4ο Φροντιστήριο Δομημένου Προγραμματισμού

Άσκηση 1:

Να γραφεί πρόγραμμα σε C που ζητά από τον χρήστη να εισαγάγει **10 ακέραιους αριθμούς** και τους αποθηκεύει σε έναν μονοδιάστατο πίνακα. Υπολογίζει και εμφανίζει: Το **άθροισμα** των στοιχείων, Τον **μέσο όρο** Το **μέγιστο** και το **ελάχιστο** στοιχείο, Πόσα στοιχεία είναι **άρτια** και πόσα **περιττά**

```
#include <stdio.h>   // βιβλιοθήκη για είσοδο/έξοδο
#define SIZE 10      // σταθερά για το μέγεθος του πίνακα
int main() {
    int numbers[SIZE];    // δήλωση πίνακα 10 ακεραίων
    int sum = 0;          // μεταβλητή για το άθροισμα
    int max, min;         // μεταβλητές για μέγιστο και ελάχιστο
    int evenCount = 0;    // πλήθος άρτιων
    int oddCount = 0;     // πλήθος περιττών
    float average;        // μέσος όρος
    // ===== Εισαγωγή δεδομένων =====
    printf("Δώσε %d ακέραιους αριθμούς:\n", SIZE);
    for (int i = 0; i < SIZE; i++) {
        printf("Αριθμός %d: ", i + 1);
        scanf("%d", &numbers[i]);  // αποθήκευση στον πίνακα
    }
    // ===== Αρχικοποίηση μέγιστου και ελάχιστου =====
    max = numbers[0];
    min = numbers[0];
    // ===== Επεξεργασία πίνακα =====
    for (int i = 0; i < SIZE; i++) {
        sum += numbers[i];              // πρόσθεση στο άθροισμα
        if (numbers[i] > max)           // έλεγχος για νέο μέγιστο
            max = numbers[i];
        if (numbers[i] < min)           // έλεγχος για νέο ελάχιστο
            min = numbers[i];
        if (numbers[i] % 2 == 0)        // έλεγχος για άρτιο
            evenCount++;
        else                            // αλλιώς είναι περιττό
            oddCount++;
    }
    // ===== Υπολογισμός μέσου όρου =====
    average = (float)sum / SIZE;
    // ===== Εμφάνιση αποτελεσμάτων =====
    printf("\n--- Αποτελέσματα ---\n");
    printf("Άθροισμα: %d\n", sum);
    printf("Μέσος όρος: %.2f\n", average);
    printf("Μέγιστος αριθμός: %d\n", max);
    printf("Ελάχιστος αριθμός: %d\n", min);
    printf("Άρτιοι: %d\n", evenCount);
    printf("Περιττοί: %d\n", oddCount);
    return 0; // τέλος προγράμματος
}
```

Επεξήγηση Κώδικα Βήμα-Βήμα

Βήμα	Τι κάνει
#define SIZE 10	Ορίζει το μέγεθος του πίνακα (10 στοιχεία)
int numbers[SIZE];	Δημιουργεί πίνακα 10 ακεραίων
for (int i = 0; i < SIZE; i++)	Διαβάζει 10 αριθμούς από τον χρήστη
sum += numbers[i];	Υπολογίζει το άθροισμα
if (numbers[i] > max)	Ελέγχει αν υπάρχει νέο μέγιστο
if (numbers[i] % 2 == 0)	Μετρά πόσα στοιχεία είναι άρτια
average = (float)sum / SIZE;	Υπολογίζει τον μέσο όρο με μετατροπή σε float
printf(...)	Εμφανίζει τα αποτελέσματα

Παράδειγμα Εκτέλεσης

Δώσε 10 ακέραιους αριθμούς:
Αριθμός 1: 5
Αριθμός 2: 8
Αριθμός 3: 2
Αριθμός 4: 9
Αριθμός 5: 3
Αριθμός 6: 10
Αριθμός 7: 6
Αριθμός 8: 4
Αριθμός 9: 7
Αριθμός 10: 1

