

Κεφάλαιο

1

Εισαγωγή



Εισαγωγή

Οκτώβριος του 1983. Ως μεταπτυχιακός φοιτητής, γνώρισα μια δεκαοχτάχρονη αμερικανιδούλα που την έλεγαν BCPL. Η σχέση μας κράτησε ένα χρόνο περίπου. Ευέλικτη αλλά δύστροπη. Δύσκολο να την κατανοήσεις και δυσκολότερο να τη χειριστείς. Την επόμενη χρονιά μου γνώρισε τη μικρότερη αδελφή της, τη C. Από την πρώτη στιγμή με εντυπωσίασε, είχε πιο δομημένη σκέψη, ήταν τρομερά ευέλικτη και πολύ πιο φιλική. Η γνωριμία αυτή δεν άργησε να εξελιχθεί σε ένα μεγάλο έρωτα, ο οποίος κρατάει ακόμη και σήμερα. Από τότε συνάντησα και άλλες πολλές, πολύ πιο νέες και εμφανίσιμες, με πολλές δυνατότητες και προσόντα, έμεινα όμως πάντα πιστός στη μεσήλικα πια αγάπη μου, τη C. Ευτυχώς που είναι μόνο ... μια γλώσσα προγραμματισμού!

Και όμως, η πολύχρονη αυτή σχέση είχε και έναν καρπό, την παραγωγή πλούσιου εκπαιδευτικού υλικού που χρησιμοποιήθηκε όλα αυτά τα χρόνια για την διδασκαλία της γλώσσας C και το οποίο αποτέλεσε τη βάση για τη συγγραφή αυτού του βιβλίου.

Το βιβλίο αυτό απευθύνεται τόσο στον αρχάριο όσο και στον έμπειρο προγραμματιστή που θέλει να γνωρίσει τις αρχές και τη φιλοσοφία του δομημένου προγραμματισμού, μέσα από μια ευέλικτη και χωρίς όρια γλώσσα όπως η C.

Με ιδιαίτερο τρόπο προσεγγίζονται όλα τα χαρακτηριστικά της γλώσσας και δίνεται έμφαση στην αναλυτική και σε βάθος επεξήγηση των "στρυφνών" της σημείων. Μεγάλο βάρος έχει δοθεί στη διδακτική σειρά αυτού του βιβλίου, ώστε η ανάγνωση και η κατανόηση ενός κεφαλαίου να προϋποθέτει **μόνο** τις γνώσεις που αποκτήθηκαν στα προηγούμενα κεφάλαια.

Πώς να διαβάσετε αυτό το βιβλίο

Αν είστε γνώστης μιας οποιασδήποτε άλλης γλώσσας προγραμματισμού μπορείτε να παραλείψετε το Κεφάλαιο 1 και να προχωρήσετε κατευθείαν στο επόμενο κεφάλαιο.

Το Κεφάλαιο 2 είναι μια μικρή "περιοδεία" στη γλώσσα. Γίνεται μια πρώτη γνωριμία, ώστε να αποκτήσετε μια σφαιρική εικόνα από τη δομή και τα χαρα-

κτηριστικά της C και να μπορέσετε να καταστρώσετε τα πρώτα σας απλά προγράμματα.

Τα επόμενα κεφάλαια αναλύουν όλα τα χαρακτηριστικά της γλώσσας, δίνοντας έμφαση στην παρουσίαση των βασικών, αλλά και των περισσότερο πολύπλοκων εννοιών, με απλό, εποπτικό και κατανοητό τρόπο.

Τα τρία τελευταία κεφάλαια (16, 17, και 18) είναι μια βαθιά "βουτιά" τόσο σε εξειδικευμένες τεχνικές προγραμματισμού όσο και στα ιδιαίτερα χαρακτηριστικά της γλώσσας.

Το Παράρτημα Α περιέχει μια αναλυτική αναφορά στις πιο συχνά χρησιμοποιούμενες συναρτήσεις της C.

Στο Παράρτημα Β θα βρείτε λεπτομέρειες για το περιβάλλον ανάπτυξης του DEV C/C++. Το DEV C/C++ είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης για τις γλώσσες C και C++, το οποίο διατίθεται δωρεάν κάτω από τη γενική άδεια χρήσης της GNU¹. Όλα τα προγράμματα που υπάρχουν σε αυτό το βιβλίο έχουν δοκιμαστεί στο περιβάλλον του DEV C/C++.

