DATE:29.12.23 DAY4- JAVA- QUIZ-1

1. **Create a class named 'Member' having the following members: Data members 1 - Name 2 Phone number 4 - Address 5 - Salary It also has a method named 'printSalary' which prints t of the members.**

```
class Member {
    String name;
    int age;
    String phoneNumber;
    String address;
    double salary;

    Member(String name, int age, String phoneNumber, String address, double salary) {
        this.name = name;
        this.age = age;
        this.phoneNumber = phoneNumber;
        this.address = address;
        this.salary = salary;
    }

    void printSalary() {
        System.out.println("Salary: " + salary);
    }
}

class Employee extends Member {
    String specialization;

    Employee(String name, int age, String phoneNumber, String address, double salary, String specialization) {
        super(name, age, phoneNumber, address, salary);
        this.specialization = specialization;
    }
}

class Manager extends Member {
    String department;

    Manager(String name, int age, String phoneNumber, String address, double salary, String department) {
        super(name, age, phoneNumber, address, salary);
        this.department = department;
    }
}

public class Main {
    public static void main(String[] args) {
        Employee employee = new Employee("John Doe", 30, "123-456-7890", "123 Main St", 50000, "IT");
        Manager manager = new Manager("Jane Smith", 35, "987-654-3210", "456 Oak St", 80000, "HR");

        employee.printSalary();
        manager.printSalary();
    }
}
```

2. You are developing a banking application in Java. Design a class hierarchy that represents account types such as SavingsAccount, CheckingAccount, and LoanAccount. Each account should h functionality like deposit, withdraw, and check balance.

```
interface Account {
    void deposit(double amount);
    void withdraw(double amount);
    double checkBalance();
}

class SavingsAccount implements Account {
    private double balance;

    @Override
    public void deposit(double amount) {
        balance += amount;
    }

    @Override
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
        } else {
            System.out.println("Insufficient funds");
        }
    }

    @Override
    public double checkBalance() {
        return balance;
    }
}

class CheckingAccount implements Account {
    private double balance;

    @Override
    public void deposit(double amount) {
        balance += amount;
    }

    @Override
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
        } else {
            System.out.println("Insufficient funds");
```

```
        }
    }

    @Override
    public double checkBalance() {
        return balance;
    }
}

class LoanAccount implements Account {
    private double balance;

    @Override
    public void deposit(double amount) {
        balance += amount;
    }

    @Override
    public void withdraw(double amount) {
        // Additional logic for loan account withdrawal
    }

    @Override
    public double checkBalance() {
        return balance;
    }
}
```

3.You are tasked with designing a university enrollment system in Java. Implement a class hierarc
includes a base class Person and two subclasses, Student and Professor and a Course class.

```
import java.util.ArrayList;
import java.util.List;

class Person {
    String name;
    int age;
    String address;

    Person(String name, int age, String address) {
        this.name = name;
        this.age = age;
        this.address = address;
    }
}

class Student extends Person {
    List<String> completedPrerequisites;

    Student(String name, int age, String address) {
        super(name, age, address);
        completedPrerequisites = new ArrayList<>();
    }
```

```
    void completePrerequisite(String prerequisite) {
        completedPrerequisites.add(prerequisite);
    }
}

class Professor extends Person {
    // Professor-specific attributes
}

class Course {
    String courseName;
    List<String> prerequisites;
    List<Student> enrolledStudents;

    Course(String courseName) {
        this.courseName = courseName;
        prerequisites = new ArrayList<>();
        enrolledStudents = new ArrayList<>();
    }

    void addPrerequisite(String prerequisite) {
        prerequisites.add(prerequisite);
    }

    void enrollStudent(Student student) {
        if (student.completedPrerequisites.containsAll(prerequisites)) {
            enrolledStudents.add(student);
            System.out.println(student.name + " enrolled in " + courseName);
        } else {
            System.out.println(student.name + " does not meet prerequisites for " + courseName);
        }
    }

    void displayEnrolledStudents() {
        System.out.println("Enrolled students in " + courseName + ":");
        for (Student student : enrolledStudents) {
            System.out.println("Name: " + student.name + ", Age: " + student.age + ", Address: " + student.address);
        }
    }
}
```