**Lab 7 Activities**

**TASK 1:**

Create a Java program that implements a class hierarchy for a library system. The hierarchy should include the following classes:

* Item: a base class that represents an item in the library and has attributes such as title, author, and year. It also defines a method toString that returns a string representation of the item's attributes.
* Book: a subclass of Item that represents a book and has additional attributes such as publisher and ISBN. It also overrides the toString method to include the book's publisher and ISBN in the string representation.
* Magazine: a subclass of Item that represents a magazine and has additional attributes such as publisher and issueNumber. It also overrides the toString method to include the magazine's publisher and issue number in the string representation.
* DVD: a subclass of Item that represents a DVD and has additional attributes such as director and length. It also overrides the toString method to include the DVD's director and length in the string representation.

Your program should include appropriate constructors for each class.

It should create instances of each subclass and demonstrate how to call their methods to retrieve their attributes and behavior, including the overridden toString method.

**Task 2:**

Create a Java program that implements a class hierarchy for a university system. The hierarchy should include the following classes:

* Person: a base class that represents a person and has attributes such as name, email, and phone. It also defines a method toString that returns a string representation of the persons attributes.
* Student: a subclass of Person that represents a student and has attributes such as studentId and major.
* It also defines a method getGPA that returns the students grade point average.
* Faculty: a subclass of Person that represents a faculty member and has attributes such as facultyId and department. It also defines a method getRank that returns the faculty members rank (e.g., professor, assistant professor, etc.).
* Staff: a subclass of Person that represents a staff member and has attributes such as staffId and jobTitle.
* It also defines a method getSalary that returns the staff members salary.

Your program should include appropriate constructors for each class.

**Task 3:**

Create a class named Robot that will input:

* the x and y coordinates of a Robot and the direction in which he wants to move using parameterized constructor. (direction could be E,W,N,S)
* Create a method to display the intial position of the robot.

Create another class named Moving Robot inherited from robot

* This class have a function named moveRobot; function will take steps to move as argument and move robot in that direction.

For example, if initially the direction = N and Y = 2, and user entered the steps=3, so after movement the updated coordinates are; Y = 5. (Since moving in North will update the +Y in Quadrant System)

* Create a display method to show the updated X and Y coordinates.

**Task 4:**

Crete a polymorphic banking application.

The application contains a class named **Accounts**. Two classes named saving account and Checking account inherit it. The account class have:

* an attribute named balance
* a member function named debit to withdraw amount from the account.
* a member function named credit function to deposit a particular amount.
* A getBalance function to show current balance.

Create a class named SavingAccount that contains:

* An attribute named timeSpan (to show the time passed over money saved in the account)
* A calculateIntereset method to calculate interest over the current balance amount
  + Formula for that is Interest = (current Balance) \* InterestRate \* timespan
  + Consider whatever interestRate you want
* Override the credit method to add the interest to account balance.

After transaction processing print the updated account balance using getBalance function.

**Task 5:**

Create a Java program that implements a class named "Sorting" with the following methods:

* sort (int[] array) sorts the given integer array in ascending order.
* sort(String[] array) sorts the given string array in ascending order
* sort (int[] array, boolean descending) sorts the given integer array in either ascending or descending order, depending on the value of the boolean parameter.
* sort(String[] array, boolean descending) sorts the given string array in either ascending or descending order, depending on the value of the boolean parameter.

Create an instance of the "Sorting" class, populate some arrays with random values, and execute all four "sort" methods with appropriate arguments.