Αποκλειστικά για το βιβλίο αυτό, έχει δημιουργηθεί ένας δικτυακός τόπος στη διεύθυνση <http://c.bytes.gr>. Εκεί θα βρείτε το περιβάλλον του DEV C/C++, τον κώδικα των περισσότερων προγραμμάτων που βρίσκονται σε αυτό το βιβλίο, καθώς και πλήθος αναφορών σχετικά με τη γλώσσα C και C++.

Σας συνιστώ να "κατεβάσετε" και να εγκαταστήσετε το περιβάλλον του DEV C/C++ ώστε να μπορείτε να δοκιμάζετε τα παραδείγματα του βιβλίου αλλά και τα δικά σας προγράμματα.

Η γλώσσα C — Ιστορική αναδρομή

Η C άρχισε να αναπτύσσεται στις αρχές του 1972 στα εργαστήρια Bell από τον Denis Ritchie ως γλώσσα προγραμματισμού συστημάτων, με σκοπό να δημιουργηθεί ένα νέο λειτουργικό σύστημα σε ένα μίνι υπολογιστή PDP-11. Η C είναι η εξέλιξη μιας προγενέστερης γλώσσας, της BCPL, η οποία αναπτύχθηκε από τον Martin Richards στα μέσα της δεκαετίας του 60.

¹ <http://www.gnu.org/philosophy>

Μια πρώτη γλωσσική περιγραφή της C δημοσιεύθηκε έξι χρόνια αργότερα (1978), από τους Kernighan και Ritchie. Η έκδοση αυτή θεωρείται μέχρι σήμερα το ευαγγέλιο της C.

Το 1983, το Εθνικό ίδρυμα προτύπων της Αμερικής ίδρυσε μία επιτροπή για να ετοιμάσει ένα πρότυπο για τη γλώσσα προγραμματισμού C. Αυτό το πρότυπο, το "ANSI C", ολοκληρώθηκε το 1988. Ενώ ετοιμαζόταν το πρότυπο ANSI, εξελισσόταν και η αρχική περιγραφή της γλώσσας, της οποίας η τελική έκδοση οριστικοποιήθηκε το 1988.

Η C έχει συνδέσει το όνομα της με το λειτουργικό σύστημα UNIX, πολλά από τα τμήματα του οποίου έχουν αναπτυχθεί στη C. Η "στάνταρ" έκδοση της C ήταν, για πολλά χρόνια, εκείνη η οποία συνόδευε το λειτουργικό σύστημα UNIX (Ver. 5), του οποίου θεωρείται και αναπόσπαστο μέρος.

Τα χαρακτηριστικά της C

Παρά το γεγονός ότι η C θεωρείται μια γλώσσα υψηλού επιπέδου (high-level), έχει αρκετά χαρακτηριστικά που συναντώνται μόνο σε γλώσσες χαμηλού επιπέδου και στη γλώσσα μηχανής. Τα χαρακτηριστικά αυτά της προσδίδουν εκπληκτική ευελιξία και τη δυνατότητα χειρισμών "χαμηλού επιπέδου".

Η ευελιξία της C, αλλά και η έλλειψη αυστηρού ελέγχου, είναι ένα γερό εργαλείο στα χέρια ενός έμπειρου προγραμματιστή αλλά μεγάλη ταλαιπωρία και βάσανο για τον αρχάριο.

Η C είναι μία "λιτή" γλώσσα. Όλες κι όλες οι δεσμευμένες λέξεις της C δεν ξεπερνούν τις 30. Αξιοσημείωτο είναι ότι η C δεν έχει ενσωματωμένες εντολές εισόδου & εξόδου, όπως και αρκετές άλλες εντολές που συναντώνται σε άλλες γλώσσες προγραμματισμού. Για το λόγο αυτόν η C περιλαμβάνει στην τυπική ("στάνταρ") εγκατάστασή της "βιβλιοθήκες" με κάθε είδους συναρτήσεις για είσοδο-έξοδο, χειρισμό χαρακτήρων, χειρισμό αρχείων, γραφικών κ.λπ.

C — Μια δομημένη γλώσσα

Το διακριτικό χαρακτηριστικό μιας δομημένης γλώσσας προγραμματισμού είναι η δυνατότητα για τμηματικό χειρισμό (modularity) του προγράμματος, με τρόπο ώστε κάθε τμήμα να μπορεί να "κρύβει" τον κώδικα και τις πληροφορίες

που περιέχει από το υπόλοιπο πρόγραμμα. Η C ενθαρρύνει τη χρήση ξεχωριστών συναρτήσεων (υποπρογραμμάτων) για κάθε συγκεκριμένη λειτουργία του προγράμματος. Κάθε συνάρτηση (function) μπορεί να έχει τις δικές της τοπικές μεταβλητές (local variables), οι οποίες είναι "κρυμμένες" από το υπόλοιπο πρόγραμμα.

