**1. 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**.

**CODE:**

import java.util.Scanner;

class Member {

    String name;

    int age;

    String phoneNumber;

    String address;

    double salary;

    public 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;

    }

    public void printSalary() {

        System.out.println("Salary: " + salary);

    }

}

class Employee extends Member {

    String specialization;

    public 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;

    public 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) {

        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter Employee Details:");

        System.out.print("Name: ");

        String employeeName = scanner.nextLine();

        System.out.print("Age: ");

        int employeeAge = scanner.nextInt();

        scanner.nextLine();

        System.out.print("Phone Number: ");

        String employeePhoneNumber = scanner.nextLine();

        System.out.print("Address: ");

        String employeeAddress = scanner.nextLine();

        System.out.print("Salary: ");

        double employeeSalary = scanner.nextDouble();

        scanner.nextLine();

        System.out.print("Specialization: ");

        String employeeSpecialization = scanner.nextLine();

        Employee employee1 = new Employee(employeeName, employeeAge, employeePhoneNumber, employeeAddress, employeeSalary, employeeSpecialization);

        System.out.println("\nEnter Manager Details:");

        System.out.print("Name: ");

        String managerName = scanner.nextLine();

        System.out.print("Age: ");

        int managerAge = scanner.nextInt();

        scanner.nextLine();

        System.out.print("Phone Number: ");

        String managerPhoneNumber = scanner.nextLine();

        System.out.print("Address: ");

        String managerAddress = scanner.nextLine();

        System.out.print("Salary: ");

        double managerSalary = scanner.nextDouble();

        scanner.nextLine();

        System.out.print("Department: ");

        String managerDepartment = scanner.nextLine();

        Manager manager1 = new Manager(managerName, managerAge, managerPhoneNumber, managerAddress, managerSalary, managerDepartment);

        System.out.println("\nEmployee Details:");

        System.out.println("Name: " + employee1.name);

        System.out.println("Age: " + employee1.age);

        System.out.println("Phone Number: " + employee1.phoneNumber);

        System.out.println("Address: " + employee1.address);

        employee1.printSalary();

        System.out.println("Specialization: " + employee1.specialization);

        System.out.println("\nManager Details:");

        System.out.println("Name: " + manager1.name);

        System.out.println("Age: " + manager1.age);

        System.out.println("Phone Number: " + manager1.phoneNumber);

        System.out.println("Address: " + manager1.address);

        manager1.printSalary();

        System.out.println("Department: " + manager1.department);

    }

}

**2. You are developing a banking application in Java. Design a class hierarchy that represents different account types such as SavingsAccount, CheckingAccount, and LoanAccount.**

**Each account should have basic functionality like deposit, withdraw, and check balance.**

**Ensure that your design follows appropriate use of interfaces and inheritance.**

**CODE:**

import java.util.Scanner;

interface Account {

    void deposit(double amount);

    void withdraw(double amount);

    double checkBalance();

}

abstract class BankAccount implements Account {

    String accountNumber;

    double balance;

    public BankAccount(String accountNumber, double initialBalance) {

        this.accountNumber = accountNumber;

        this.balance = initialBalance;

    }

    @Override

    public void deposit(double amount) {

        balance += amount;

        System.out.println("Deposited: $" + amount);

    }

    @Override

    public void withdraw(double amount) {

        if (amount <= balance) {

            balance -= amount;

            System.out.println("Withdrawn: $" + amount);

        } else {

            System.out.println("Insufficient funds!");

        }

    }

    @Override

    public double checkBalance() {

        return balance;

    }

}

class SavingsAccount extends BankAccount {

    double interestRate;

    public SavingsAccount(String accountNumber, double initialBalance, double interestRate) {

        super(accountNumber, initialBalance);

        this.interestRate = interestRate;

    }

    public void addInterest() {

        double interest = balance \* interestRate;

        balance += interest;

        System.out.println("Interest added: $" + interest);

