαναλαμβανόμενη διαδικασία (αλγόριθμος) η οποία απαιτεί μόνο πολλαπλασιασμό επί δύο (ολίσθηση προς τα αριστερά), διαίρεση διά δύο (ολίσθηση προς τα δεξιά) και πρόσθεση.

Αλγόριθμος πολλαπλασιασμού αλά Ρωσικά

```
Αλγόριθμος Πολλαπλασιασμός_αλά_ρωσικά
Δεδομένα // M1, M2 //                          ! ακέραιοι
P ← 0
Όσο M2 > 0 επανάλαβε
    Αν M2 mod 2 = 1 τότε
        P ← P + M1
    Τέλος_αν
    M1 ← M1 * 2
    M2 ← M2 div 2
Τέλος_επανάληψης
Αποτελέσματα // P //                           ! το γινόμενο των ακεραίων M1,M2
Τέλος Πολλαπλασιασμός_αλά_ρωσικά
```

Κεφάλαιο 3

Από ποιες σκοπιές μελετά η Πληροφορική τα δεδομένα

- Υλικού
- Γλωσσών προγραμματισμού
- Δομών Δεδομένων
- Ανάλυσης Δεδομένων

Τι είναι Δομή Δεδομένων

Δομή Δεδομένων είναι ένα σύνολο αποθηκευμένων δεδομένων που υφίστανται επεξεργασία από ένα σύνολο λειτουργιών.

Βασικές λειτουργίες (πράξεις) επί των δομών δεδομένων

- **Προσπέλαση**, πρόσβαση σε ένα κόμβο με σκοπό να εξετασθεί ή να τροποποιηθεί το περιεχόμενό του.
- **Εισαγωγή**, δηλαδή η προσθήκη νέων κόμβων σε μία υπάρχουσα δομή.
- **Διαγραφή**, που αποτελεί το αντίστροφο της εισαγωγής, δηλαδή ένας κόμβος αφαιρείται από μία δομή.
- **Αναζήτηση**, κατά την οποία προσπελαύνονται οι κόμβοι μιας δομής, προκειμένου να εντοπιστούν ένας ή περισσότεροι που έχουν μια δεδομένη ιδιότητα.
- **Ταξινόμηση**, όπου οι κόμβοι μιας δομής διατάσσονται κατά αύξουσα ή φθίνουσα σειρά.
- **Αντιγραφή**, κατά την οποία όλοι οι κόμβοι ή μερικοί από τους κόμβους μίας δομής αντιγράφονται σε μία άλλη δομή.
- **Συγχώνευση**, κατά την οποία δύο ή περισσότερες δομές συνενώνονται σε μία ενιαία δομή.
- **Διαχωρισμός**, που αποτελεί την αντίστροφη πράξη της συγχώνευσης.

Εξάρτηση δομής δεδομένων και αλγορίθμου

Υπάρχει μεγάλη εξάρτηση μεταξύ της δομής δεδομένων και του αλγόριθμου που επεξεργάζεται τη δομή. Μάλιστα, το πρόγραμμα πρέπει να θεωρεί τη δομή δεδομένων και τον αλγόριθμο ως μία **αδιάσπαστη ενότητα**.

Αλγόριθμοι + Δομές Δεδομένων = Προγράμματα

Κατηγορίες δομών δεδομένων

Οι δομές δεδομένων διακρίνονται σε δύο μεγάλες κατηγορίες: τις στατικές και τις δυναμικές.

Δυναμική παραχώρηση μνήμης

Είναι η τεχνική με την οποία αποθηκεύονται στη μνήμη του υπολογιστή οι δυναμικές δομές δεδομένων.

Τι είναι οι δυναμικές δομές δεδομένων

Οι δυναμικές δομές δεν αποθηκεύονται σε συνεχόμενες θέσεις μνήμης αλλά στηρίζονται στην τεχνική της λεγόμενης δυναμικής παραχώρησης μνήμης. Δηλαδή οι δυναμικές δομές δεν έχουν σταθερό μέγεθος, αλλά ο αριθμός των κόμβων τους μεγαλώνει και μικραίνει καθώς στη δομή εισάγονται νέα δεδομένα ή διαγράφονται κάποια δεδομένα αντίστοιχα.

Τι είναι οι στατικές δομές δεδομένων

Με τον όρο στατική δομή δεδομένων εννοείται ότι το ακριβές μέγεθος της απαιτούμενης κύριας μνήμης καθορίζεται κατά τη στιγμή του προγραμματισμού τους και κατά συνέπεια κατά τη στιγμή της μετάφρασής τους και όχι κατά τη στιγμή της εκτέλεσης τους προγράμματος. Τα στοιχεία των στατικών δομών αποθηκεύονται σε συνεχόμενες θέσεις μνήμης. Στην πράξη υλοποιούνται με πίνακες.

Στοίβα

Μία στοίβα δεδομένων μοιάζει με μία στοίβα από πιάτα. Τα δεδομένα που βρίσκονται στην κορυφή της στοίβας λαμβάνονται πρώτα, ενώ αυτά που βρίσκονται στο βάθος της στοίβας λαμβάνονται τελευταία. Αυτή η μέθοδος επεξεργασίας ονομάζεται Τελευταίο μέσα, πρώτο έξω ή απλούστερα με την αγγλική συντομογραφία LIFO (Last-In-First-Out).

Κύριες λειτουργίας στοίβας. Τι πρέπει να ελέγχουν

Δύο είναι οι κύριες λειτουργίες σε μία στοίβα:

- η **ώθηση** (push) στοιχείου στην κορυφή της στοίβας, και
- η **απώθηση** (pop) στοιχείου από τη στοίβα.

Η διαδικασία της ώθησης πρέπει οπωσδήποτε να ελέγχει, αν η στοίβα είναι γεμάτη, οπότε λέγεται ότι συμβαίνει υπερχείλιση της στοίβας. Αντίστοιχα, η διαδικασία απώθησης ελέγχει, αν υπάρχει ένα τουλάχιστον στοιχείο στη στοίβα, δηλαδή ελέγχει αν γίνεται υποχείλιση της στοίβας.