C — Μια γλώσσα για προγραμματιστές

Η C παρέχει στον πραγματικό επαγγελματία προγραμματιστή αυτό που ακριβώς ζητάει:

- Λίγους περιορισμούς και μεγάλη ευελιξία
- Δυνατότητα για δομημένα προγράμματα
- Λίγες αλλά ισχυρά δομημένες εντολές

Η C έχει όμως και απαιτήσεις. Η έλλειψη περιορισμών και το γεγονός ότι υπάρχει μικρός βαθμός ελέγχου λαθών, αναγκάζουν τον προγραμματιστή να είναι πολύ προσεκτικός και να ελέγχει, μέσα από τον κώδικα του προγράμματος, πράγματα που ελέγχονται αυτόματα από άλλες γλώσσες προγραμματισμού. Ένα χαρακτηριστικό παράδειγμα είναι ότι αν έχουμε ένα πίνακα 100 θέσεων μπορούμε κάλλιστα να προσπελάσουμε την θέση 105!!! Τώρα το τι θα βρούμε εκεί μέσα και το τι παρενέργειες θα υπάρξουν, αφήστε το για αργότερα.

C — Μια μεταφραστική γλώσσα

Όλες οι γλώσσες προγραμματισμού (εκτός από τη γλώσσα μηχανής), και ανάλογα με τη φιλοσοφία με την οποία μεταφράζουν το πηγαίο πρόγραμμα σε γλώσσα μηχανής, παρουσιάζονται είτε σε ερμηνευτική (interpreter's), είτε σε μεταγλωττιζόμενη (compiler's) μορφή.

Η BASIC π.χ. είναι μια γλώσσα που συχνά τη συναντούμε σε ερμηνευτική μορφή. Αντίθετα η C είναι σχεδόν πάντα σε μεταγλωττιζόμενη μορφή. Στο σχήμα που ακολουθεί βλέπετε μια απλοποιημένη διαδικασία μεταγλωττισμού μέσω μεταγλωττιστή (compiler).

Το αρχείο που περιέχει τον πηγαίο κώδικα του προγράμματος (source code) μπορεί να δημιουργηθεί με οποιονδήποτε επεξεργαστή κειμένου ή μέσα από το ολοκληρωμένο περιβάλλον της γλώσσας (αν η έκδοση της γλώσσας διαθέτει τέτοιο περιβάλλον).

Κατά τη διάρκεια της μεταγλώττισης (compile time), ο μεταγλωττιστής εντοπίζει τυχόν συντακτικά λάθη που υπάρχουν στον πηγαίο κώδικα. Εφόσον ο μεταγλωττιστής δεν εντοπίσει κανένα λάθος, θα δημιουργήσει ένα αρχείο που περιέχει τον εκτελέσιμο κώδικα (executable code), δηλαδή πρόγραμμα σε γλώσσα μηχανής, άμεσα εκτελέσιμο από τον Η/Υ.

Στη πραγματικότητα, η διαδικασία της μεταγλώττισης είναι λίγο διαφορετική, ιδίως όταν γίνεται χρήση βιβλιοθηκών στην οποία θα αναφερθούμε αναλυτικά στο αντίστοιχο κεφάλαιο.



Βασικές έννοιες του προγραμματισμού

Το τμήμα αυτό είναι προσανατολισμένο στους αρχάριους στον προγραμματισμό. Αναφέρει και επεξηγεί βασικές έννοιες του προγραμματισμού, απαραίτητες για τον αναγνώστη που έρχεται πρώτη φορά σε επαφή με μία γλώσσα προγραμματισμού.

Κάθε πρόγραμμα, ανεπτυγμένο σε οποιαδήποτε γλώσσα προγραμματισμού, επεξεργάζεται δεδομένα και δίνει αποτελέσματα. Η επεξεργασία των δεδομένων γίνεται πάντα με έναν προκαθορισμένο τρόπο (αλγόριθμο).

Τα συστατικά που συνθέτουν ένα πρόγραμμα είναι οι τύποι δεδομένων, οι μεταβλητές, οι σταθερές, και οι εντολές.

Τύποι δεδομένων

Οι τύποι δεδομένων ορίζουν τα διαφορετικά είδη δεδομένων που μπορεί να χειριστεί μία γλώσσα προγραμματισμού. Συνήθεις τύποι δεδομένων είναι:

- **Ο ακέραιος τύπος δεδομένων:** Αναφέρεται σε δεδομένα που είναι ακέραιοι αριθμοί (στη C ο τύπος αυτός συμβολίζεται με `int`).

