

Άσκηση 1 – Απλή / σύνθετη / πολλαπλή επιλογή Εκφώνηση

Να γραφεί πρόγραμμα C που:

- α) Διαβάζει έναν ακέραιο βαθμό (0–10).**
- β) Αν ο βαθμός είναι < 5, εμφανίζει «Ανεπιτυχής».**
- γ) Αν ο βαθμός είναι από 5 έως 8, εμφανίζει «Καλός».**
- δ) Αν ο βαθμός είναι 9 ή 10, εμφανίζει «Άριστος».**

```
#include <stdio.h>

int main(void) {
    int grade;

    printf("Δώσε βαθμό (0-10): ");
    scanf("%d", &grade);

    if (grade < 0 || grade > 10) {
        printf("Μη έγκυρος βαθμός\n");
    }
    else if (grade < 5) {
        printf("Ανεπιτυχής\n");
    }
    else if (grade <= 8) {
        printf("Καλός\n");
    }
    else { /* εδώ μένουν μόνο 9 ή 10 */
        printf("Άριστος\n");
    }

    return 0;
}
```

Άσκηση 2 – Μενού με switch

Εκφώνηση

Να γραφεί πρόγραμμα C που εμφανίζει το μενού:

Πρόσθεση δύο ακεραίων

Αφαίρεση δύο ακεραίων

Έξοδος

- α) Να ζητά από τον χρήστη επιλογή (1–3).**
- β) Να υλοποιεί το μενού με switch.**

γ) Σε περίπτωση λάθους επιλογής, να εμφανίζει μήνυμα κατάλληλο.

Ενδεικτική λύση

```
int choice, a, b;  
printf("1. Πρόσθεση\n2. Αφαίρεση\n3. Έξοδος\n");  
scanf("%d", &choice);
```

```
switch (choice) {  
    case 1:  
        scanf("%d%d", &a, &b);  
        printf("Άθροισμα = %d\n", a + b);  
        break;  
    case 2:  
        scanf("%d%d", &a, &b);  
        printf("Διαφορά = %d\n", a - b);  
        break;  
    case 3:  
        printf("Έξοδος...\n");  
        break;  
    default:  
        printf("Λάθος επιλογή\n");  
}
```

Άσκηση 3 – for και while

Εκφώνηση

α) Να γραφεί πρόγραμμα με for που εμφανίζει τους αριθμούς από 1 έως 10 στην ίδια γραμμή.

β) Να γραφεί πρόγραμμα με while που ζητά συνεχώς έναν αριθμό και σταματά όταν ο χρήστης δώσει 0.

Ενδεικτική λύση

```
c  
/* (α) for */  
for (int i = 1; i <= 10; i++)
```

```

printf("%d ", i);

/* (β) while */
int x;
scanf("%d", &x);
while (x != 0) {
    printf("Δώσατε %d\n", x);
    scanf("%d", &x);
}

```

Άσκηση 4 – Ατέρμονη επανάληψη (εντοπισμός λάθους)

Εκφώνηση

Δίνεται το απόσπασμα:

```

c
int i = 1;
while (i <= 10) {
    printf("%d\n", i);
}

```

α) Γιατί ο βρόχος είναι ατέρμονος;

β) Να διορθώσετε τον κώδικα ώστε να εμφανίζει τους αριθμούς 1–10 και να τερματίζει.

Λύση

α) Η μεταβλητή *i* δεν αλλάζει μέσα στο while, άρα η συνθήκη *i* <= 10 μένει πάντα αληθής.

β) Διόρθωση:

```

c
int i = 1;
while (i <= 10) {
    printf("%d\n", i);
    i++;    /* απαραίτητη μεταβολή της i */
}

```

Άσκηση 5 – Πίνακας 5 βαθμών

Εκφώνηση

Να γραφεί πρόγραμμα C που:

α) Δηλώνει πίνακα 5 θέσεων για βαθμούς float.

β) Διαβάζει τους 5 βαθμούς από το πληκτρολόγιο.

γ) Τους εμφανίζει όλους και στο τέλος τον μέσο όρο.

Ενδεικτική λύση

```
c
float grade[5], sum = 0.0;
for (int i = 0; i < 5; i++) {
    scanf("%f", &grade[i]);
    sum += grade[i];
}

for (int i = 0; i < 5; i++)
    printf("grade[%d] = %.2f\n", i, grade[i]);

printf("Μέσος όρος = %.2f\n", sum / 5);
```

Άσκηση 6 – Γραμμική αναζήτηση σε πίνακα

Εκφώνηση

Δίνεται πίνακας 10 ακεραίων. Να γραφεί συνάρτηση linearSearch που:

Δέχεται τον πίνακα, το μέγεθος και τον «ζητούμενο» αριθμό.

