Lab Objective

1. **Write** the definition of the super class and extend it to create multiple subclasses.
2. **Write** codes to implement polymorphism.

Lab Activities

A. Defining the Superclass

Design a class named **Account** that contains:

- A private **int** data field named **id** for the account (default **0**).
- A private **double** data field named **balance** for the account (default **0.0**).
- A private **double** data field named **annualInterestRate** that stores the current interest rate (default **0.0**).
- A no-arg constructor that creates a default account.
- A constructor that creates an account with the specified id, initial balance and annual interest rate.
- The accessor and mutator methods for **id**, **balance**, and **annualInterestRate**.
- A method named **getMonthlyInterestRate()** that returns the monthly interest rate. **monthlyInterestRate** is **annualInterestRate / 12**. Note that **annualInterestRate** is a percentage e.g., like 4.5%. You need to divide it by 100
- The method **getMonthlyInterestAmount()** is to return monthly interest amount, not the interest rate. Monthly interest amount is **balance * monthlyInterestRate**.
- A method named **withdraws** that withdraws a specified amount from the account.
- A method named **deposit** that deposits a specified amount to the account

Write a test program that creates an **Account** object with an account ID of 1122, a balance of $20,000, and an annual interest rate of 4.5%. Use the **withdraw** method to withdraw $2,500, use the **deposit** method to deposit $3,000, and print the ID, balance and the monthly interest of the Account.

B. Creating the Subclasses

Create two subclasses for checking and saving accounts named as **Checking Account** and **Savings Account**.

- A checking account has an overdraft limit which is double type variable.
- A savings account has issued with a credit card automatically. It holds the 16-digit card number. Savings Account class must have a method getCreditBalance that returns a credit balance which is three times of the current balance in the account.

C. Defining an Array/ArrayList of Account type objects in Main method

To understand polymorphism, you need to define an array list of Account type objects based on userprovided option. Your main () method must display the following first:

```
              Press (1) for creating a Checking Account
Press (2) for creating a Savings Account
```

You must create at least 4 account type objects in this manner and perform one deposit and one withdraws operation for each account.

Afterwards, print the followings for each account using the concept of Polymorphism. You must not use any toString() method.

For a Checking Account: This is a Checking Account Account ID: Current Balance: Annual Interest Rate: Monthly Interest Amount: Overdraft Limit:	For a Savings Account: This is a Savings Account Account ID: Current Balance: Annual Interest Rate: Monthly Interest Amount: Credit Card Number: Credit Balance:

Questions:

i. Create a class with a method that prints "This is parent class" and its subclass with another method that prints "This is child class". Now, create an object for each of the class and call
1 - method of parent class by object of parent class
2 - method of child class by object of child class
3 - method of parent class by object of child class

ii. In the above example, declare the method of the parent class as private and then repeat the first two operations (You will get error in the third).

iii. Create a class named 'Member' having the following members: Data members
1 - Name
2 - Age
3 - Phone number
4 - Address
5 - Salary
It also has a method named 'printSalary' which prints the salary of the members.
Two classes 'Employee' and 'Manager' inherits the 'Member' class. The 'Employee' and 'Manager' classes have data members 'specialization' and 'department' respectively. Now, assign name, age, phone number, address and salary to an employee and a manager by making an object of both of these classes and print the same.

iv. Create a class named 'Rectangle' with two data members 'length' and 'breadth' and two methods to print the area and perimeter of the rectangle respectively. Its constructor having parameters for length and breadth is used to initialize length and breadth of the rectangle. Let class 'Square' inherit the 'Rectangle' class with its constructor having a parameter for its side (suppose s) calling the constructor of its parent class as 'super(s,s)'. Print the area and perimeter of a rectangle and a square.

v. Now repeat the above example to print the area of 10 squares. Hint-Use array of objects

vi. Create a class named 'Shape' with a method to print "This is This is shape". Then create two other classes named 'Rectangle', 'Circle' inheriting the Shape class, both having a method to print "This is rectangular shape" and "This is circular shape" respectively. Create a subclass 'Square' of 'Rectangle' having a method to print "Square is a rectangle". Now call the method of 'Shape' and 'Rectangle' class by the object of 'Square' class.