- **Ο πραγματικός τύπος δεδομένων:** Αναφέρεται σε δεδομένα που είναι αριθμοί κινητής υποδιαστολής με μικρή ή μεγάλη ακρίβεια (στη C ο τύπος αυτός συμβολίζεται, αντίστοιχα, με `float` ή `double`).
- **Ο τύπος δεδομένων χαρακτήρα:** Αναφέρεται σε μεμονωμένους χαρακτήρες (στη C ο τύπος αυτός συμβολίζεται με `char`).
- **Ο τύπος δεδομένων συνόλου χαρακτήρων:** Αναφέρεται σε σύνολα χαρακτήρων (η C δεν υποστηρίζει άμεσα αυτό τον τύπο δεδομένων και χειρίζεται διαφορετικά τα σύνολα χαρακτήρων).
- **Ο λογικός τύπος δεδομένων:** Αναφέρεται σε δεδομένα με τιμή αλήθεια ή ψέμα (η C δεν διαθέτει ξεχωριστό τύπο για λογικά δεδομένα αλλά τα χειρίζεται σαν αριθμούς).

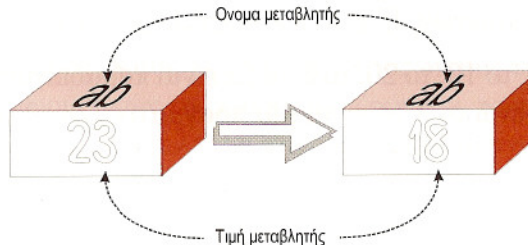
Μεταβλητές

Τα δεδομένα, οι ενδιάμεσες πράξεις, αλλά και τα αποτελέσματα ενός προγράμματος αποθηκεύονται στην κεντρική μνήμη (RAM) του Η/Υ. Η διαχείριση της μνήμης είναι βασικό μέλημα μιας γλώσσας προγραμματισμού.

Για να χρησιμοποιηθεί μία θέση μνήμης μέσα από μία γλώσσα προγραμματισμού πρέπει να της δοθεί ένα όνομα.

Σε κάθε θέση μνήμης που χρησιμοποιείται σε ένα πρόγραμμα, ανατίθεται (αντιστοιχίζεται, αποδίδεται) και ένα όνομα, το δε περιεχόμενο της θέσης μνήμης αποτελεί την τιμή της.

Φανταστείτε την κάθε θέση μνήμης που χρησιμοποιούμε σαν ένα κουτί, στο εξωτερικό του οποίου είναι γραμμένο το όνομα του. Το περιεχόμενο του κουτιού, αποτελεί την τιμή αυτής της θέσης μνήμης.



Ας θεωρήσουμε τη θέση μνήμης **ab** η οποία περιέχει στην αρχή τον αριθμό 23 και με κατάλληλη εντολή αλλάζουμε το περιεχόμενο της σε 18.

Παρατηρούμε ότι το κουτί (θέση μνήμης) παραμένει το ίδιο (ab) αλλά το περιεχόμενο του μεταβάλλεται. Ένα όνομα λοιπόν που αντιστοιχεί σε μία θέση μνήμης μπορεί να πάρει διαφορετικές τιμές, για αυτόν τον λόγο κάθε όνομα θέσης μνήμης καλείται μεταβλητή (variable).

Στο παραπάνω παράδειγμα η μεταβλητή **ab** είχε στην αρχή την τιμή 23 και μετά έχει την τιμή 18.

Μεταβλητή είναι μία θέση μνήμης, την οποία έχουμε δεσμεύσει και της έχουμε δώσει ένα όνομα. Το περιεχόμενο αυτής της θέσης μνήμης αποτελεί την τιμή αυτής της μεταβλητής.

Ανάλογα με τους τύπους δεδομένων που υποστηρίζει μια γλώσσα προγραμματισμού έχουμε και τα αντίστοιχα είδη (τύπους) μεταβλητών. Έτσι έχουμε ακέραιες μεταβλητές για αποθήκευση ακέραιων αριθμών, μεταβλητές χαρακτήρων για την αποθήκευση χαρακτήρων κ.λπ.

Στις περισσότερες γλώσσες προγραμματισμού, πριν χρησιμοποιήσουμε μία μεταβλητή πρέπει να δηλώσουμε τον τύπο της.

Η C υποστηρίζει διάφορους τύπους μεταβλητών, οι βασικοί από τους οποίους είναι τρεις και αναφέρονται αναλυτικά στο επόμενο κεφάλαιο.

Σταθερές

Οι σταθερές είναι προκαθορισμένες τιμές που δεν μεταβάλλονται κατά τη διάρκεια εκτέλεσης ενός προγράμματος. Υπάρχουν αντίστοιχες σταθερές για κάθε τύπο δεδομένων. Π.χ. ακέραια σταθερά είναι ένας ακέραιος αριθμός.