Υλοποίηση στοίβας με πίνακα

Μια βοηθητική μεταβλητή (με όνομα συνήθως top) χρησιμοποιείται για να δείχνει το στοιχείο που τοποθετήθηκε τελευταίο στην κορυφή της στοίβας. Για την εισαγωγή ενός νέου στοιχείου στη στοίβα (ώθηση) αρκεί να αυξηθεί η μεταβλητή top κατά ένα και στη θέση αυτή να εισέλθει το στοιχείο. Αντίθετα για την εξαγωγή ενός στοιχείου από τη στοίβα (απώθηση) εξέρχεται πρώτα το στοιχείο που δείχνει η μεταβλητή top και στη συνέχεια η top μειώνεται κατά ένα για να δείχνει τη νέα κορυφή.

Ουρά

Υλοποιεί τη μέθοδο επεξεργασίας κατά την οποία το στοιχείο που εισέρχεται πρώτο είναι αυτό που εξέρχεται και πρώτο (όπως σε μία ουρά αναμονής) και η οποία ονομάζεται Πρώτο μέσα, πρώτο έξω ή απλούστερα ακολουθώντας την αγγλική συντομογραφία FIFO (First-In-First-Out).

Κύριες λειτουργίας ουράς. Τι πρέπει να ελέγχουν

Δύο είναι οι κύριες λειτουργίες που εκτελούνται σε μία ουρά:

- η **εισαγωγή** (enqueue) στοιχείου στο πίσω άκρο της ουράς, και
- η **εξαγωγή** (dequeue) στοιχείου από το εμπρός άκρο της ουράς.

Η διαδικασία της εισαγωγής πρέπει να ελέγχει αν υπάρχει ελεύθερος χώρος στην δομή, ενώ η διαδικασία της εξαγωγής, εάν υπάρχει ένα τουλάχιστον στοιχείο για εξαγωγή.

Υλοποίηση ουράς με πίνακα

Απαιτούνται δύο δείκτες: ο εμπρός (front) και ο πίσω (rear) δείκτης, που μας δίνουν τη θέση του στοιχείου που σε πρώτη ευκαιρία θα εξαχθεί και τη θέση του στοιχείου που μόλις εισήλθε.
Για την εισαγωγή ενός νέου στοιχείου στην ουρά αυξάνεται ο δείκτης rear κατά ένα και στη θέση αυτή αποθηκεύεται το στοιχείο. Αντίστοιχα για τη λειτουργία της εξαγωγής, εξέρχεται το στοιχείο που δείχνει ο δείκτης front, ο οποίος στη συνέχεια αυξάνεται κατά ένα, για να δείχνει το επόμενο στοιχείο που πρόκειται να εξαχθεί.

Αλγόριθμος σειριακής αναζήτησης

```
Αλγόριθμος Sequential_Search
    Δεδομένα // n, table, key //
    done ← ψευδής
    position ← 0
    i ← 1
    Όσο (done=ψευδής) και (i<=n) επανάλαβε
        Αν table[i]=key τότε
            done ← αληθής
            position ← i
            αλλιώς
            i ← i+1
        Τέλος_αν
    Τέλος_επανάληψης
    Αποτελέσματα //done, position //
Τέλος Sequential_Search
```

Πότε δικαιολογείται η χρήση της μεθόδου σειριακής αναζήτησης

Η σειριακή μέθοδος αναζήτησης είναι η πιο απλή, αλλά και η λιγότερη αποτελεσματική μέθοδος αναζήτησης. Έτσι, δικαιολογείται η χρήση της μόνο σε περιπτώσεις όπου:

- ο πίνακας είναι μη ταξινομημένος,
- ο πίνακας είναι μικρού μεγέθους (για παράδειγμα, $n \leq 20$),
- η αναζήτηση σε ένα συγκεκριμένο πίνακα γίνεται σπάνια.

Πως ορίζεται η ταξινόμηση

Δοθέντων των στοιχείων a_1 ,a_2 ,...,a_n η ταξινόμηση συνίσταται στη μετάθεση της θέσης των στοιχείων, ώστε να τοποθετηθούν σε μία σειρά a_{k1} ,a_{k2} ,...,a_{kn} έτσι ώστε, δοθείσης μίας συνάρτησης διάταξης, f, να ισχύει:

$$f(a_{k1}) \leq f(a_{k2}) \leq \ldots \leq f(a_{kn})$$

Δομές Δεδομένων δευτερεύουσας μνήμης

Σε μεγάλες πρακτικές εμπορικές/επιστημονικές εφαρμογές, το μέγεθος της κύριας μνήμης δεν επαρκεί για την αποθήκευση των δεδομένων. Στην περίπτωση αυτή χρησιμοποιούνται ειδικές δομές για την αποθήκευση των δεδομένων στη δευτερεύουσα μνήμη, δηλαδή κυρίως στο μαγνητικό δίσκο. Οι ειδικές αυτές δομές ονομάζονται **αρχεία** (files). Τα στοιχεία ενός αρχείου ονομάζονται **εγγραφές** (records), όπου κάθε εγγραφή αποτελείται από ένα ή περισσότερα **πεδία** (fields), που ταυτοποιούν την εγγραφή, και από άλλα πεδία που περιγράφουν διάφορα χαρακτηριστικά της εγγραφής.

Ταξινόμηση ευθείας ανταλλαγής (φυσσαλίδας)

```
Αλγόριθμος Φυσσαλίδα
    Δεδομένα // table, n //
    Για i από 2 μέχρι n
        Για j από n μέχρι i με_βήμα –1
            Αν table[j-1] > table[j] τότε
                αντιμετάθεσε table[j-1], table[j]
            Τέλος_αν
        Τέλος_επανάληψης
    Τέλος_επανάληψης
    Αποτελέσματα // table //
Τέλος Φυσσαλίδα
```

Εναλλακτική της εντολής αντιμετάθεσε

```
temp ← table[j-1]
table[j-1] ← table[j]
table[j] ← temp
```

Άλλοι αλγόριθμοι ταξινόμησης

Για την ταξινόμηση δεδομένων έχουν εκπονηθεί πάρα πολλοί αλγόριθμοι. Άλλοι σχετικά απλοί αλγόριθμοι είναι η ταξινόμηση με επιλογή και η ταξινόμηση με παρεμβολή. **Ο πιο γρήγορος αλγόριθμος ταξινόμησης είναι η "γρήγορη ταξινόμηση" (quicksort).**
Η ταξινόμηση φυσσαλίδας είναι ο πιο απλός και ταυτόχρονα ο πιο αργός αλγόριθμος ταξινόμησης.

Κεφάλαιο 6

Τι είναι πρόγραμμα

Πρόγραμμα είναι το σύνολο των εντολών που πρέπει να δοθούν στον υπολογιστή, ώστε να υλοποιηθεί ο αλγόριθμος για την επίλυση του προβλήματος.

