07.05.2025

BLG102E

Introduction to Scientific and Engineering Computation (C)

Programming Practice Questions 4

Question 1 - Sensor Deviation Detector

A lab is recording daily temperature values from a sensor. Occasionally, sensor errors cause outlier values that need to be flagged. A value is considered an "outlier" if its absolute difference from the previous day exceeds 10°C.

Task:

- Read temperature values into an array (max 50 values).
- Write a function to find all outlier days.
- Print the index and value of each outlier.
- Example function:

```
void detectOutliers(int temps[], int n);
```

Example Input:

```
Enter number of days: 6
Enter temperatures: 23 24 22 38 39 25
```

Example Output:

```
Outlier at day 3: 38 (difference from previous: 16)
Outlier at day 5: 25 (difference from previous: 14)
```

Question 2 - Rainfall Analyzer

You are tracking rainfall (mm) across 5 cities for 4 weeks. Data is stored in a 5x4 matrix.

Task:

- Read the matrix from the user.
- Compute and print total rainfall for each **week** (i.e., sum per column).
- Example function:

```
void weeklyRainfall(int data[][4], int rows);
```

Example Output:

```
Week 1: 240 mm
Week 2: 220 mm
Week 3: 195 mm
Week 4: 260 mm
```

Question 3 - Grade Difference Detector

You have a list of students, each with **2 exam scores**. You want to identify students whose grades improved or dropped drastically.

- Read a 6x2 matrix: 6 students, 2 scores each.
- Print the difference between exam 1 and exam 2 for each student.
- Flag students whose score changed more than 15 points.
- Example function:

```
void detectChanges(int scores[][2], int studentCount);
```

Example Output:

```
Student 1: +12
Student 2: -18  ⚠ Significant drop
Student 3: +5
Student 4: +25  ⚠ Significant improvement
Student 5: -2
Student 6: +0
```

Question 4 - Temperature Trend Analyzer

You are developing a climate app. The app must detect whether a given sequence of daily temperature measurements shows a **steady increasing trend**, **steady decreasing trend**, or **no consistent trend**.

- Prompt the user to enter the number of days (`n`) and then read `n` temperature values into a 1D array.
- Check if the values are **strictly increasing**, **strictly decreasing**, or neither.
- Print a message based on the trend type:
    - o `Temperatures are increasing.`
    - o `Temperatures are decreasing.`
    - o `No consistent trend.`
- Example function:

```
void analyzeTrend(int arr[], int n);
```

Example Input:

```
Enter number of days: 5
Temperatures: 20 22 25 28 30
```

Example Output:

```
Temperatures are increasing.
```

Question 5 - Reverse an Array

- Ask the user for the number of integers (`n`) and read them into a regular array.
- Write a function that uses **only pointer arithmetic** (no square brackets!) to print the elements in reverse order.
- Emphasize pointer incrementing/decrementing in your implementation.
- Example function:

```
void printReversed(int* arr, int size);
```

Question 6 - GPA Tracker

- Ask the user how many semesters they've completed.
- Dynamically allocate a float array using `malloc` to store GPA values.
- Read GPAs from the user and calculate their average using a separate function.
- Free the allocated memory after use.
- Example function:

```
float calculateAverage(float* gpas, int count);
```

Question 7 - Hospital Patient Record

- Define a struct `Patient` with `name`, `age`, and `temperature`.
- Read data for 4 patients into an array.
- Traverse the array to find the patient with the highest temperature.
- Print the name and temperature of that patient.

Struct Definition:

```
struct Patient {
    char name[30];
    int age;
    float temp;
};
```

Question 8 - Election Vote Counter

- Define a `Candidate` struct with `name` and `vote count`.
- Read information for 3 candidates.
- Find and display the name of the candidate with the most votes.
- Print all vote counts for transparency.

Struct Definition:

```
struct Candidate {
    char name[30];
    int votes;
};
```

Question 9- Product Discount

- Define a `Product` struct with `name` and `price`.
- Create a function that accepts a pointer to a product and applies a **10% discount** by modifying the `price` field.
- Call this function for at least 2 different products and show the before/after prices.

Struct & Function:

```
struct Product {
    char name[30];
    float price;
};

void applyDiscount(struct Product* p);
```

Question 10 - Export Students

- Define a `Student` struct with `name` and `grade`.
- Read and store information for 3 students into an array.
- Open a file `students.txt` using `fopen` in write mode.
- Write each student's name and grade to a new line in the file.
- Close the file and show a success message to the user.

Struct Definition:

```
struct Student {
    char name[30];
    float grade;
};
```

❖