Αριθμοί όπως το 10, το 20, το 17 κ.λπ είναι παραδείγματα ακέραιων σταθερών (τύπου **int**), ενώ το 3.14 είναι μία πραγματική σταθερά (τύπου **float**).

Εντολές

Εντολή είναι μία **φράση** της γλώσσας προγραμματισμού που αναγκάζει τον Η/Υ να εκτελέσει μία συγκεκριμένη λειτουργία.

Κάθε εντολή μιας γλώσσας προγραμματισμού έχει μια αυστηρά καθορισμένη σύνταξη, η οποία ορίζεται από την ίδια τη γλώσσα.

Οι εντολές επιτελούν διάφορες λειτουργίες. Μια εντολή μπορεί να κάνει μια απλή πράξη ή έναν έλεγχο, να επιβάλει μια επαναληπτική διαδικασία, να εμφανίσει κάτι στην οθόνη, να γράψει κάτι στο δίσκο του Η/Υ κ.λπ. Σε κάθε περίπτωση, ένα πρόγραμμα αποτελείται από έναν αριθμό εντολών που εκτελούνται η μία μετά την άλλη (εκτός αν κάποια εντολή ορίσει μία διαφορετική σειρά εκτέλεσης).

Το πρόγραμμα σε μια γλώσσα προγραμματισμού μπορεί να είναι ενιαίο ή να αποτελείται από περισσότερα τμήματα (υποπρογράμματα).

Οι περισσότερες γλώσσες ενθαρρύνουν τον προγραμματιστή να χωρίζει το πρόγραμμα σε μικρότερα τμήματα, τα οποία ονομάζονται υποπρογράμματα. Η C είναι μια γλώσσα, η οποία με τη δομή της σχεδόν επιβάλλει τον τμηματικό προγραμματισμό. Ένα τμήμα προγράμματος (υποπρόγραμμα) στη C ονομάζεται συνάρτηση (function). Οι συναρτήσεις καλούνται χρησιμοποιώντας απλώς το όνομά τους. Μια συνάρτηση μπορεί απλώς να εκτελεί μια λειτουργία ή και να επιστρέφει κάποια τιμή στο πρόγραμμα που την κάλεσε.

Αναγνωριστικά

Αναγνωριστικά (identifiers) καλούνται τα ονόματα που χρησιμοποιεί μια γλώσσα για να αναφέρεται στα επί μέρους στοιχεία της. Για παράδειγμα, το όνομα μιας μεταβλητής ή το όνομα ενός υποπρογράμματος αποτελούν αναγνωριστικά της γλώσσας. Κάθε γλώσσα έχει τους δικούς της κανόνες για την σύνταξη των αναγνωριστικών της και καθορίζουν π.χ. το πλήθος και το είδος των επιτρεπτών χαρακτήρων κ.λπ. Οι κανόνες για τα αναγνωριστικά της C αναφέρονται στο επόμενο κεφάλαιο.

Λογικό διάγραμμα


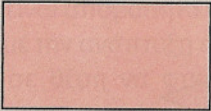
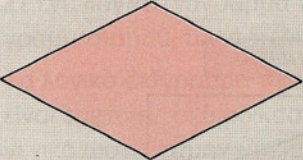
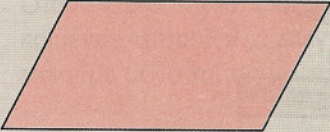

Είναι προφανές ότι το γράψιμο ενός προγράμματος προϋποθέτει τη γνώση του αντικειμένου στο οποίο θα αναφέρεται το συγκεκριμένο πρόγραμμα. Για παράδειγμα, αν θέλουμε να φτιάξουμε ένα πρόγραμμα το οποίο να απευθύνεται σε οδοντίατρους, θα πρέπει καταρχήν να γνωρίζουμε τον τρόπο με τον οποίο δουλεύει ένας οδοντίατρος, τι απαιτήσεις έχει, τι πληροφορίες θέλει να κρατάει για τους ασθενείς του κ.λπ. Αυτό το στάδιο, κατά το οποίο εμείς ουσιαστικά "μαθαίνουμε" τον τρόπο δουλειάς του οδοντίατρου λέγεται **ανάλυση**.

