Q1. Write a C++ program to manage employee records using a struct named Employee. The program should perform the following tasks:

1. Define the Employee struct:
   - An integer field id for the employee's ID.
   - A string field name for the employee's name.
   - A floating-point field salary for the employee's monthly salary.
2. Functions:
   - Implement a function createEmployee that returns an Employee object.
   - Implement a function findHighestSalary that takes the array of employees and the total number of employees as arguments and returns the employee with the highest salary.
3. Main Method:
   - Initialize an array of employees in the main function.
   - Use the createEmployee function to input data for each employee.
   - Use the findHighestSalary function to determine which employee has the highest salary.
   - Display the ID, name, and salary of the employee with the highest salary.

Q2. Write a C++ program to manage movie records using a struct named Movie. The program should perform the following tasks:

1. Define the Movie struct:
   - A string field name to store the name of the movie.
   - An integer field releaseYear to store the release year of the movie.
   - A floating-point field rating to store the rating of the movie (between 0.0 and 10.0).
2. Sort Movies by Rating:
   - Implement a sorting algorithm to sort the movies based on their rating in descending order. Don't use built in sorting algorithm.
   - After sorting, display the movie details in the sorted order.

Q3. Write a C++ program that manages a bank account using a struct named BankAccount. The program should allow you to perform deposits and withdrawals while ensuring that the balance never goes negative. The program should also display the account details (account number, name, and balance) after each transaction.

Requirements:

1. Define the BankAccount struct:
   - An integer field accountNumber to store the account number.
   - A string field name to store the account holder's name.
   - A floating-point field balance to store the account's balance.
2. Functions to deposit and withdraw:
   - Implement a function deposit that allows adding money to the balance.
   - Implement a function withdraw that allows withdrawing money, but only if the balance is sufficient (i.e., the balance cannot go negative).
3. Main Method:
   - Initialize a BankAccount instance.
   - Use the deposit and withdraw functions to perform transactions.

  - After each transaction, display the updated account details.

Q4. Write a C++ program that defines a struct Box to represent a 3D box with the following attributes:

- Length (length),
- Width (width),
- Height (height).

You need to overload the comparison operators (==, !=): To compare two Box objects based on their volume.

Q5. Write a C++ program to manage the enrollment of students in courses using nested structs. The program should allow you to define a Student struct that contains a list of courses the student is enrolled in, store marks for each course, and then display the student's information, including their enrolled courses and grades.

1. Define the Course struct:
   - An integer field courseID for the course's ID.
   - A string field courseName for the course's name.
   - An integer field credits for the number of credits the course is worth.
   - An array of integers marks[] to store marks obtained by the student in the course.
   - A function calculateGrade that calculates the grade for the course based on the total marks (we are assuming that total marks from which student is graded is 100), using the following grading scheme:

Average Marks	Grade
Average >= 90	A
80 <= Average < 90	B
70 <= Average < 80	C
60 <= Average < 70	D
Average < 60	F

2. Define the Student struct:
   - An integer field studentID for the student's ID.
   - A string field studentName for the student's name.
   - A nested Course struct to represent the courses the student is enrolled in. The Student struct should hold an array or list of Course structs.
   - A field numCourses representing the number of courses the student is enrolled in.
   - A function calculateGPA that calculates the overall GPA for the student based on their course grades.
     1. A-4
     2. B-3
     3. C-2
     4. D-1

        5. F-0

3. Functions:
    - addStudent: This functions to allow to add new student.
    - displayStudentDetails: Displays the student's ID, name, the list of courses they are enrolled in, the marks for each course, and the grade for each course.
4. Main Method:
    - Initialize an array or list of students.
    - Use the addStudent function to input details for each student and the courses they are enrolled in, including the marks for each course.
    - Use the displayStudentDetails function to display the details of each student, including their enrolled courses, marks, and grades.