ΕΠΑΝΑΛΗΠΤΙΚΑ ΘΕΜΑΤΑ ΠΡΟΕΤΟΙΜΑΣΙΑΣ ΤΕΛΙΚΗΣ ΕΞΕΤΑΣΗΣ ΘΕΩΡΙΑΣ

Θέμα 1 – Δομές επιλογής
1Α. Θεωρία

α) Δομές επιλογής είναι οι εντολές με τις οποίες το πρόγραμμα αποφασίζει ποια εντολή θα εκτελέσει, ανάλογα με κάποια συνθήκη (π.χ. if, if–else, switch). Χωρίς αυτές θα είχαμε μόνο σειριακή εκτέλεση, χωρίς «αποφάσεις».

Δώστε παραδείγματα από απλή , σύνθετη και πολλαπλή δομή.

1Β. Σύγκριση if–else και switch

α) Η if–else χρησιμοποιείται γενικά για κάθε είδους συνθήκη (με <, >, ==, λογικούς τελεστές κτλ.), ενώ η switch είναι πιο κατάλληλη όταν συγκρίνουμε μία μεταβλητή μόνο με συγκεκριμένες σταθερές ακέραιες ή χαρακτήρες (cases). Η switch κάνει τον κώδικα πιο καθαρό όταν έχουμε πολλές σταθερές επιλογές (π.χ. 1,2,3,4).

β) Για το μενού: ο χρήστης δίνει έναν ακέραιο (επιλογή) και σε switch(choice) ορίζονται case 1: για πρόσθεση, case 2: για αφαίρεση, case 3: για έξοδο, και στο default μπαίνει μήνυμα «Λάθος επιλογή».

Θέμα 2 – Δομές επανάληψης
2Α. Θεωρία επαναλήψεων

α) Οι δομές επανάληψης επιτρέπουν την επανάληψη εντολών πολλές φορές, χωρίς να τις γράφουμε ξανά και ξανά. Στη C βασικές δομές επανάληψης είναι οι while, do…while και for.

while: ελέγχει πρώτα τη συνθήκη και μετά (αν είναι αληθής) εκτελεί το σώμα.

do…while: εκτελεί πρώτα το σώμα και μετά ελέγχει τη συνθήκη (εκτελείται τουλάχιστον μία φορά).

for: είναι βολική όταν γνωρίζουμε από πριν πόσες επαναλήψεις θέλουμε (έχει σε μία γραμμή αρχικοποίηση, συνθήκη, μεταβολή).

β) Παράδειγμα με for: «Εμφάνισε τους αριθμούς από 1 μέχρι 10» (χρησιμοποιούμε μετρητή από 1 έως 10). Παράδειγμα με while: «Ζήτα από τον χρήστη έναν αριθμό μέχρι να δώσει 0» (δεν ξέρουμε πόσες φορές θα επαναληφθεί).

2Β. Ατέρμονη επανάληψη

α) Ατέρμονη επανάληψη συμβαίνει όταν η συνθήκη της επανάληψης δεν γίνεται ποτέ ψευδής, οπότε ο βρόχος δεν σταματά. Τυπικό λάθος: ξεχνάμε να αλλάξουμε τη μεταβλητή ελέγχου (π.χ. i δεν αυξάνεται μέσα στο while).

β) Αν η μεταβλητή ελέγχου δεν αλλάξει όπως πρέπει ή η συνθήκη είναι γραμμένη λάθος (π.χ. while (x != 0) αλλά ποτέ δεν δίνουμε 0), τότε η συνθήκη μένει πάντα αληθής και ο βρόχος δεν τερματίζεται.

Θέμα 3 – Πίνακες, αναζήτηση, ταξινόμηση

3Α. Πίνακες

α) Πίνακας στη C είναι μια συλλογή από στοιχεία του ίδιου τύπου, που έχουν κοινό όνομα αλλά διαφορετικό αριθμοδείκτη (index). Για δήλωση πίνακα 10 ακεραίων: int a[10];, όπου τα στοιχεία είναι a[0] έως a[9]. Το 1ο στοιχείο είναι a[0], το 5ο είναι a[4], το τελευταίο (10ο) είναι a[9].