--- Αποτελέσματα ---
Άθροισμα: 55
Μέσος όρος: 5.50
Μέγιστος αριθμός: 10
Ελάχιστος αριθμός: 1
Άρτιοι: 5
Περιττοί: 5

Άσκηση 2: Να γραφεί πρόγραμμα σε C που δημιουργεί έναν πίνακα 5x5 ακεραίων. Ζητά από τον χρήστη να εισαγάγει τα στοιχεία του πίνακα.Υπολογίζει και εμφανίζει: Το άθροισμα και τον μέσο όρο κάθε γραμμής, Το άθροισμα και τον μέσο όρο κάθε στήλης, Το μέγιστο και το ελάχιστο στοιχείο του πίνακα

```
#include <stdio.h>
#define ROWS 5
#define COLS 5
int main() {
    int matrix[ROWS][COLS];
    int rowSum, colSum;
    int max, min;
    float rowAvg, colAvg;
    // ===== Εισαγωγή στοιχείων πίνακα =====
    printf("Δώσε %d x %d ακέραιους αριθμούς για τον πίνακα:\n", ROWS, COLS);
    for (int i = 0; i < ROWS; i++) {
        for (int j = 0; j < COLS; j++) {
            printf("Στοιχείο [%d][%d]: ", i+1, j+1);
            scanf("%d", &matrix[i][j]);
        }
    }
    // ===== Αρχικοποίηση μέγιστου και ελάχιστου =====
    max = min = matrix[0][0];
    // ===== Υπολογισμοί ανά γραμμή =====
    printf("\n--- Άθροισμα και μέσος όρος ανά γραμμή ---\n");
    for (int i = 0; i < ROWS; i++) {
        rowSum = 0;
        for (int j = 0; j < COLS; j++) {
            rowSum += matrix[i][j];
            // Ελέγχουμε μέγιστο και ελάχιστο
            if (matrix[i][j] > max) max = matrix[i][j];
            if (matrix[i][j] < min) min = matrix[i][j];
        }
        rowAvg = (float)rowSum / COLS;
        printf("Γραμμή %d -> Άθροισμα: %d, Μέσος Όρος: %.2f\n", i+1, rowSum, rowAvg);
    }
    // ===== Υπολογισμοί ανά στήλη =====
    printf("\n--- Άθροισμα και μέσος όρος ανά στήλη ---\n");
    for (int j = 0; j < COLS; j++) {
        colSum = 0;
        for (int i = 0; i < ROWS; i++) {
            colSum += matrix[i][j];
        }
        colAvg = (float)colSum / ROWS;
        printf("Στήλη %d -> Άθροισμα: %d, Μέσος Όρος: %.2f\n", j+1, colSum, colAvg);
    }
    // ===== Εμφάνιση μέγιστου και ελάχιστου =====
    printf("\nΜέγιστο στοιχείο πίνακα: %d\n", max);
    printf("Ελάχιστο στοιχείο πίνακα: %d\n", min);
    return 0;
}
```

Σχολιασμός Κώδικα Δισδιάστατου Πίνακα

- Ο πίνακας δηλώνεται ως int matrix[ROWS][COLS];.
- Οι δύο for βρόχοι χρησιμοποιούνται για είσοδο στοιχείων.
- Άθροισμα και μέσος όρος υπολογίζονται ξεχωριστά για γραμμές και στήλες.
- Μέγιστο και ελάχιστο βρίσκονται σε έναν ενιαίο βρόχο για αποτελεσματικότητα.