    }

}

class CheckingAccount extends BankAccount {

    double overdraftLimit;

    public CheckingAccount(String accountNumber, double initialBalance, double overdraftLimit) {

        super(accountNumber, initialBalance);

        this.overdraftLimit = overdraftLimit;

    }

    @Override

    public void withdraw(double amount) {

        if (amount <= balance + overdraftLimit) {

            balance -= amount;

            System.out.println("Withdrawn: $" + amount);

        } else {

            System.out.println("Exceeded overdraft limit!");

        }

    }

}

class LoanAccount extends BankAccount {

    double interestRate;

    public LoanAccount(String accountNumber, double initialBalance, double interestRate) {

        super(accountNumber, initialBalance);

        this.interestRate = interestRate;

    }

    public void chargeInterest() {

        double interest = balance \* interestRate;

        balance += interest;

        System.out.println("Interest charged: $" + interest);

    }

}

public class Main {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.println("Creating a Savings Account:");

        System.out.print("Enter account number: ");

        String savingsAccountNumber = scanner.nextLine();

        System.out.print("Enter initial balance: $");

        double savingsInitialBalance = scanner.nextDouble();

        scanner.nextLine();

        System.out.print("Enter interest rate: ");

        double savingsInterestRate = scanner.nextDouble();

        scanner.nextLine();

        SavingsAccount savingsAccount = new SavingsAccount(savingsAccountNumber, savingsInitialBalance, savingsInterestRate);

        System.out.println("\nCreating a Checking Account:");

        System.out.print("Enter account number: ");

        String checkingAccountNumber = scanner.nextLine();

        System.out.print("Enter initial balance: $");

        double checkingInitialBalance = scanner.nextDouble();

        scanner.nextLine();

        System.out.print("Enter overdraft limit: $");

        double overdraftLimit = scanner.nextDouble();

        scanner.nextLine();

        CheckingAccount checkingAccount = new CheckingAccount(checkingAccountNumber, checkingInitialBalance, overdraftLimit);

        System.out.println("\nCreating a Loan Account:");

        System.out.print("Enter account number: ");

        String loanAccountNumber = scanner.nextLine();

        System.out.print("Enter initial loan amount: $");

        double loanInitialAmount = scanner.nextDouble();

        scanner.nextLine();

        System.out.print("Enter interest rate: ");

        double loanInterestRate = scanner.nextDouble();

        scanner.nextLine();

        LoanAccount loanAccount = new LoanAccount(loanAccountNumber, -loanInitialAmount, loanInterestRate);

        System.out.println("\nInteracting with accounts:");

        System.out.print("Enter deposit amount for Savings Account: $");

        double depositAmount = scanner.nextDouble();

        scanner.nextLine();

        savingsAccount.deposit(depositAmount);

        System.out.print("Enter withdrawal amount for Checking Account: $");

        double withdrawAmount = scanner.nextDouble();

        scanner.nextLine();

        checkingAccount.withdraw(withdrawAmount);

        System.out.println("\nSavings Account Balance: $" + savingsAccount.checkBalance());

        savingsAccount.addInterest();

        System.out.println("Updated Savings Account Balance: $" + savingsAccount.checkBalance());

        System.out.println("\nChecking Account Balance: $" + checkingAccount.checkBalance());

        System.out.print("Enter withdrawal amount for Loan Account: $");

        double loanWithdrawAmount = scanner.nextDouble();

        scanner.nextLine();

        loanAccount.withdraw(loanWithdrawAmount);

        System.out.println("\nLoan Account Balance: $" + loanAccount.checkBalance());

        loanAccount.chargeInterest();

        System.out.println("Updated Loan Account Balance: $" + loanAccount.checkBalance());

    }

}

**3. You are tasked with designing a university enrollment system in Java. Implement a class hierarchy that includes a base class Person and two subclasses, Student and Professor and a Course class. Each class should have the necessary attributes. Each course should have a list of prerequisites and enrolled students.**

**Your tasks are as follows:**

**i) Students should only be enrolled if they have completed all the required prerequisites. In the course class, include logic for enrolling students.**