β) Βήματα δημιουργίας και εμφάνισης ενός πίνακα με 5 βαθμούς μαθητών:
Δηλώνουμε έναν πίνακα 5 θέσεων για τους βαθμούς (π.χ. float grade).
Για κάθε μαθητή, ζητάμε από τον χρήστη τον βαθμό και τον αποθηκεύουμε σε κατάλληλο στοιχείο του πίνακα.
Στο τέλος, περνάμε τον πίνακα με επανάληψη και εμφανίζουμε όλους τους βαθμούς.

3Β. Γραμμική αναζήτηση

α) Στη γραμμική αναζήτηση ελέγχουμε τα στοιχεία του πίνακα ένα ένα, από την αρχή μέχρι το τέλος, μέχρι είτε να βρούμε το στοιχείο που ψάχνουμε είτε να τελειώσει ο πίνακας. Αν το βρούμε, σταματάμε και συνήθως επιστρέφουμε τη θέση του, αλλιώς δηλώνουμε ότι δεν βρέθηκε.

β) Ψευδοκώδικας:

Θέτω i = 0.

Όσο i < N και δεν βρέθηκε το στοιχείο:

Αν A[i] == ζητούμενο τότε «βρέθηκε».

Αλλιώς i = i + 1.

Αν βρέθηκε, επιστρέφω τη θέση i, αλλιώς μήνυμα «δεν βρέθηκε».

3Γ. Απλή ταξινόμηση (bubble sort)

α) Ο bubble sort συγκρίνει συνεχώς γειτονικά στοιχεία και αν είναι σε λάθος σειρά, τα ανταλλάσσει. Με πολλά «περάσματα» πάνω από τον πίνακα, τα μεγάλα στοιχεία «ανεβαίνουν» προς το τέλος και ο πίνακας τελικά γίνεται ταξινομημένος. (για αύξουσα σειρά ανεβαίνουν προς το τέλος)

β) Αν δεν συγκρίνουμε πάντα τα σωστά (γειτονικά) ζεύγη, κάποια στοιχεία μπορεί να μη μετακινηθούν εκεί που πρέπει. Έτσι, μετά το τέλος των περασμάτων, ο πίνακας μπορεί να μην είναι πλήρως ταξινομημένος (κάποια ζεύγη μένουν σε λάθος σειρά).

Θέμα 4 – Αρχεία και δείκτες
4Α. Αρχεία κειμένου

α) Αρχείο κειμένου είναι ένα αρχείο στο δίσκο που περιέχει χαρακτήρες (γραμμές κειμένου), τους οποίους μπορούμε να γράψουμε ή να διαβάσουμε από πρόγραμμα C. Ο τύπος FILE * είναι ένας δείκτης σε δομή που περιέχει πληροφορίες για το αρχείο (κατάσταση, θέση μέσα στο αρχείο κτλ.) και χρησιμοποιείται από συναρτήσεις όπως fopen, fprintf, fscanf, fclose.

β) Αποθήκευση ονομάτων και βαθμών μαθητών σε αρχείο:

Ανοίγουμε αρχείο σε λειτουργία εγγραφής (π.χ. "w" ή "w+").

Για κάθε μαθητή γράφουμε στο αρχείο το όνομα και τον βαθμό (π.χ. μία γραμμή ανά μαθητή).
Κλείνουμε το αρχείο.

Ανάγνωση:
Ανοίγουμε το αρχείο σε λειτουργία ανάγνωσης ("r").
Διαβάζουμε γραμμή-γραμμή ή πεδίο-πεδίο όνομα και βαθμό και εμφανίζουμε

Κλείνουμε το αρχείο.

4Β. Δείκτες – βασική ιδέα

α) Δείκτης είναι μια μεταβλητή που αντί για «κανονική» τιμή (π.χ. 5 ή 3.14) κρατά διεύθυνση μνήμης μιας άλλης μεταβλητής που είναι ένας δεκαεξαδικός αριθμός. Δηλαδή «δείχνει» στη θέση μνήμης όπου βρίσκεται η άλλη μεταβλητή, οπότε μέσω του δείκτη μπορούμε να διαβάσουμε ή να αλλάξουμε την τιμή της.