Άσκηση 3: Να γραφεί πρόγραμμα σε C που: Δημιουργεί έναν τρισδιάστατο πίνακα 2 x 3 x 4 (2 "πίνακες" των 3x4). Ζητά από τον χρήστη να εισάγει τα στοιχεία του πίνακα. Υπολογίζει και εμφανίζει: Το άθροισμα κάθε επιπέδου (το πρώτο επίπεδο, το δεύτερο...), Το μέγιστο και ελάχιστο όλων των στοιχείων

```
#include <stdio.h>
#define LEVELS 2
#define ROWS 3
#define COLS 4
int main() {
    int matrix[LEVELS][ROWS][COLS];
    int sum, max, min;
    // ===== Εισαγωγή στοιχείων =====
    for (int l = 0; l < LEVELS; l++) {
        printf("Επίπεδο %d:\n", l+1);
        for (int i = 0; i < ROWS; i++) {
            for (int j = 0; j < COLS; j++) {
                printf("Στοιχείο [%d][%d][%d]: ", l+1, i+1, j+1);
                scanf("%d", &matrix[l][i][j]);
            }
        }
    }
    // ===== Αρχικοποίηση μέγιστου και ελάχιστου =====
    max = min = matrix[0][0][0];

    // ===== Άθροισμα ανά επίπεδο =====
    for (int l = 0; l < LEVELS; l++) {
        sum = 0;
        for (int i = 0; i < ROWS; i++) {
            for (int j = 0; j < COLS; j++) {
                sum += matrix[l][i][j];

                // Έλεγχος μέγιστου/ελάχιστου
                if (matrix[l][i][j] > max) max = matrix[l][i][j];
                if (matrix[l][i][j] < min) min = matrix[l][i][j];
            }
        }
        printf("Άθροισμα επιπέδου %d: %d\n", l+1, sum);
    }

    // ===== Εμφάνιση μέγιστου και ελάχιστου =====
    printf("\nΜέγιστο στοιχείο πίνακα: %d\n", max);
    printf("Ελάχιστο στοιχείο πίνακα: %d\n", min);

    return 0;
}
```

Σχολιασμός Κώδικα Τρισδιάστατου Πίνακα

- Ο πίνακας δηλώνεται ως matrix[LEVELS][ROWS][COLS].
- Τρεις for βρόχοι χρησιμοποιούνται για την είσοδο στοιχείων.
- Το άθροισμα κάθε επιπέδου υπολογίζεται σε εσωτερικό βρόχο.
- Μέγιστο και ελάχιστο υπολογίζονται ταυτόχρονα για όλα τα στοιχεία.

Άσκηση 4: Να γραφεί πρόγραμμα σε C που διαβάζει **3 ακέραιους αριθμούς** από τον χρήστη. Χρησιμοποιεί **μόνο συναρτήσεις** για να υπολογίσει: Το **άθροισμα,** Το **μέσο όρο,** Το **μέγιστο,** Το **ελάχιστο. Δεν χρησιμοποιούνται πίνακες**, όλα γίνονται με απλές μεταβλητές και παραμέτρους συναρτήσεων.

```
#include <stdio.h>
// Δήλωση συναρτήσεων
int sum3(int a, int b, int c);
float avg3(int a, int b, int c);
int max3(int a, int b, int c);
int min3(int a, int b, int c);
int main() {
    int num1, num2, num3;
    // Εισαγωγή αριθμών
    printf("Δώσε τρεις ακέραιους αριθμούς:\n");
    printf("Αριθμός 1: ");
    scanf("%d", &num1);
    printf("Αριθμός 2: ");
    scanf("%d", &num2);
    printf("Αριθμός 3: ");
    scanf("%d", &num3);
    // Κλήση συναρτήσεων
    int sum = sum3(num1, num2, num3);
    float avg = avg3(num1, num2, num3);
    int maximum = max3(num1, num2, num3);
    int minimum = min3(num1, num2, num3);
    // Εμφάνιση αποτελεσμάτων
    printf("\nΑποτελέσματα:\n");
    printf("Άθροισμα: %d\n", sum);
    printf("Μέσος όρος: %.2f\n", avg);
    printf("Μέγιστος αριθμός: %d\n", maximum);
    printf("Ελάχιστος αριθμός: %d\n", minimum);
    return 0;
}
// ===== ΥΛΟΠΟΙΗΣΗ ΣΥΝΑΡΤΗΣΕΩΝ =====
// Υπολογισμός αθροίσματος
int sum3(int a, int b, int c) {
    return a + b + c;
}
// Υπολογισμός μέσου όρου
float avg3(int a, int b, int c) {
    return (float)(a + b + c) / 3;
}
// Εύρεση μέγιστου
int max3(int a, int b, int c) {
    int max = a;
    if (b > max) max = b;
    if (c > max) max = c;
    return max;
}
// Εύρεση ελάχιστου
int min3(int a, int b, int c) {
    int min = a;
    if (b < min) min = b;
    if (c < min) min = c;
    return min;
}
```