Μετά το στάδιο της ανάλυσης ακολουθεί το στάδιο του σχεδιασμού. Στο στάδιο αυτό μπορούμε, αν αυτό κριθεί απαραίτητο, να προτείνουμε νέους καλύτερους τρόπους εργασίας ή ακόμη και νέες δυνατότητες. Η ανάλυση και ο σχεδιασμός δεν έχουν απαραίτητα άμεση σχέση με τον προγραμματισμό αλλά χρησιμοποιούνται, σαν αναγκαία εργαλεία, για τη σωστή και επαγγελματική κατάσχεση προγραμμάτων.

Στα στάδια της ανάλυσης και του σχεδιασμού, ο αναλυτής χρησιμοποιεί εκτός των άλλων "εργαλείων" του, γεωμετρικά σχήματα για να δώσει μια εικόνα της λειτουργίας του υπό μελέτη συστήματος. Η σχηματική αυτή απεικόνιση ενός συστήματος καλείται λογικό διάγραμμα ή διάγραμμα ροής (flow chart). Σε αυτό το βιβλίο χρησιμοποιείται συχνά το λογικό διάγραμμα για να επεξηγηθεί η λειτουργία μιας διαδικασίας ή ενός προγράμματος.

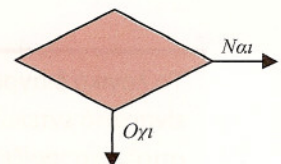
Τα γεωμετρικά σχήματα που χρησιμοποιούνται σε ένα λογικό διάγραμμα, είναι καθορισμένα και το κάθε ένα δείχνει μια συγκεκριμένη λειτουργία. Κάθε σχήμα αντιστοιχεί σε μία κατηγορία διαδικασίας του "υπό μελέτη" συστήματος. Για παράδειγμα, ο ρόμβος δείχνει **έλεγχο** ενώ το ορθογώνιο παραλληλόγραμμο **επεξεργασία**.

Στον πίνακα που ακολουθεί, αναφέρονται τα σχήματα που συνήθως χρησιμοποιούνται σε ένα λογικό διάγραμμα.

Γεωμετρικό σχήμα	Λειτουργία
	Αρχή/Τέλος διαδικασίας
	Επεξεργασία
	Έλεγχος
	Είσοδος/Εξοδος πληροφοριών
	Μεταφορά ελέγχου

Μέσα σε κάθε γεωμετρικό σχήμα περιγράφεται με απλά λόγια η διαδικασία που εκτελείται.

Στη διαδικασία του ελέγχου είναι δυνατές μόνο δύο αποφάσεις: **Ναι** ή **Όχι**. Ανάλογα με το αποτέλεσμα του ελέγχου, το διάγραμμα ροής μας οδηγεί σε διαφορετικό παρακλάδι του.



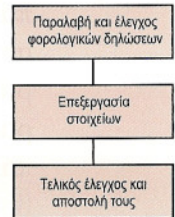
Το επόμενο λογικό διάγραμμα δείχνει τη διαδικασία που ακολουθείται όταν κάποιος καλεί το ασανσέρ μέχρι να πάει στον όροφο

που τον ενδιαφέρει. Αναμφίβολα υπάρχουν πολλές εκδοχές για το σύστημα αυτό. Το λογικό διάγραμμα που ακολουθεί δείχνει εποπτικά τη λειτουργία της συγκεκριμένης διαδικασίας.



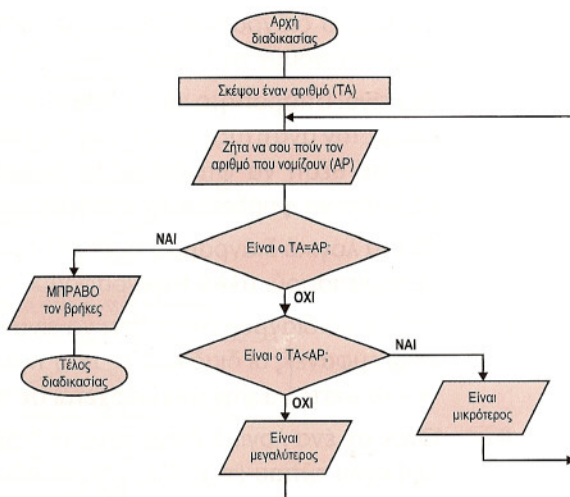
Το λογικό διάγραμμα ενός προβλήματος δεν καθορίζεται μονοσήμαντα. Μπορεί να είναι από εντελώς επιγραμματικό έως πολύ αναλυτικό, ανάλογα με τη χρήση για την οποία προορίζεται.