Ποια είναι τα βασικά στοιχεία ενός προγράμματος

Βασικά στοιχεία του προγράμματος είναι ο αλγόριθμος, τα δεδομένα και οι δομές δεδομένων επί των οποίων ενεργεί.

Στάδια επίλυσης ενός προβλήματος με υπολογιστή. Με ποιο ασχολείται ο προγραμματισμός

Η επίλυση ενός προβλήματος με τον υπολογιστή περιλαμβάνει τρία εξίσου σημαντικά στάδια.

- Τον ακριβή προσδιορισμό του προβλήματος.
- Την ανάπτυξη του αντίστοιχου αλγορίθμου.
- Τη διατύπωση του αλγορίθμου σε κατανοητή μορφή από τον υπολογιστή.
  (με αυτό το στάδιο ασχολείται ο προγραμματισμός, δηλαδή η δημιουργία του προγράμματος)

Γενιές γλωσσών προγραμματισμού

- **Γλώσσες 1ης γενιάς** ή αλλιώς γλώσσες μηχανής. Μια ακολουθία δυαδικών ψηφίων (0 και 1), που αποτελούν εντολές κατανοητές προς τον υπολογιστή, αλλά ακατανόητες για τον άνθρωπο.
- **Γλώσσες 2ης γενιάς** ή συμβολικές γλώσσες ή γλώσσες χαμηλού επιπέδου. Το έργο της μετάφρασης (σε γλώσσα μηχανής) το αναλαμβάνει ένα ειδικό πρόγραμμα, ο συμβολομεταφραστής (assembler).
- **Γλώσσες 3ης γενιάς ή υψηλού επιπέδου**. Περιέχουν περισσότερες εντολές για την εκτέλεση πολύπλοκων εργασιών. Τα προγράμματα μεταφράζονται σε γλώσσα μηχανής είτε από τον μεταγλωττιστή, είτε από τον διερμηνευτή.
- **Γλώσσες 4ης γενιάς**. Είναι εφοδιασμένες με εργαλεία προγραμματισμού που αποκρύπτουν πολλές λεπτομέρειες από τις τεχνικές υλοποίησης και με αυτά ο χρήστης μπορεί να επιλύει μόνος του μικρά προβλήματα εφαρμογών.

Τι είναι ένα πρόγραμμα σε γλώσσα μηχανής

Ένα πρόγραμμα σε **γλώσσα μηχανής** είναι μια ακολουθία δυαδικών ψηφίων, που αποτελούν εντολές προς τον επεξεργαστή για στοιχειώδεις λειτουργίες.

Μειονεκτήματα γλώσσας μηχανής

- Μορφή κατανοητή από τον υπολογιστή αλλά ακατανόητη από τον άνθρωπο.
- Απαιτεί βαθιά γνώση του υλικού και της αρχιτεκτονικής του υπολογιστή.

Εντολές συμβολικής γλώσσας και μετάφρασή τους

Οι εντολές σε συμβολική γλώσσα αποτελούνται από συμβολικά ονόματα που αντιστοιχούν σε εντολές της γλώσσας μηχανής.
Το έργο της μετάφρασης το αναλαμβάνει ένα ειδικό πρόγραμμα, ο **συμβολομεταφραστής**.

Μειονεκτήματα συμβολικών γλωσσών

- Είναι στενά συνδεδεμένες με την αρχιτεκτονική του κάθε υπολογιστή.
- Δεν διαθέτουν εντολές πιο σύνθετων λειτουργιών και οδηγούν έτσι σε μακροσκελή προγράμματα, που είναι δύσκολο να γραφούν και κύρια να συντηρηθούν.
- Τα προγράμματα δεν μπορούν να μεταφερθούν σε άλλον διαφορετικό υπολογιστή, ακόμη και του ίδιου κατασκευαστή.

Οπτικός Προγραμματισμός

Με τον όρο οπτικό εννοούμε τη δυνατότητα να δημιουργούμε γραφικά ολόκληρο το περιβάλλον της εφαρμογής για παράδειγμα τα πλαίσια διαλόγου ή τα μενού.

Οι πιο διαδεδομένες γλώσσες προγραμματισμού σε γραφικό περιβάλλον για προσωπικούς υπολογιστές είναι η Visual Basic, η Visual C++ και η Java.

Οδηγούμενος από το γεγονός Προγραμματισμός

Με τον όρο οδηγούμενο από το γεγονός εννοούμε τη δυνατότητα να ενεργοποιούνται λειτουργίες του προγράμματος με την εκτέλεση ενός γεγονότος, για παράδειγμα την επιλογή μίας εντολής από ένα μενού ή το κλικ του ποντικιού.

Διάφορες γλώσσες υψηλού επιπέδου:

- FORTRAN (μαθηματικά και επιστημονικά προβλήματα)
- COBOL (εμπορικές εφαρμογές)
- ALGOL (γενικής φύσης, με ελάχιστη πρακτική εφαρμογή)
- PL/1 (επιστημονικές και εμπορικές εφαρμογές, χωρίς επιτυχία)
- LISP (εφαρμογές τεχνητής νοημοσύνης)
- PROLOG (εφαρμογές τεχνητής νοημοσύνης)
- BASIC (γλώσσα γενικής χρήσης, κατάλληλη για αρχάριους)
- PASCAL (γλώσσα γενικής χρήσης, κατάλληλη για εκπαίδευση, με παράγωγες τις ADA και MODULA-2)
- C (ισχυρή γλώσσα ανάπτυξης συστημάτων με δυνατότητες χαμηλού επιπέδου)
- C++ (εξέλιξη της C, αντικειμενοστραφής)
- JAVA (αντικειμενοστραφής γλώσσα κατάλληλη για ανάπτυξη εφαρμογών που θα εκτελούνται στο διαδίκτυο)

Πλεονεκτήματα των γλωσσών υψηλού επιπέδου

Στα πλεονεκτήματα των γλωσσών προγραμματισμού υψηλού επιπέδου σε σχέση με τις συμβολικές μπορούν να αναφερθούν:

- Ο φυσικότερος και πιο "ανθρώπινος" τρόπος έκφρασης των προβλημάτων. Τα προγράμματα σε γλώσσα υψηλού επιπέδου είναι πιο κοντά στα προβλήματα που επιλύουν.
- Η ανεξαρτησία από τον τύπο του υπολογιστή. Προγράμματα σε μία γλώσσα υψηλού επιπέδου μπορούν να εκτελεστούν σε οποιονδήποτε υπολογιστή με ελάχιστες ή καθόλου μετατροπές. Η δυνατότητα της **μεταφερσιμότητας** των προγραμμάτων είναι σημαντικό προσόν.
- Η ευκολία της εκμάθησης και εκπαίδευσης ως απόρροια των προηγουμένων.
- Η διόρθωση λαθών και η συντήρηση προγραμμάτων σε γλώσσα υψηλού επιπέδου είναι πολύ ευκολότερο έργο.