Σχολιασμός Κώδικα

1. **Κάθε συνάρτηση** παίρνει τους τρεις αριθμούς ως παραμέτρους και επιστρέφει το αποτέλεσμα.
2. **main()** δεν κάνει κανέναν υπολογισμό παρά μόνο καλεί τις συναρτήσεις.
3. Όλες οι πράξεις γίνονται **χωρίς πίνακα**, ιδανικό για αρχάριους στη χρήση συναρτήσεων.
4. Η επιστροφή τιμών γίνεται με return, ώστε να κρατάμε τον κώδικα καθαρό και δομημένο.

Άσκηση 5: Ένας καθηγητής θέλει να αναλύσει τα αποτελέσματα 10 μαθητών σε 5 μαθήματα. Να γραφεί πρόγραμμα σε C που αποθηκεύει τους βαθμούς των μαθητών σε **δισδιάστατο πίνακα 10x5** (10 μαθητές x 5 μαθήματα). Υπολογίζει: Το **μέγιστο, ελάχιστο και μέσο όρο βαθμό** κάθε μαθητή, Το **μέγιστο, ελάχιστο και μέσο όρο βαθμό** κάθε μαθήματος. Το **μέγιστο, ελάχιστο και μέσο όρο συνολικά** για όλους τους μαθητές.

```
#include <stdio.h>
#define STUDENTS 10
#define SUBJECTS 5
// ===== ΠΡΩΤΟΤΥΠΑ ΣΥΝΑΡΤΗΣΕΩΝ =====
// Συναρτήσεις για μαθητή (γραμμή)
int maxStudent(int grades[STUDENTS][SUBJECTS], int student);
int minStudent(int grades[STUDENTS][SUBJECTS], int student);
float avgStudent(int grades[STUDENTS][SUBJECTS], int student);
// Συναρτήσεις για μάθημα (στήλη)
int maxSubject(int grades[STUDENTS][SUBJECTS], int subject);
int minSubject(int grades[STUDENTS][SUBJECTS], int subject);
float avgSubject(int grades[STUDENTS][SUBJECTS], int subject);
// Συναρτήσεις για όλο τον πίνακα
int maxAll(int grades[STUDENTS][SUBJECTS]);
int minAll(int grades[STUDENTS][SUBJECTS]);
float avgAll(int grades[STUDENTS][SUBJECTS]);
int main() {
  int grades[STUDENTS][SUBJECTS];

  // ===== Εισαγωγή βαθμών =====
  printf("Δώσε τους βαθμούς των %d μαθητών σε %d μαθήματα:\n", STUDENTS, SUBJECTS);
  for (int i = 0; i < STUDENTS; i++) {
    for (int j = 0; j < SUBJECTS; j++) {
      printf("Μαθητής %d, Μάθημα %d: ", i+1, j+1);
      scanf("%d", &grades[i][j]);
    }
  }
  // ===== Ανάλυση ανά μαθητή =====
  printf("\n--- Ανάλυση ανά μαθητή ---\n");
  for (int i = 0; i < STUDENTS; i++) {
    printf("Μαθητής %2d -> Max: %3d, Min: %3d, Μέσος: %6.2f\n",
      i+1, maxStudent(grades, i), minStudent(grades, i), avgStudent(grades, i));
  }
  // ===== Ανάλυση ανά μάθημα =====
  printf("\n--- Ανάλυση ανά μάθημα ---\n");
  for (int j = 0; j < SUBJECTS; j++) {
    printf("Μάθημα %d -> Max: %3d, Min: %3d, Μέσος: %6.2f\n",
      j+1, maxSubject(grades, j), minSubject(grades, j), avgSubject(grades, j));
  }
  // ===== Ανάλυση συνολικά =====
  printf("\n--- Ανάλυση συνολικά ---\n");
  printf("Μέγιστος βαθμός: %d\n", maxAll(grades));
  printf("Ελάχιστος βαθμός: %d\n", minAll(grades));
  printf("Μέσος βαθμός: %.2f\n", avgAll(grades));