- ☞ Το λογικό διάγραμμα είναι ανεξάρτητο από τον προγραμματισμό. Χρησιμοποιείται συχνά για να μπορέσουμε να απεικονίσουμε τη λογική ενός προγράμματος πριν αρχίσουμε να το αναπτύσσουμε.
- ☞ Το λογικό διάγραμμα βοηθάει να κατανοήσουμε τον τρόπο με τον οποίο θα προσεγγίσουμε τη λύση.
- ☞ Δεν είναι απαραίτητο να φτιάχνουμε οπωσδήποτε λογικό διάγραμμα πριν αρχίσουμε την ανάπτυξη ενός προγράμματος. Το βέβαιο είναι όμως ότι, αν δεν είμαστε σε θέση να φτιάξουμε το λογικό διάγραμμα ενός προγράμματος, είναι αδύνατο να φτιάξουμε το πρόγραμμα και να δουλεύει σωστά.
- ☞ Σε ένα λογικό διάγραμμα ποτέ δεν αναφέρουμε εντολές κάποιας γλώσσας προγραμματισμού, αλλά περιγράφουμε τις διαδικασίες με απλά λόγια.
- ☞ Το λογικό διάγραμμα πρέπει να είναι τόσο αναλυτικό ώστε να είναι εμφανείς οι διαδικασίες που πρέπει να δείξουμε και τόσο γενικό ώστε να μην υπεισέρχεται σε άχρηστες λεπτομέρειες.
- ☞ Όταν σε ένα λογικό διάγραμμα οι διαδικασίες δεν είναι συγκεκριμένου τύπου (π.χ. έλεγχος, είσοδος/έξοδος) τότε χρησιμοποιούμε μόνο το σχήμα της επεξεργασίας (ορθογώνιο) και το

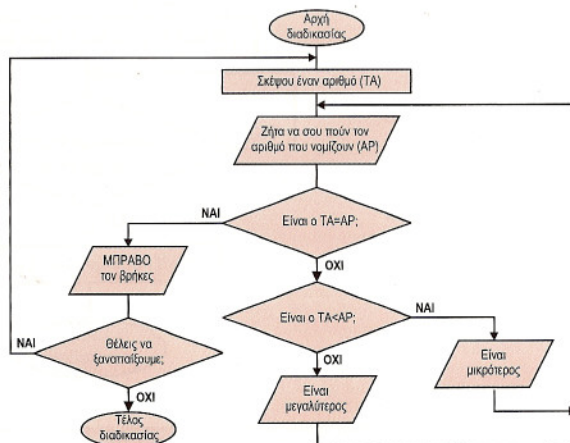


Παραδείγματα

Π.1 Ας υποθέσουμε ότι θέλουμε να παίξουμε ένα παιχνίδι στο οποίο κάποιος βάζει έναν τυχαίο αριθμό στο μυαλό του και εμείς προσπαθούμε να τον βρούμε. Λέμε συνέχεια αριθμούς στον παίκτη και αυτός μας απαντά αν ο κρυφός αριθμός είναι μεγαλύτερος ή μικρότερος. Το παιχνίδι τελειώνει μόλις βρούμε τον αριθμό. Το λογικό διάγραμμα απεικονίζει τη διαδικασία του παιχνιδιού



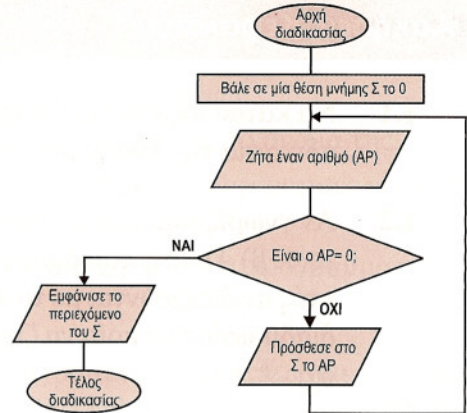
Μια μικρή παραλλαγή του παιχνιδιού, στην οποία μόλις βρεθεί ο αριθμός ο παίκτης μας ρωτάει αν θέλουμε να ξαναπαίξουμε, φαίνεται στο διπλανό λογικό διάγραμμα.



Στην περίπτωση που απαντήσουμε ΝΑΙ, τότε ο παίκτης βάζει έναν άλλο αριθμό στο μυαλό του και το παιχνίδι ξαναρχίζει. Στην περίπτωση που απαντήσουμε αρνητικά το παιχνίδι σταματάει.

Π.2 Η διαδικασία που απεικονίζει το διπλανό λογικό διάγραμμα, ζητάει συνέχεια αριθμούς και εμφανίζει στο τέλος το συνολικό τους άθροισμα. Η διαδικασία σταματάει όταν δοθεί ο αριθμός 0.

Σκεφτείτε τις τροποποιήσεις που θα πρέπει να γίνουν ώστε η διαδικασία να σταματάει όταν δοθούν 100 αριθμοί.

