SCS 1310 - Object Oriented Programming
Practical 11

1. Create a C++ program for managing employee salaries.
   a. Define a base class **Employee** with a common function **calculateSalary()** that computes the salary.

   b. Then derive three classes, **FullTimeEmployee, PartTimeEmployee**, and **ContractEmployee** from the **Employee** base class.

   c. Implement the **calculateSalary()** function for each derived class based on the following scenarios:
      i. Kamal is a full-time employee with a monthly salary of Rs. 60,000.
      ii. Piyal is a part-time employee earning Rs. 2,000 per hour. She works 20 hours a week. iii. Damith is a contract employee who receives a fixed payment of Rs. 30,000 per month.

   d. Calculate the salaries for Kamal, Piyal, and Damith using the implemented **calculateSalary()** functions.

2.
   a.
      i. The class **Doctor** should have two integer members: doctor_specialist and doctor_id.

      ii. There will be two member functions: **getdata** and **putdata.**
         - The function getdata should get the input from the user: the name, age and doctor_specialist of the doctor.
         - The function putdata should print the name, age, doctor_specialis and the doctor_id of the doctor

   b.
      i. The class **Patient** should have two data members: admission_date and patient_id.

      ii. It has two member functions: **getdata** and **putdata**.
         - The function getdata should get the input from the user: the name, age, and admission_date.
         - The function putdata should print the name, age, sum of the marks and the patient_id of the patient.

   c. For each object being created of the Doctor or the Patient class, sequential ids should be assigned to them starting from 1.Solve this problem using virtual functions, constructors and static variables. You can create more data members if you want.

3.

a. Define a base class named Transaction with **transaction_ID** and **date** data members and the following virtual functions
    i. **recordTransaction()** for recording the transaction
    ii. **displayInfo()** for displaying transaction details

b. Create a derived class named **IncomeTransaction** from the **Transaction** class.
    i. Override the **recordTransaction()** function to record income transactions.
    ii. Override the displayInfo() function to display the details of income transactions, including the source of income and amount received.

c. Create another derived class named **ExpenseTransaction** from the Transaction class.
    i. Override the **recordTransaction()** function to record expense transactions. ii. Override the **displayInfo()** function to display the details of expense transactions

d. Your implementation should allow the user to input information for both income and expense transactions and display their details.

4. Write the C++ code to implement the library management system based on the requirements provided above.

    a. Define a class named **Book** to represent a book in the library. Include private data members for the book's **title, author, ISBN,** and **availability status** (whether the book is currently available or checked out).

    b. Implement public member functions to:
        - Set and get the book's title, author, and ISBN.
        - Check out and return the book.
        - Display the book's information.

    c. Define a class named **Patron** to represent a library patron. Include private data members for the patron's **name, library card number, and a list of borrowed books**.

    d. Implement public member functions to:
        - Set and get the patron's name and library card number.
        - Borrow and return a book.
        - Display the patron's information, including the books they have borrowed.

    e. Ensure that the implementation follows the principles of encapsulation, with data members properly encapsulated and accessed through public member functions.