β) Αν x είναι int και p είναι δείκτης σε int που δείχνει στη x, τότε:

Αποθηκεύουμε στη p τη διεύθυνση της x (p = &x).

Για να αλλάξουμε την τιμή της x μέσω p, χρησιμοποιούμε *p (αποαναφορά), π.χ. *p = 10; αλλάζει την τιμή της x σε 10.

Θέμα 5 – Συναρτήσεις και παράμετροι
5Α. Έννοια συνάρτησης

α) Συνάρτηση στη C είναι ένα τμήμα κώδικα με όνομα, που εκτελεί μια συγκεκριμένη εργασία και μπορεί να κληθεί από άλλα σημεία του προγράμματος. Η δήλωση (prototype) ενημερώνει τον μεταγλωττιστή για το όνομα, τον τύπο επιστροφής και τους τύπους παραμέτρων, ενώ ο ορισμός περιέχει τον πραγματικό κώδικα της συνάρτησης (το σώμα).

β) Πλεονεκτήματα:
Τμηματικός προγραμματισμός: σπάμε το πρόγραμμα σε μικρά, πιο εύκολα κομμάτια.
Επαναχρησιμοποίηση: μπορούμε να καλέσουμε την ίδια συνάρτηση από πολλά σημεία, χωρίς να ξαναγράφουμε τον ίδιο κώδικα.

5Β. Ροή εκτέλεσης

α) Για συνάρτηση που επιστρέφει τιμή:

Όταν γίνεται η κλήση, **αντιγράφονται** οι πραγματικές παράμετροι στις τυπικές παραμέτρους της συνάρτησης.
Εκτελείται ο κώδικας της συνάρτησης.
Με την εντολή return, παράγεται μια τιμή επιστροφής, η οποία «επιστρέφεται» στο σημείο κλήσης της συνάρτησης και μπορεί να αποθηκευτεί σε μεταβλητή ή να χρησιμοποιηθεί σε έκφραση.

β) Για συνάρτηση τύπου void:
Πάλι αντιγράφονται οι παράμετροι και εκτελείται το σώμα της συνάρτησης.
Δεν επιστρέφεται τιμή μέσω return και συνήθως κάνουμε μια λειτουργία που δεν επιστρέφει τιμή μέσω return. Μπορεί να είναι συνήθως εμφάνιση αποτελεσμάτων. Επίσης void βάζω στον μηχανισμό call by reference

Θέμα 6 – Call by value και "call by reference"
6A. Call by value στη C

α) Call by value σημαίνει ότι όταν καλούμε μια συνάρτηση, δίνονται αντίγραφα των τιμών των παραμέτρων, όχι οι ίδιες οι μεταβλητές. Άρα, μέσα στη συνάρτηση δουλεύουμε με αντίγραφα και όχι με τις αρχικές μεταβλητές.

β) Όταν η συνάρτηση προσπαθεί να αλλάξει μια int παράμετρο, αλλά αυτή περνά «κατά τιμή», αλλάζει μόνο το αντίγραφο που υπάρχει μέσα στη συνάρτηση. Η αρχική μεταβλητή στον καλούντα κώδικα δεν επηρεάζεται, γιατί δεν δόθηκε η διεύθυνσή της αλλά μόνο η τιμή της.

6B. Προσομοίωση call by reference με δείκτες

α) Για να πετύχουμε call by reference, αντί να δίνουμε τη μεταβλητή, δίνουμε στη συνάρτηση τη διεύθυνσή της (δηλαδή δείκτη προς αυτήν). Έτσι, μέσα στη συνάρτηση μπορούμε, μέσω του δείκτη, να αλλάξουμε την τιμή της μεταβλητής που βρίσκεται στον καλούντα κώδικα.

β) Παράδειγμα σε λόγια:
Έχουμε μια μεταβλητή x στον κύριο κώδικα και θέλουμε μια συνάρτηση που να την αυξάνει κατά 1.

Αντί να περάσουμε «την τιμή του x», περνάμε «τη διεύθυνση του x».

Μέσα στη συνάρτηση, χρησιμοποιούμε αποαναφορά του δείκτη (*p) για να αυξήσουμε τη μεταβλητή που δείχνεται (π.χ. *p = *p + 1), οπότε αλλάζει η πραγματική x