C-DAC MUMBAI

Object Oriented Programming Using C++

Assigment-2

Q1. Create a class called Student with the following private data members:

1. name (string): to store the name of the student.
2. rollNumber (int): to store the roll number of the student.
3. marks (float): to store the marks obtained by the student.
4. grade (char): to store the grade calculated based on the marks.

Implement getter and setter member functions for each data member

Create a member function calculateGrade() that calculates the grade based on the following grading scale:

A: 90-100

B: 80-89

C: 70-79

D: 60-69

F: Below 60

Implement a menu-driven program in the main() function with the following options:

1. Accept Information
2. Display information
3. Calculate Grade
4. Exit the program.

Q2. Create a C++ program for a simple banking system. You need to implement a class called

1. BankAccount with the following data members:
2. accountNumber (int): The account number of the bank account.
3. accountHolderName (string): The name of the account holder.
4. balance (double): The current balance in the account.

The BankAccount class should have the following member functions:

1. Getter and Setter Methods:

2. deposit method: A method that allows the user to deposit money into the account. It should take an amount as a parameter and update the balance accordingly.

3. withdraw method: A method that allows the user to withdraw money from the account. It should take an amount as a parameter and update the balance. Make sure to check if there is sufficient balance before allowing the withdrawal.

4. displayAccountDetails method: A method that displays the account details (account number, account holder name, and balance).

Now, create a menu-driven program in the `main` function that allows the user to perform the following operations:

1. Deposit money into an existing account.
2. Withdraw money from an existing account.
3. Display the account details.
4. Exit the program.

Q3. Imagine you are tasked with creating a program to simulate a toll booth. The toll booth keeps
track of the number of vehicles that have passed through it and the total amount of money collected. You need to implement a class TollBooth with appropriate data members and member functions to accomplish this.

Here are the details for the TollBooth class:

1. **Data Members:**

- totalVehicles: An integer to keep track of the total number of vehicles that have passed through the toll booth.
- totalRevenue: A double to store the total revenue collected from tolls.

2. **Member Functions:**

1. void reset(): Resets both the totalVehicles and totalRevenue to zero.
2. void vehiclePayingToll(int vehicleType, double tollAmount): Accepts an integer vehicleType and a double tollAmount. The vehicleType represents the type of car (1 for standard car, 2 for truck, 3 for bus). The function should increment totalVehicles by 1 and add tollAmount to totalRevenue based on the car type.
3. int getTotalVehicles() : A getter method that returns the total number of vehicles passed through the booth.
4. double getTotalRevenue() : A getter method that returns the total revenue collected.

3. **Menu-Driven Program:**

Write a menu-driven program in main() that allows the user to interact with the TollBooth class:

- Display a menu with the following options:

1. Add a standard car and collect toll

2. Add a truck and collect toll
3. Add a bus and collect toll
4. Display total cars passed
5. Display total revenue collected
6. Reset booth statistics
7. Exit

- Implement the logic for each menu option using appropriate member functions of the TollBooth class.
- Continue displaying the menu until the user chooses to exit.
- Define a fixed toll amount for each type of car (e.g., $2 for standard cars, $5 for trucks, $10 for buses).

Q4. You are tasked with creating an Employee Payroll Management System in C++. Your program should allow the user to perform the following tasks through a menu-driven interface:

1. Add a new employee:
   - Create a class Employee with the following private data members:
   - int empID (Employee ID)
   - string empName (Employee Name)
   - double empSalary (Employee Salary)
   - Include appropriate getter and setter methods for these data members.
   - Ensure that the Employee ID is unique for each employee.
2. Calculate the gross salary for an employee:
   - Create a member function calculateGrossSalary in the Employee class.
   - The gross salary should be calculated based on the following rules:
   - If the employee's salary is less than or equal to 5000, add a 10% bonus.
   - If the employee's salary is greater than 5000 but less than or equal to 10000, add a 15% bonus.
   - If the employee's salary is greater than 10000, add a 20% bonus.
   - Display the gross salary for the chosen employee.
3. Display the employee details:
   - Create a member function displayEmployeeDetails in the Employee class to display all the details of an employee (ID, Name, Salary, and Gross Salary).
4. Update employee information:
   - Allow the user to update the employee's name and salary using setter methods.
5. Exit the program.