Συνολικά οι γλώσσες υψηλού επιπέδου ελάττωσαν σημαντικά το χρόνο και το κόστος παραγωγής νέων προγραμμάτων, αφού λιγότεροι προγραμματιστές μπορούν σε μικρότερο χρόνο να αναπτύξουν προγράμματα που χρησιμοποιούνται σε περισσότερους υπολογιστές.

Ταξινόμηση γλωσσών Προγραμματισμού με βάση τις κατηγορίες προβλημάτων

- **Διαδικασιακές**. Αποτελούν τη μεγάλη πλειοψηφία και είναι γνωστές επίσης και ως αλγοριθμικές γλώσσες, γιατί είναι σχεδιασμένες για να επιτρέπουν την υλοποίηση αλγορίθμων.
- **Αντικειμενοστραφείς** γλώσσες.
- **Συναρτησιακές** γλώσσες, π.χ. LISP
- **Μη διαδικασιακές** γλώσσες, π.χ. PROLOG. Χαρακτηρίζονται επίσης και ως γλώσσες πολύ υψηλού επιπέδου.
- **Γλώσσες ερωταπαντήσεων**, π.χ. SQL.

Ταξινόμηση γλωσσών Προγραμματισμού με βάση τις κατηγορίες προβλημάτων

- Γλώσσες γενικής χρήσης. Θεωρητικά κάθε γλώσσα γενικής χρήσης μπορεί να χρησιμοποιηθεί για την επίλυση οποιουδήποτε προβλήματος. Στην πράξη ωστόσο κάθε γλώσσα έχει σχεδιαστεί για να ανταποκρίνεται καλύτερα σε ορισμένη κατηγορία προβλημάτων. Διακρίνονται σε:
  - **Γλώσσες επιστημονικής κατεύθυνσης**, π.χ. FORTRAN
  - **Γλώσσες εμπορικής κατεύθυνσης**, π.χ. COBOL.

Ας σημειωθεί ότι ορισμένες γλώσσες τα καταφέρνουν εξίσου καλά και στους δύο προηγούμενους τομείς π.χ. BASIC, Pascal.

- **Γλώσσες προγραμματισμού συστημάτων**, π.χ. C.
- **Γλώσσες τεχνητής νοημοσύνης**, π.χ. LISP, PROLOG.
- **Γλώσσες ειδικής χρήσης**. Πρόκειται για γλώσσες που χρησιμοποιούνται σε ειδικές περιοχές εφαρμογών όπως π.χ. στα γραφικά με υπολογιστή, στη ρομποτική, στη σχεδίαση ολοκληρωμένων κυκλωμάτων, στα Συστήματα Διοίκησης Βάσεων Δεδομένων, στην εκπαίδευση μέσω υπολογιστή κ.α.

Από τι εξαρτάται η επιλογή της γλώσσας προγραμματισμού για την ανάπτυξη μίας εφαρμογής

- από το είδος της εφαρμογής,
- το υπολογιστικό περιβάλλον στο οποίο θα εκτελεστεί,
- τα προγραμματιστικά περιβάλλοντα που διαθέτουμε,
- και κυρίως από τις γνώσεις του προγραμματιστή.

Από τι προσδιορίζεται μία γλώσσα

Κάθε γλώσσα προσδιορίζεται από το αλφάβητο της, το λεξιλόγιο της, τη γραμματική της και τη σημασιολογία της.

Διαφορά φυσικών και τεχνητών γλωσσών

Οι φυσικές γλώσσες εξελίσσονται συνεχώς, ενώ οι τεχνητές γλώσσες χαρακτηρίζονται από στασιμότητα, αφού κατασκευάζονται συνειδητά για ένα συγκεκριμένο σκοπό.

Ποιος είναι ο σκοπός της ιεραρχικής σχεδίασης

Σκοπός της ιεραρχικής ή **"από πάνω προς τα κάτω"** σχεδίασης είναι η συνεχής διάσπαση του προβλήματος σε απλούστερα υποπροβλήματα, τα οποία να είναι εύκολο να επιλυθούν οδηγώντας στην επίλυση του αρχικού προβλήματος.

Τι είναι ο τμηματικός προγραμματισμός και τι πλεονεκτήματα έχει

Είναι η υλοποίηση της ιεραρχικής σχεδίασης, όπου κάθε ένα από τα στοιχειώδη υποπροβλήματα αποτελεί ανεξάρτητη ενότητα και προγραμματίζεται ξεχωριστά από τα υπόλοιπα τμήματα.

- Διευκολύνει τη δημιουργία του προγράμματος,
- μειώνει τα λάθη και
- επιτρέπει την ευκολότερη παρακολούθηση, κατανόηση και συντήρηση του προγράμματος από τρίτους.

Τι είναι ο δομημένος προγραμματισμός

Είναι η μεθοδολογία Προγραμματισμού που έχει επικρατήσει και η οποία αναπτύχθηκε από την ανάγκη να υπάρχει μία κοινή μεθοδολογία στην ανάπτυξη των προγραμμάτων και τη μείωση των εντολών GOTO που χρησιμοποιούνται στο πρόγραμμα.
Περιέχει τόσο την ιεραρχική σχεδίαση, όσο και τον τμηματικό προγραμματισμό.

Ποιες είναι οι αρχές του δομημένου προγραμματισμού

- Ο δομημένος προγραμματισμός στηρίζεται στη χρήση τριών και μόνο στοιχειωδών λογικών δομών, τη δομή της ακολουθίας, τη δομή της επιλογής και τη δομή της επανάληψης. Όλα τα προγράμματα μπορούν να γραφούν χρησιμοποιώντας μόνο αυτές τις τρεις δομές καθώς και συνδυασμό τους.
- Κάθε πρόγραμμα όπως και κάθε ενότητα προγράμματος έχει μόνο μία είσοδο και μόνο μία έξοδο.