  return 0;
}
```

```
// ===== ΥΛΟΠΟΙΗΣΗ ΣΥΝΑΡΤΗΣΕΩΝ =====

// --- Μαθητής ---
int maxStudent(int grades[STUDENTS][SUBJECTS], int student) {
  int max = grades[student][0];
  for (int j = 1; j < SUBJECTS; j++)
    if (grades[student][j] > max) max = grades[student][j];
  return max;
}

int minStudent(int grades[STUDENTS][SUBJECTS], int student) {
  int min = grades[student][0];
  for (int j = 1; j < SUBJECTS; j++)
    if (grades[student][j] < min) min = grades[student][j];
  return min;
}

float avgStudent(int grades[STUDENTS][SUBJECTS], int student) {
  int sum = 0;
  for (int j = 0; j < SUBJECTS; j++)
    sum += grades[student][j];
  return (float)sum / SUBJECTS;
}

// --- Μάθημα ---
int maxSubject(int grades[STUDENTS][SUBJECTS], int subject) {
  int max = grades[0][subject];
  for (int i = 1; i < STUDENTS; i++)
    if (grades[i][subject] > max) max = grades[i][subject];
  return max;
}

int minSubject(int grades[STUDENTS][SUBJECTS], int subject) {
  int min = grades[0][subject];
  for (int i = 1; i < STUDENTS; i++)
    if (grades[i][subject] < min) min = grades[i][subject];
  return min;
}

float avgSubject(int grades[STUDENTS][SUBJECTS], int subject) {
  int sum = 0;
  for (int i = 0; i < STUDENTS; i++)
    sum += grades[i][subject];
  return (float)sum / STUDENTS;
}
```

```
// --- Όλος ο πίνακας ---
int maxAll(int grades[STUDENTS][SUBJECTS]) {
  int max = grades[0][0];
  for (int i = 0; i < STUDENTS; i++)
    for (int j = 0; j < SUBJECTS; j++)
      if (grades[i][j] > max) max = grades[i][j];
  return max;
}

int minAll(int grades[STUDENTS][SUBJECTS]) {
  int min = grades[0][0];
  for (int i = 0; i < STUDENTS; i++)
    for (int j = 0; j < SUBJECTS; j++)
      if (grades[i][j] < min) min = grades[i][j];
  return min;
}

float avgAll(int grades[STUDENTS][SUBJECTS]) {
  int sum = 0;
  for (int i = 0; i < STUDENTS; i++)
    for (int j = 0; j < SUBJECTS; j++)
      sum += grades[i][j];
  return (float)sum / (STUDENTS * SUBJECTS);
}
```

Σχολιασμός και Λειτουργία

1. **Συναρτήσεις για μαθητή (γραμμή)**:
    - Υπολογίζουν το μέγιστο, ελάχιστο και μέσο όρο για κάθε μαθητή.
    - Παίρνουν ως όρισμα τον πίνακα και τον αριθμό της γραμμής.
2. **Συναρτήσεις για μάθημα (στήλη)**:
    - Υπολογίζουν το μέγιστο, ελάχιστο και μέσο όρο για κάθε μάθημα.
    - Παίρνουν ως όρισμα τον πίνακα και τον αριθμό της στήλης.
3. **Συναρτήσεις για όλο τον πίνακα**:
    - Σαρώνουν όλα τα στοιχεία για να βρουν το συνολικό μέγιστο, ελάχιστο και μέσο όρο.
4. **main()**:
    - Διαβάζει βαθμούς μαθητών
    - Καλεί όλες τις συναρτήσεις και εμφανίζει τα αποτελέσματα με ξεκάθαρο τρόπο.