Επιστρέφει τη θέση όπου βρέθηκε ή -1 αν δεν βρέθηκε.

Ενδεικτική λύση

```
c
int linearSearch(int A[], int n, int target) {
    int i;
    for (i = 0; i < n; i++) {
        if (A[i] == target)
            return i; /* βρέθηκε, επιστρέφω θέση */
    }
    return -1; /* δεν βρέθηκε */
}
```

Άσκηση 7 – Ένας απλός bubble sort

Εκφώνηση

Να γραφεί συνάρτηση bubbleSort που ταξινομεί έναν πίνακα 5 ακεραίων σε αύξουσα σειρά, χρησιμοποιώντας τη μέθοδο bubble sort (σύγκριση γειτονικών στοιχείων και ανταλλαγή όπου χρειάζεται).

Ενδεικτική λύση

```
c
void bubbleSort(int A[], int n) {
    int i, j, temp;
    for (i = 0; i < n - 1; i++) {
        for (j = 0; j < n - 1 - i; j++) {
```

```

        if (A[j] > A[j + 1]) {
            temp = A[j];
            A[j] = A[j + 1];
            A[j + 1] = temp;
        }
    }
}
}

```

Άσκηση 8 – Αρχείο κειμένου με ονόματα και βαθμούς

Εκφώνηση

Να γραφεί πρόγραμμα C που:

- α) Ανοίγει ένα αρχείο «students.txt» για εγγραφή.
- β) Διαβάζει από το πληκτρολόγιο 3 ζεύγη (όνομα, βαθμός) και τα γράφει στο αρχείο.
- γ) Κλείνει το αρχείο.
- δ) Το ξανανοίγει για ανάγνωση και εμφανίζει μόνο τους μαθητές με βαθμό ≥ 10 .

Ενδεικτική λύση

```

c
#include <stdio.h>

int main(void) {
    FILE *fp;
    char name[20];
    float grade;
    int i;

    fp = fopen("students.txt", "w");
    for (i = 0; i < 3; i++) {
        scanf("%s%f", name, &grade);
        fprintf(fp, "%s %.2f\n", name, grade);
    }
    fclose(fp);

    fp = fopen("students.txt", "r");
    while (fscanf(fp, "%s%f", name, &grade) == 2) {
        if (grade >= 10.0)
            printf("%s %.2f\n", name, grade);
    }
    fclose(fp);
    return 0;
}

```

Άσκηση 9 – Call by value

Εκφώνηση

Δίνεται η συνάρτηση:

```
void addOne(int x) {  
    x = x + 1;  
}
```

α) Να γραφεί main που δηλώνει μια μεταβλητή `int a = 5`, καλεί τη συνάρτηση `addOne(a)` και στη συνέχεια εμφανίζει την τιμή της `a`.

β) Να εξηγηθεί γιατί η `a` παραμένει 5 (call by value).

Ενδεικτική λύση

```
c  
int main(void) {  
    int a = 5;  
    addOne(a);  
    printf("a = %d\n", a); /* θα εκτυπώσει 5 */  
    return 0;  
}
```

Εξήγηση: στη συνάρτηση περνά αντίγραφο της τιμής της `a`, όχι η ίδια η μεταβλητή· η αλλαγή του `x` δεν επηρεάζει το `a`.

Άσκηση 10 – Call by reference με δείκτη

Εκφώνηση

α) Να γραφεί συνάρτηση `increase` που να δέχεται δείκτη σε `int` και να αυξάνει την τιμή της μεταβλητής κατά 1.

β) Να γραφεί main που δηλώνει `int x = 7`, καλεί την `increase(&x)` και εμφανίζει πριν και μετά την κλήση την τιμή της `x`.

Ενδεικτική λύση

```
c  
void increase(int *p) {  
    *p = *p + 1;
```

```
}
```

```
int main(void) {  
    int x = 7;  
    printf("Πριν: x = %d\n", x);  
    increase(&x);  
    printf("Μετά: x = %d\n", x); /* θα εκτυπώσει 8 */  
    return 0;  
}
```

Εδώ η συνάρτηση παίρνει τη διεύθυνση της x, άρα μέσω *p αλλάζει την πραγματική μεταβλητή (call by reference με δείκτη).