Ποια είναι τα πλεονεκτήματα του δομημένου προγραμματισμού

- Δημιουργία απλούστερων προγραμμάτων.
- Άμεση μεταφορά των αλγορίθμων σε προγράμματα.
- Διευκόλυνση ανάλυσης του προγράμματος σε τμήματα.
- Περιορισμός των λαθών κατά την ανάπτυξη του προγράμματος.
- Διευκόλυνση στην ανάγνωση και κατανόηση του προγράμματος από τρίτους.
- Ευκολότερη διόρθωση και συντήρηση.

Τι είναι ο αντικειμενοστραφής προγραμματισμός

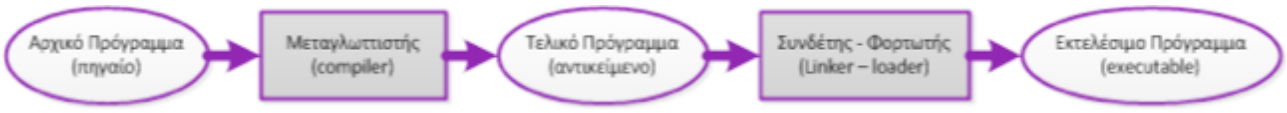
Ο αντικειμενοστραφής προγραμματισμός εκλαμβάνει ως πρωτεύοντα δομικά στοιχεία ενός προγράμματος τα δεδομένα, από τα οποία δημιουργούνται με κατάλληλη μορφοποίηση τα αντικείμενα.
Χρησιμοποιεί την ιεραρχική σχεδίαση, τον τμηματικό προγραμματισμό και ακολουθεί τις αρχές του δομημένου προγραμματισμού.

Τι είναι ο παράλληλος προγραμματισμός

Είναι ο προγραμματισμός που προσπαθεί να εκμεταλλευτεί τις δυνατότητες που προσφέρουν οι υπολογιστές που διαθέτουν περισσότερους του ενός επεξεργαστές, διαιρώντας το πρόβλημα σε τμήματα που μπορούν να εκτελεστούν παράλληλα.

Πως λειτουργεί ο μεταγλωττιστής

1. Λαμβάνει στην είσοδό του ένα πρόγραμμα γραμμένο σε μία γλώσσα υψηλού επιπέδου, το λεγόμενο **πηγαίο** πρόγραμμα.
2. Παράγει ένα ισοδύναμο πρόγραμμα σε γλώσσα μηχανής, το οποίο ονομάζεται **αντικείμενο** πρόγραμμα.
3. Το αντικείμενο πρόγραμμα συνδέεται με άλλα τμήματα προγράμματος που είτε τα γράφει ο προγραμματιστής, είτε βρίσκονται σε βιβλιοθήκες, μέσω ενός προγράμματος που ονομάζεται **συνδέτης – φορτωτής**.
4. Ο συνδέτης παράγει το **εκτελέσιμο** πρόγραμμα, το οποίο είναι αυτό που εκτελείται από τον υπολογιστή.



Πως λειτουργεί ο διερμηνευτής

Διαβάζει μία προς μία τις εντολές του αρχικού προγράμματος (πηγαίο) και για κάθε μία εκτελεί αμέσως μια ισοδύναμη ακολουθία εντολών μηχανής.

Τι είναι τα Συντακτικά λάθη

Είναι αυτά που οφείλονται σε αναγραμματισμούς ονομάτων εντολών, σε παράλειψη δήλωσης δεδομένων κ.τ.λ. (π.χ. " Όσο (α > 0) τότε ") και εντοπίζονται κατά τη μεταγλώττιση από τον μεταγλωττιστή ή το διερμηνευτή.

Τι είναι τα Λογικά λάθη

Είναι αυτά που οφείλονται σε σφάλματα κατά την σχεδίαση-υλοποίηση του αλγορίθμου (π.χ. ΜΟ ← α+β/2), εντοπίζονται κατά την εκτέλεση του προγράμματος και είναι τα πλέον σοβαρά και δύσκολα στη διόρθωσή τους.

Υπέρ και κατά μεταγλωττιστών - διερμηνευτών

Η χρήση μεταγλωττιστή έχει το μειονέκτημα της μεταγλώττισης και σύνδεσης ολόκληρου του προγράμματος πριν από την εκτέλεση σε αντίθεση με την χρήση διερμηνευτή που έχει το πλεονέκτημα της άμεσης εκτέλεσης και διόρθωσης.
Από την άλλη η χρήση διερμηνευτή καθιστά την εκτέλεση του προγράμματος πιο αργή σε σχέση με αυτή του ισοδύναμου εκτελέσιμου προγράμματος που παράγει ο μεταγλωττιστής.

Ποια προγράμματα και εργαλεία πρέπει να περιέχει ένα προγραμματιστικό περιβάλλον

- Συντάκτη
- Μεταγλωττιστή
- Συνδέτη

Κεφάλαια 7 και 8

Ποιους τύπους δεδομένων υποστηρίζει η γλώσσα

Οι τύποι δεδομένων που υποστηρίζει η ΓΛΩΣΣΑ είναι οι αριθμητικοί, που περιλαμβάνουν τους ακέραιους και τους πραγματικούς αριθμούς, οι χαρακτήρες και τέλος οι λογικοί.

Ακέραιος τύπος. Ο τύπος αυτός περιλαμβάνει τους ακέραιους που είναι γνωστοί από τα μαθηματικά. Οι ακέραιοι μπορούν να είναι θετικοί, αρνητικοί ή μηδέν. Παραδείγματα ακεραίων είναι οι αριθμοί 1, 3409, 0, -980.

Πραγματικός τύπος. Ο τύπος αυτός περιλαμβάνει τους πραγματικούς αριθμούς που γνωρίζουμε από τα μαθηματικά. Οι αριθμοί 3.14159, 2.71828, -112.45, 0.45 είναι πραγματικοί αριθμοί. Και οι πραγματικοί αριθμοί μπορούν να είναι θετικοί, αρνητικοί ή μηδέν.

Χαρακτήρας. Ο τύπος αυτός αναφέρεται τόσο σε ένα χαρακτήρα όσο και μία σειρά χαρακτήρων. Τα δεδομένα αυτού του τύπου μπορούν να περιέχουν οποιοδήποτε χαρακτήρα παράγεται από το πληκτρολόγιο. Παραδείγματα χαρακτήρων είναι 'Κ', 'Κώστας', 'σήμερα είναι Τετάρτη', 'Τα πολλαπλάσια του 15 είναι'.
Οι χαρακτήρες πρέπει υποχρεωτικά να βρίσκονται μέσα σε απλά εισαγωγικά, ' '. Τα δεδομένα αυτού του τύπου, επειδή περιέχουν τόσο αλφαβητικούς όσο και αριθμητικούς χαρακτήρες, ονομάζονται συχνά αλφαριθμητικά.