**ii) Display enrolled students in a particular with relevant information.**

**CODE:**

import java.util.ArrayList;

import java.util.Scanner;

class Person {

    String name;

    int age;

    public Person(String name, int age) {

        this.name = name;

        this.age = age;

    }

    @Override

    public String toString() {

        return "Name: " + name + ", Age: " + age;

    }

}

class Student extends Person {

    int studentId;

    public Student(String name, int age, int studentId) {

        super(name, age);

        this.studentId = studentId;

    }

    @Override

    public String toString() {

        return "Student ID: " + studentId + ", " + super.toString();

    }

}

class Professor extends Person {

    String department;

    public Professor(String name, int age, String department) {

        super(name, age);

        this.department = department;

    }

    @Override

    public String toString() {

        return "Department: " + department + ", " + super.toString();

    }

}

class Course {

    String courseName;

    ArrayList<String> prerequisites;

    ArrayList<Student> enrolledStudents;

    public Course(String courseName, ArrayList<String> prerequisites) {

        this.courseName = courseName;

        this.prerequisites = prerequisites;

        this.enrolledStudents = new ArrayList<>();

    }

    public void enrollStudent(Student student) {

        if (checkPrerequisites(student)) {

            enrolledStudents.add(student);

            System.out.println("Enrolled " + student.name + " in " + courseName);

        } else {

            System.out.println("Student " + student.name + " cannot be enrolled in " + courseName

                    + ". Prerequisites not met.");

        }

    }

    private boolean checkPrerequisites(Student student) {

        for (String prerequisite : prerequisites) {

            boolean hasCompleted = false;

            for (Student enrolledStudent : student.enrolledStudents) {

                if (enrolledStudent.name.equals(prerequisite)) {

                    hasCompleted = true;

                    break;

                }

            }

            if (!hasCompleted) {

                return false;

            }

        }

        return true;

    }

    public void displayEnrolledStudents() {

        System.out.println("Enrolled Students in " + courseName + ":");

        for (Student student : enrolledStudents) {

            System.out.println(student);

        }

    }

}

public class UniversityEnrollmentSystem {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter student details:");

        Student student1 = createStudent(scanner);

        Student student2 = createStudent(scanner);

        System.out.println("\nEnter professor details:");

        Professor professor = createProfessor(scanner);

        System.out.println("\nEnter course details:");

        Course course1 = createCourse(scanner);

        Course course2 = createCourse(scanner);

        System.out.println("\nEnrolling students in courses:");

        course1.enrollStudent(student1);

        course1.enrollStudent(student2);

        course2.enrollStudent(student1);

        System.out.println("\nDisplaying enrolled students:");

        course1.displayEnrolledStudents();

        course2.displayEnrolledStudents();

    }

    private static Student createStudent(Scanner scanner) {

        System.out.print("Enter student name: ");

        String name = scanner.nextLine();

        System.out.print("Enter student age: ");

        int age = scanner.nextInt();

        System.out.print("Enter student ID: ");

        int studentId = scanner.nextInt();

        scanner.nextLine();

        return new Student(name, age, studentId);

    }

    private static Professor createProfessor(Scanner scanner) {

        System.out.print("Enter professor name: ");

        String name = scanner.nextLine();

        System.out.print("Enter professor age: ");

        int age = scanner.nextInt();

        scanner.nextLine();

        System.out.print("Enter professor department: ");

        String department = scanner.nextLine();

        return new Professor(name, age, department);

    }

    private static Course createCourse(Scanner scanner) {

        System.out.print("Enter course name: ");

        String courseName = scanner.nextLine();

        System.out.print("Enter number of prerequisites: ");

        int numPrerequisites = scanner.nextInt();

        scanner.nextLine();

        ArrayList<String> prerequisites = new ArrayList<>();

        for (int i = 0; i < numPrerequisites; i++) {

            System.out.print("Enter prerequisite " + (i + 1) + ": ");

            String prerequisite = scanner.nextLine();

            prerequisites.add(prerequisite);

        }

        return new Course(courseName, prerequisites);

    }

}