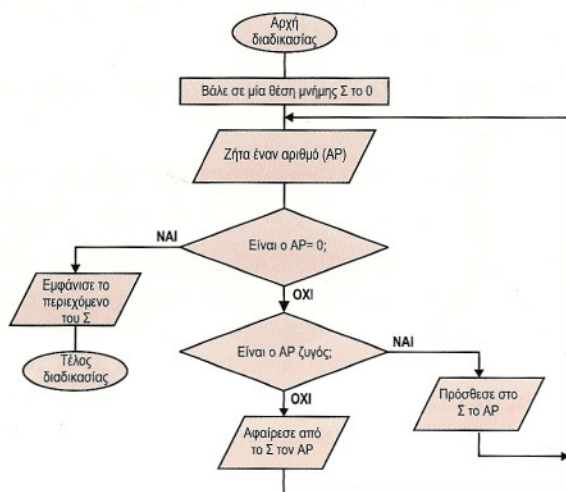


Ανασκόπηση Κεφαλαίου 1

- Η C είναι μια μεταγλωττιζόμενη γλώσσα τα χαρακτηριστικά της οποίας ευνοούν το δομημένο και τμηματικό προγραμματισμό.
- Η C είναι μια "λιτή" γλώσσα, η οποία για τις περισσότερες λειτουργίες της βασίζεται στις συναρτήσεις βιβλιοθήκης που τη συνοδεύουν.
- Τα διαφορετικά είδη δεδομένων που μπορεί να χειριστεί μια γλώσσα λέγονται τύποι δεδομένων.
- Μια μεταβλητή είναι το όνομα μιας θέσης μνήμης, της οποίας η τιμή μπορεί να μεταβάλλεται.
- Το λογικό διάγραμμα είναι ένας εποπτικός τρόπος παρουσίασης μιας διαδικασίας ή ενός προγράμματος και συμβάλλει στην κατανόησή του.
- Το λογικό διάγραμμα χρησιμοποιεί συγκεκριμένα, αλλά διαφορετικά, γεωμετρικά σχήματα για να απεικονίσει τις διαδικασίες ανάλογα με το είδος τους.

Ασκήσεις Κεφαλαίου 1

- 1.1 Να καταστρώσετε ένα λογικό διάγραμμα το οποίο να απεικονίζει την διαδικασία της λύσης μιας εξίσωσης δευτέρου βαθμού. ★ ★
- 1.2 Αν γνωρίζουμε ότι: α). Δίσεκτο είναι ένα έτος όταν διαιρείται ακριβώς με το 4. β) Τα έτη που διαιρούνται ακριβώς με το 100 δεν είναι δίσεκτα εκτός αν διαιρούνται με το 400, καταστρώσετε ένα λογικό διάγραμμα μιας διαδικασίας η οποία να ζητάει το έτος και να απαντάει αν είναι δίσεκτο ή όχι. ★ ★
- 1.3 Αν γνωρίζουμε ότι ένας φορολογούμενος δεν πληρώνει φόρο στην περίπτωση που έχει ετήσιο εισόδημα κάτω από 7000€, πληρώνει 10% στην περίπτωση που έχει εισόδημα από 7000€ έως 15000€ και 20% για παραπάνω εισόδημα, καταστρώσετε ένα λογικό διάγραμμα μιας διαδικασίας η οποία να ζητάει το εισόδημα και να υπολογίζει το φόρο. ★ ★
- 1.4 Μελετήστε το παρακάτω λογικό διάγραμμα. Τι αποτέλεσμα θα έχει η διαδικασία αν δώσουμε με τη σειρά τους αριθμούς 12, 3, 10, 7, 1, 4, και 0. ★ ★ ★



1.5 Ποια από τα παρακάτω αληθεύουν: ★

- ☐ Η C είναι μια γλώσσα με αυστηρό έλεγχο.
- ☐ Η C συναντάται συνήθως σε ερμηνευτική μορφή.
- ☐ Σε μια μεταβλητή δεν μπορούμε να αλλάξουμε το όνομά της.
- ☐ Οι τύποι δεδομένων μπορεί να διαφέρουν σε διαφορετικές γλώσσες προγραμματισμού.
- ☐ Το λογικό διάγραμμα εξαρτάται από τη γλώσσα προγραμματισμού που χρησιμοποιούμε.

1.6 Μελετήστε το παρακάτω λογικό διάγραμμα. Τι αποτέλεσμα θα έχει η διαδικασία αν δώσουμε τους αριθμούς 12, 15, 145 και για τον κάθε ένα αριθμό ξεχωριστά. Πότε θα σταματήσει η διαδικασία; ★ ★