Λογικός. Αυτός ο τύπος δέχεται μόνο δύο τιμές ΑΛΗΘΗΣ και ΨΕΥΔΗΣ. Οι τιμές αντιπροσωπεύουν αληθείς ή ψευδείς συνθήκες.

Ποια είναι η δομή ενός προγράμματος

- Η πρώτη εντολή κάθε προγράμματος είναι υποχρεωτικά η επικεφαλίδα του προγράμματος, η οποία είναι η λέξη ΠΡΟΓΡΑΜΜΑ ακολουθούμενη από το όνομα του προγράμματος. Το τελευταίο πρέπει να υπακούει στους κανόνες δημιουργίας ονομάτων της ΓΛΩΣΣΑΣ.
- Στη συνέχεια ακολουθεί το τμήμα δήλωσης των σταθερών του προγράμματος, αν βέβαια το πρόγραμμα μας χρησιμοποιεί σταθερές.
- Αμέσως μετά είναι το τμήμα δήλωσης μεταβλητών, όπου δηλώνονται υποχρεωτικά τα ονόματα όλων των μεταβλητών καθώς και ο τύπος τους.
- Ακολουθεί το κύριο μέρος του προγράμματος, που περιλαμβάνει όλες τις εκτελέσιμες εντολές. Οι εντολές αυτές περιλαμβάνονται υποχρεωτικά ανάμεσα στις λέξεις ΑΡΧΗ και ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ.
- Τέλος αν το πρόγραμμα χρησιμοποιεί διαδικασίες (βλ. κεφ. 10), αυτές γράφονται μετά το ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ.
- Κάθε εντολή γράφεται σε ξεχωριστή γραμμή. Αν μία εντολή πρέπει να συνεχιστεί και στην επόμενη γραμμή, τότε ο πρώτος χαρακτήρας αυτής της γραμμής πρέπει να είναι ο χαρακτήρας &.
- Αν ο πρώτος χαρακτήρας είναι το θαυμαστικό (!), σημαίνει ότι αυτή η γραμμή περιέχει σχόλια και όχι εκτελέσιμες εντολές.

Διαφορές Αλγορίθμου - Προγράμματος

- Ο αλγόριθμος ξεκινά με Αλγόριθμος <όνομα> και τελειώνει με Τέλος <όνομα>
- Το πρόγραμμα ξεκινά με τη γραμμή ΠΡΟΓΡΑΜΜΑ <όνομα> και τελειώνει με τη γραμμή ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ <όνομα> ή απλά με ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
- Οι εντολές και οι τελεστές στο πρόγραμμα γράφονται με κεφαλαία (πχ. ΔΙΑΒΑΣΕ, DIV, ΟΧΙ)
- Στον αλγόριθμο έχουμε με_βήμα ενώ στο πρόγραμμα ΜΕ ΒΗΜΑ (χωρίς κάτω παύλα)
- Στον αλγόριθμο ως εντολές εξόδου έχουμε τις Εμφάνισε, Εκτύπωσε ή Αποτελέσματα ενώ στο πρόγραμμα την εντολή ΓΡΑΨΕ.
- Στο πρόγραμμα δεν υπάρχει η εντολή εισόδου Δεδομένα.
- Στο πρόγραμμα δηλώνουμε το μέγεθος των πινάκων.
- Στο πρόγραμμα δεν υπάρχει η εντολή αντιμετάθεσε.
- Στον αλγόριθμο η εκχώρηση γίνεται με το σύμβολο <-- ενώ στο πρόγραμμα με το <-
- Στο πρόγραμμα οι χαρακτήρες περικλείονται υποχρεωτικά σε 'απλά' εισαγωγικά, ενώ στον αλγόριθμο σε "διπλά" (στις εξετάσεις είναι δεκτά και τα 'απλά')
- Στο πρόγραμμα δηλώνουμε τις μεταβλητές και μπορούμε να δηλώνουμε σταθερές, δηλαδή προκαθορισμένες τιμές που δεν μεταβάλλονται κατά τη διάρκεια εκτέλεσης του προγράμματος και οι οποίες μπορεί να είναι ακέραιες ή πραγματικές ή χαρακτήρες ή λογικές.

Ποιο είναι το αλφάβητο της ΓΛΩΣΣΑΣ

- Κεφαλαία ελληνικού αλφαβήτου (Α-Ω)
- Πεζά ελληνικού αλφαβήτου (α-ω)
- Κεφαλαία λατινικού αλφαβήτου (A-Z)
- Πεζά λατινικού αλφαβήτου (a-z)
- 0-9
- + - * / = ^ ( ) . , ‘ ! & κενός χαρακτήρας

Ποιες είναι οι ενσωματωμένες συναρτήσεις της ΓΛΩΣΣΑΣ

- ΗΜ(Χ) Υπολογισμός ημιτόνου
- ΣΥΝ(Χ) Υπολογισμός συνημιτόνου
- ΕΦ(Χ) Υπολογισμός εφαπτομένης
- Τ_Ρ(Χ) Υπολογισμός τετραγωνικής ρίζας
- ΛΟΓ(Χ) Υπολογισμός φυσικού λογαρίθμου
- Ε(Χ) Υπολογισμός του ex
- Α_Μ(Χ) Ακέραιο μέρος του Χ
- Α_Τ(Χ) Απόλυτη τιμή του Χ

Κεφάλαιο 9

Τι είναι πίνακας

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.

Τι είναι ο δείκτης πίνακα

Ο δείκτης είναι μία μεταβλητή που μπορεί να έχει οποιοδήποτε δεκτό όνομα.

Ποιοι πίνακες ονομάζονται μονοδιάστατοι

Οι πίνακες που χρησιμοποιούν ένα μόνο δείκτη για την αναφορά των στοιχείων τους, ονομάζονται **μονοδιάστατοι** πίνακες.

Ποια δομή επανάληψης είναι καλύτερη για την υλοποίηση λειτουργιών σε πίνακα

Η ανάγνωση, η επεξεργασία και η εκτύπωση των στοιχείων των πινάκων γίνεται πάντοτε από βρόχους,οι οποίοι επαναλαμβάνονται προκαθορισμένο αριθμό φορών, όσα είναι τα στοιχεία του πίνακα και υλοποιούνται καλύτερα στον προγραμματισμό με την εντολή επανάληψης ΓΙΑ.