Άσκηση 5: Να γραφεί πρόγραμμα σε C που Διαβάζει δύο πίνακες 3x3 από τον χρήστη. Υπολογίζει το γινόμενο των δύο πινάκων χρησιμοποιώντας συνάρτηση. Εμφανίζει το αποτέλεσμα σε μορφή πίνακα 3x3. Σημείωση: Το γινόμενο δύο πινάκων A·B ορίζεται ως:

$$C[i][j] = \sum_{k=0}^{n-1} A[i][k] * B[k][j]$$

```
#include <stdio.h>
#define SIZE 3
// Δήλωση συνάρτησης
void multiplyMatrices(int a[SIZE][SIZE], int b[SIZE][SIZE],
int result[SIZE][SIZE]);
int main() {
    int A[SIZE][SIZE];
    int B[SIZE][SIZE];
    int C[SIZE][SIZE];
    // Εισαγωγή πρώτου πίνακα
    printf("Δώσε στοιχεία του πρώτου πίνακα 3x3:\n");
    for(int i=0; i<SIZE; i++)
        for(int j=0; j<SIZE; j++) {
            printf("A[%d][%d]: ", i+1, j+1);
            scanf("%d", &A[i][j]);
        }
    // Εισαγωγή δεύτερου πίνακα
    printf("Δώσε στοιχεία του δεύτερου πίνακα 3x3:\n");
    for(int i=0; i<SIZE; i++)
        for(int j=0; j<SIZE; j++) {
            printf("B[%d][%d]: ", i+1, j+1);
            scanf("%d", &B[i][j]);
        }
    // Κλήση συνάρτησης για γινόμενο
    multiplyMatrices(A, B, C);
    // Εμφάνιση αποτελέσματος
    printf("\nΤο γινόμενο των δύο πινάκων είναι:\n");
    for(int i=0; i<SIZE; i++) {
        for(int j=0; j<SIZE; j++)
            printf("%5d", C[i][j]);
        printf("\n");
    }
    return 0;
}
// ===== ΥΛΟΠΟΙΗΣΗ ΣΥΝΑΡΤΗΣΗΣ =====
void multiplyMatrices(int a[SIZE][SIZE], int b[SIZE][SIZE], int
result[SIZE][SIZE]) {
    for(int i=0; i<SIZE; i++) {
        for(int j=0; j<SIZE; j++) {
            result[i][j] = 0;           // Αρχικοποίηση
            for(int k=0; k<SIZE; k++)
                result[i][j] += a[i][k] * b[k][j]; // Άθροισμα γινομένων
        }
    }
}
```

Σχολιασμός Κώδικα

1. **Συνάρτηση multiplyMatrices**:
    - Παίρνει δύο πίνακες a και b και τον πίνακα result για το αποτέλεσμα.
    - Χρησιμοποιεί **τρεις βρόχους**:
        - i → γραμμές πρώτου πίνακα
        - j → στήλες δεύτερου πίνακα
        - k → για το άθροισμα των γινομένων
2. **main()**:
    - Διαβάζει τα στοιχεία των δύο πινάκων.
    - Καλεί τη συνάρτηση και εμφανίζει το αποτέλεσμα με σωστή μορφοποίηση.
3. **Αρχικοποίηση result[i][j] = 0**:
    - Απαραίτητο πριν από το άθροισμα για να μην έχει σκουπίδια η τιμή.