

Q1)

Define a structure named **Student** with the following fields:

- **rollNo** (int)
- **name** (string)
- **marks** (float)

Create an array of 5 students. Prompt the user to enter details for each student and then display all the entered student records.

Expected Output:

```
Name: ST
Marks: 25

Enter details for student 2:
Roll Number: 02
Name: DF
Marks: 35

Enter details for student 3:
Roll Number: 03
Name: BN
Marks: 50

Enter details for student 4:
Roll Number: 04
Name: VC
Marks: 12

Enter details for student 5:
Roll Number: 05
Name: MN
Marks: 46

--- Student Records ---
```

Q2)

Modify the previous program so that it uses a **pointer to a structure** to display student details instead of directly accessing the array.

Q3)

Extend the student record program to allow the user to enter a roll number and search whether a student with that roll number exists. If found, display the student's details; otherwise, show a "*not found*" message.

Expected Output:

```
Enter details for student 3:
Roll Number: 3
Name: h
Marks: 85

Enter details for student 4:
Roll Number: 1
Name: t
Marks: 95

Enter details for student 5:
Roll Number: 7
Name: g
Marks: 66

Enter roll number to search: 5

Student Found:
Roll Number: 5
Name: f
Marks: 26

== Code Execution Successful ==
```

Q4)

Create a program that defines a structure **Student** with fields **rollNo**, **name**, and **marks**. Read details for 5 students and then calculate and display the average marks of the class.

Expected Output:

```
Output  
Clear  
Student 2:  
Roll Number: 102  
Name: Ruma  
Marks: 78  
  
Student 3:  
Roll Number: 103  
Name: Divi  
Marks: 92.5  
  
Student 4:  
Roll Number: 104  
Name: Sam  
Marks: 66  
  
Student 5:  
Roll Number: 105  
Name: Gesha  
Marks: 74.5  
  
Average Marks of the class: 79.3  
  
==== Code Execution Successful ===
```

Q5)

Create a structure **Student** with fields **rollNo**, **name**, and **marks**. Read data for 5 students and sort them in descending order of their marks.

Expected Output:

```
Output  
Clear  
--- Students sorted by marks (descending) ---  
Student 1:  
Roll Number: 103  
Name: Divi  
Marks: 92.5  
  
Student 2:  
Roll Number: 101  
Name: Anju  
Marks: 85.5  
  
Student 3:  
Roll Number: 102  
Name: Ruma  
Marks: 78  
  
Student 4:  
Roll Number: 105  
Name: Gesha  
Marks: 74.5  
  
Student 5:  
Roll Number: 104  
Name: Sam  
.
```

Q6)

Write a program using **new** and **delete** to dynamically allocate memory for an array of students based on user input (i.e., the user enters how many students they want to input).

Expected Output:

```
Output
Enter number of students: 3
Enter details for student 1:
Roll No: 01
Name: DF
Marks: 56

Enter details for student 2:
Roll No: 2
Name: GH
Marks: 63

Enter details for student 3:
Roll No: 3
Name: CX
Marks: 85

--- Student Details ---
Roll No: 1, Name: DF, Marks: 56
Roll No: 2, Name: GH, Marks: 63
Roll No: 3, Name: CX, Marks: 85

==== Code Execution Successful ====
```

Q7)

Create a structure for **Student** with fields **rollNo**, **name**, and **marks**.

- Accept details for 3 students
- Write them to a file
- Read the file and display the student records.

Q8)

Using dynamic memory, allow the user to add and remove student records.

- Allow addition of new students
- Remove a student by roll number
- Show the updated list.

Expected Output:

```
Output Clear
1. Add Student
2. Remove Student
3. Display All
4. Exit
Choose: 1
Enter Roll No: 2
Enter Name: df
Enter Marks: 63

1. Add Student
2. Remove Student
3. Display All
4. Exit
Choose: 3

--- Student List ---
Roll No: 2, Name: df, Marks: 63

1. Add Student
2. Remove Student
3. Display All
4. Exit
Choose: 2
Enter roll number to remove:
```

Q9)

Write a program where student details are passed to a function by reference. The function should update the student's marks by adding 5 bonus marks.

Q10)

Demonstrate the difference between:

- Array of Structures
  - Structure of Arrays
- By storing marks of students in both ways and printing them.

Expected Output:

```
Output Clear
Array of Structures:
1 Asha 90
2 Kaja 85
3 Niro 88

Structure of Arrays:
10 Vidhu 78.5
11 Eshan 80
12 Fariha 69.2

==== Code Execution Successful ====
```