Μειονεκτήματα από τη χρήση πινάκων

Οι πίνακες απαιτούν μνήμη. Κάθε πίνακας δεσμεύει από την αρχή του προγράμματος πολλές θέσεις μνήμης. Σε ένα μεγάλο και σύνθετο πρόγραμμα η άσκοπη χρήση μεγάλων πινάκων μπορεί να οδηγήσει ακόμη και σε αδυναμία εκτέλεσης του προγράμματος.

Οι πίνακες περιορίζουν τις δυνατότητες του προγράμματος. Αυτό γιατί οι πίνακες είναι στατικές δομές και το μέγεθος τους πρέπει να δηλώνεται στην αρχή του προγράμματος, ενώ παραμένει υποχρεωτικά σταθερό κατά την εκτέλεση του προγράμματος.

Πότε πρέπει να χρησιμοποιούμε πίνακες

Η απόφαση για την χρήση ή όχι πίνακα για την διαχείριση των δεδομένων είναι κυρίως θέμα εμπειρίας στον προγραμματισμό.
Γενικά, αν τα δεδομένα που εισάγονται σε ένα πρόγραμμα πρέπει να διατηρούνται στη μνήμη μέχρι το τέλος της εκτέλεσης, τότε η χρήση πινάκων βοηθάει ή συχνά είναι απαραίτητη για την επίλυση του προβλήματος.
Σε άλλη περίπτωση μπορεί να αποφεύγεται η χρήση τους.

Πως υλοποιούνται οι λειτουργίες σε πολυδιάστατους πίνακες

Η ανάγνωση, η επεξεργασία καθώς και η εκτύπωση των στοιχείων πολυδιάστατων πινάκων γίνεται πάντοτε από βρόχους,οι οποίοι υλοποιούνται στον προγραμματισμό με εμφωλευμένες εντολές επανάληψης ΓΙΑ.

Ποιες είναι οι τυπικές επεξεργασίες πινάκων

Τα προγράμματα τα οποία χρησιμοποιούν πίνακες πολύ συχνά απαιτούν συγκεκριμένες επεξεργασίες στα στοιχεία του πίνακα. Οι τυπικές αυτές επεξεργασίες είναι:

- Υπολογισμός αθροισμάτων στοιχείων του πίνακα.
- Εύρεση του μέγιστου ή του ελάχιστου στοιχείου.
- Ταξινόμηση των στοιχείων του πίνακα.
- Αναζήτηση ενός στοιχείου του πίνακα.
- Συγχώνευση δύο πινάκων.

Ποιοι είναι οι πλέον διαδεδομένοι αλγόριθμοι αναζήτησης

Δύο είναι οι πλέον διαδεδομένοι αλγόριθμοι αναζήτησης:

- Η σειριακή αναζήτηση
- Η δυαδική αναζήτηση

Η σειριακή μέθοδος αναζήτησης είναι η πιο απλή, αλλά και η λιγότερη αποτελεσματική μέθοδος. Χρησιμοποιείται όμως υποχρεωτικά για πίνακες που δεν είναι ταξινομημένοι. Αντίθετα η δυαδική αναζήτηση χρησιμοποιείται μόνο σε ταξινομημένους πίνακες και είναι σαφώς αποδοτικότερη από τη σειριακή μέθοδο.

Τι είναι η συγχώνευση δύο πινάκων

Η συγχώνευση είναι μία από τις βασικές λειτουργίες σε πίνακες. Σκοπός της είναι η δημιουργία από τα στοιχεία δύο (ή περισσότερων) ταξινομημένων πινάκων ενός άλλου, που είναι και αυτός ταξινομημένος.

Κεφάλαιο 10

Τι ονομάζεται τμηματικός προγραμματισμός

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

Τι είναι το υποπρόγραμμα

Υποπρόγραμμα είναι ένα τμήμα προγράμματος που επιτελεί ένα αυτόνομο έργο και έχει γραφεί χωριστά από το υπόλοιπο πρόγραμμα.

Ποιες ιδιότητες διακρίνουν τα υποπρογράμματα

- **Κάθε υποπρόγραμμα έχει μόνο μία είσοδο και μία έξοδο**. Στην πραγματικότητα κάθε υποπρόγραμμα ενεργοποιείται με την είσοδο σε αυτό που γίνεται πάντοτε από την αρχή του, εκτελεί ορισμένες ενέργειες, και απενεργοποιείται με την έξοδο από αυτό που γίνεται πάντοτε από το τέλος του.
- **Κάθε υποπρόγραμμα πρέπει να είναι ανεξάρτητο από τα άλλα**. Αυτό σημαίνει ότι κάθε υποπρόγραμμα μπορεί να σχεδιαστεί, να αναπτυχθεί και να συντηρηθεί αυτόνομα χωρίς να επηρεαστούν άλλα υποπρογράμματα. Στην πράξη βέβαια η απόλυτη ανεξαρτησία είναι δύσκολο να επιτευχθεί.
- **Κάθε υποπρόγραμμα πρέπει να μην είναι πολύ μεγάλο**. Η έννοια του μεγάλου προγράμματος είναι υποκειμενική, αλλά πρέπει κάθε υποπρόγραμμα να είναι τόσο, ώστε να είναι εύκολα κατανοητό για να μπορεί να ελέγχεται. Γενικά κάθε υποπρόγραμμα πρέπει να εκτελεί μόνο μία λειτουργία. Αν εκτελεί περισσότερες λειτουργίες, τότε συνήθως μπορεί και πρέπει να διασπαστεί σε ακόμη μικρότερα υποπρογράμματα.

Ποια είναι τα πλεονεκτήματα του τμηματικού προγραμματισμού

- Διευκολύνει την ανάπτυξη του αλγορίθμου και του αντιστοίχου προγράμματος.
- Διευκολύνει την κατανόηση και διόρθωση του προγράμματος.
- Απαιτεί λιγότερο χρόνο και προσπάθεια στη συγγραφή του προγράμματος.
- Επεκτείνει τις δυνατότητες των γλωσσών προγραμματισμού.

Τι είναι παράμετρος

Μία παράμετρος είναι μία μεταβλητή που επιτρέπει το πέρασμα της τιμής της από ένα τμήμα προγράμματος σε ένα άλλο.

Τι είναι συνάρτηση

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

Τι είναι διαδικασία

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

Ποιος είναι ο τρόπος κλήσης των υποπρογραμμάτων

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.

Σε ποια θέση του προγράμματος τοποθετούνται τα υποπρογράμματα

Τόσο οι συναρτήσεις όσο και οι διαδικασίες τοποθετούνται μετά το τέλος του κυρίου προγράμματος.

Ποια είναι η δομή μίας συνάρτησης

Κάθε συνάρτηση έχει την ακόλουθη δομή.

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

Ποιος μπορεί να είναι ο τύπος μίας συνάρτησης

Οι συναρτήσεις μπορούν να επιστρέφουν τιμές όλων των τύπων δεδομένων που υποστηρίζει η γλώσσα. Μια συνάρτηση λοιπόν μπορεί να είναι ΠΡΑΓΜΑΤΙΚΗ, ΑΚΕΡΑΙΑ, ΧΑΡΑΚΤΗΡΑΣ, ΛΟΓΙΚΗ.

Ποια εντολή πρέπει οπωσδήποτε να περιέχει μία συνάρτηση

Στις εντολές του σώματος της συνάρτησης πρέπει υποχρεωτικά να υπάρχει μία εντολή εκχώρησης τιμής στο όνομα της συνάρτησης.

Πόσες παραμέτρους μπορεί να έχει μία συνάρτηση

Μία συνάρτηση πρέπει να έχει στη λίστα της τουλάχιστον μία παράμετρο, η οποία χρησιμοποιείται για να περάσουν τιμές στη συνάρτηση.

Καμία παράμετρος δεν επιστρέφει τιμές στο τμήμα προγράμματος που ενεργοποίησε (κάλεσε) τη συνάρτηση.

Πόσες τιμές επιστρέφει μία συνάρτηση και πως

Η συνάρτηση επιστρέφει μία και μόνο μία τιμή με το όνομά της. Δηλαδή το όνομα της συνάρτησης είναι και αυτό παράμετρος.

Ποια είναι η δομή μίας διαδικασίας

ΔΙΑΔΙΚΑΣΙΑ Όνομα (λίστα παραμέτρων)
Τμήμα δηλώσεων
ΑΡΧΗ
εντολές
ΤΕΛΟΣ_ΔΙΑΔΙΚΑΣΙΑΣ

Πόσες παραμέτρους μπορεί να έχει μία διαδικασία

Στη λίστα παραμέτρων μίας διαδικασίας, μπορούν να υπάρχουν καμία, μία ή περισσότερες παράμετροι. Όταν υπάρχουν πολλές παράμετροι, τότε άλλες χρησιμοποιούνται για να μεταβιβάσουν τιμές στη διαδικασία και άλλες για να επιστρέψουν τιμές στο κύριο πρόγραμμα.

Ποια είναι η γενική μορφή της εντολής ΚΑΛΕΣΕ και πως λειτουργεί

ΚΑΛΕΣΕ όνομα διαδικασίας (λίστα παραμέτρων)

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.

Ποιες παράμετροι ονομάζονται πραγματικές και ποιες τυπικές

Οι **πραγματικές παράμετροι** είναι μεταβλητές του προγράμματος (κύριου ή υποπρογράμματος) το οποίο καλεί ένα υποπρόγραμμα (δηλαδή έχουν δηλωθεί στο πρόγραμμα που κάνει την κλήση).
Η λίστα των πραγματικών παραμέτρων καθορίζει τις παραμέτρους στην κλήση του υποπρογράμματος.

Οι **τυπικές παράμετροι** είναι μεταβλητές του υποπρογράμματος το οποίο καλείται (δηλαδή έχουν δηλωθεί στο υποπρόγραμμα που δέχεται την κλήση).
Η λίστα των τυπικών παραμέτρων καθορίζει τις παραμέτρους στη δήλωση του υποπρογράμματος.

Ποια είναι η ισχύς των μεταβλητών, άρα και των παραμέτρων

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.

Πως αλλιώς ονομάζονται οι παράμετροι

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

Κανόνες για τις λίστες παραμέτρων

- 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.

Ποιες οι διαφορές μεταξύ διαδικασίας και συνάρτησης

- Η συνάρτηση ενεργοποιείται με απλή αναφορά του ονόματός της μέσα σε μία αλγοριθμική έκφραση, ενώ η διαδικασία με τη χρήση της δεσμευμένης λέξη ΚΑΛΕΣΕ.
- Η συνάρτηση πρέπει να έχει τουλάχιστον μία παράμετρο (είσοδος), ενώ η διαδικασία μπορεί και να μην έχει παραμέτρους π.χ. ΚΑΛΕΣΕ Διαδικασία( ).
- Η συνάρτηση υπολογίζει και επιστρέφει (έξοδος) μία (και μόνο) τιμή με το όνομά της, ενώ η διαδικασία μπορεί να επιστρέψει από καμία έως και πολλές τιμές (ακόμα και ολόκληρο πίνακα), μέσω των παραμέτρων της.
- Η μεταβολή των τιμών των παραμέτρων της συνάρτησης (τυπικές), δεν αντανακλά στις αντίστοιχες παραμέτρους του προγράμματος που την κάλεσε (πραγματικές), ενώ στη διαδικασία όλες οι μεταβολές μεταφέρονται και στο πρόγραμμα που την κάλεσε (υπάρχει αλληλεπίδραση).
- Στη συνάρτηση πρέπει να δηλώνεται στην αρχή, αμέσως μετά το όνομα της συνάρτησης, ο τύπος της τιμής που επιστρέφει (ΑΚΕΡΑΙΑ, ΠΡΑΓΜΑΤΙΚΗ, ΧΑΡΑΚΤΗΡΑΣ, ΛΟΓΙΚΗ).
- Η συνάρτηση δεν εκτελεί τις εντολές ΔΙΑΒΑΣΕ και ΓΡΑΨΕ, ενώ η διαδικασία εκτελεί όσες εντολές εκτελεί και ένα κανονικό πρόγραμμα.

Τι ονομάζεται στοίβα χρόνου εκτέλεσης

Όταν μία διαδικασία ή συνάρτηση καλείται από το κύριο πρόγραμμα, τότε η αμέσως επόμενη διεύθυνση του κύριου προγράμματος, που ονομάζεται διεύθυνση επιστροφής, αποθηκεύεται από το μεταφραστή σε μία στοίβα που ονομάζεται στοίβα χρόνου εκτέλεσης.