Advanced Programming Language

Assignment – 3

Chinmay Dorge

LCS2020022

Q1) Create a student class that contains three variables (name, roll number and phone number) and two methods for taking the inputs and display the results of those filed. Next class teacher and staff inherit the student class. The teach class contains another two variables (name and area of teaching) and two methods for taking the inputs and display the results of those filed. The staff class contains another two variables (name and work) and two methods for taking the inputs and display the results of those filed. Further, teacher class contains three further classes, namely science, arts and commerce and each of the class contain a variable number of student and further contains two methods for taking the inputs and display the results of this filed.

Code: Student.java

```
import java.util.Scanner;
public class Student{
    Scanner sc = new Scanner(System.in);
    protected String name;
    private int rollNumber;
    private long mobileNumber;


    public void takeInput(){
        System.out.print("Enter Student's name: ");
        name = sc.nextLine();
        System.out.print("Enter Student's roll number: ");
        rollNumber = sc.nextInt();
        System.out.print("Enter Student's mobile number: ");
        mobileNumber = sc.nextLong();
    }
```

```
    public void displayData(){
        System.out.println("Student's name: " + name);
        System.out.println("Student's roll number: " + rollNumber);
        System.out.println("Student's mobile number: " + mobileNumber);
    }

}
```

Staff.java

```
import java.util.Scanner;
public class Staff extends Student{
    Scanner sc = new Scanner(System.in);
    private String work;

    public void takeInput(){
        System.out.print("Enter Staff's name: ");
        name = sc.nextLine();
        System.out.print("Enter Staff's work: ");
        work = sc.nextLine();
    }
    public void displayData(){
        System.out.println("Staff's name: " + name);
        System.out.println("Staff's work: " + work);
    }
}
```

Teacher.java

```
import java.util.Scanner;
public class Teacher extends Student{
    Scanner sc = new Scanner(System.in);
    private String areaOfTeaching;

    public void takeInput(){
        System.out.print("Enter Teacher's name: ");
        name = sc.nextLine();
        System.out.print("Enter Teacher's area of teaching: ");
        areaOfTeaching = sc.nextLine();
    }

    public void displayData(){
        System.out.println("Teacher's name " + name);
        System.out.println("Teacher's area of teaching " + areaOfTeaching);
    }

    public class Science{
        private int numberOfStudents;
        void takeInput(){
            System.out.println("Enter number of students for Science
stream: ");
            numberOfStudents = sc.nextInt();
        }
        void displayData(){
            System.out.println("Number of students for Science = " +
numberOfStudents);
```

```
        }
    }

    public class Arts{
        private int numberOfStudents;
        void takeInput(){
            System.out.println("Enter number of students for Arts stream:
");
            numberOfStudents = sc.nextInt();
        }
        void displayData(){
            System.out.println("Number of students for Arts = " +
numberOfStudents);
        }
    }

    public class Commerce{
        private int numberOfStudents;
        void takeInput(){
            System.out.println("Enter number of students for Commerce
stream: ");
            numberOfStudents = sc.nextInt();
        }
        void displayData(){
            System.out.println("Number of students for Commerce = " +
numberOfStudents);
        }
    }
}
```

Q1.java

```
public class Q1 {
    public static void main(String[] arg){
        Student s1 = new Student();
        s1.takeInput();
        s1.displayData();
        System.out.println();

        Student s2 = new Student();
        s2.takeInput();
        s2.displayData();
        System.out.println();

        Teacher t1 = new Teacher();
        t1.takeInput();
        t1.displayData();
        System.out.println();

        Teacher t2 = new Teacher();
        t2.takeInput();
        t2.displayData();
        System.out.println();

        Teacher.Science sci = t1.new Science();
        sci.takeInput();
        sci.displayData();

        Teacher.Commerce com = t1.new Commerce();
```

```
        com.takeInput();
        com.displayData();

        Teacher.Arts art = t2.new Arts();
        art.takeInput();
        art.displayData();

        Staff st1 = new Staff();
        st1.takeInput();
        st1.displayData();
        System.out.println();

    }
}
```

Output:

```
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3> cd Q1
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q1> javac Q1.java
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q1> java Q1
Enter Student's name: Dhruv
Enter Student's roll number: 62
Enter Student's mobile number: 45678964323
Student's name: Dhruv
Student's roll number: 62
Student's mobile number: 45678964323

Enter Student's name: Chinmay
Enter Student's roll number: 22
Enter Student's mobile number: 9876543210
Student's name: Chinmay
Student's roll number: 22
Student's mobile number: 9876543210

Enter Teacher's name: Saurabh S
Enter Teacher's area of teaching: Science
Teacher's name Saurabh S
Teacher's area of teaching Science

Enter Teacher's name: Niharika Anand
Enter Teacher's area of teaching: Commerce
Teacher's name Niharika Anand
Teacher's area of teaching Commerce
```

```
Enter number of students for Science stream:
1
Number of students for Science = 1
Enter number of students for Commerce stream:
2
Number of students for Commerce = 2
Enter number of students for Arts stream:
3
Number of students for Arts = 3
Enter Staff's name: Elon Musk
Enter Staff's work: Security
Staff's name: Elon Musk
Staff's work: Security
```

Q2) Create a java class namely arithmetic which contains four abstract methods namely addition, subtraction, multiplication and division that perform arithmetic operation by taking two numbers as inputs. Further, write four non-abstract classes that defines the functionality of the above-mentioned operations and display the results of those functions by creating another main class.

Code: arithmetic.java

```java
abstract class arithmeticOperations{
    abstract public double addition(double a ,double b);
    abstract public double subtraction(double a ,double b);
    abstract public double multiplication(double a ,double b);
    abstract public double division(double a ,double b);
}
public class arithmetic extends arithmeticOperations{
    public double addition(double a , double b){
        return a+b;
    }
    public double subtraction(double a , double b){
        return a-b;
    }
    public double multiplication(double a , double b){
        return a*b;
```

```
    }
    public double division(double a , double b){
        return a/b;
    }
}
```

main.java

```
import java.util.Scanner;
public class main {
    public static void main(String[] arg){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter two numbers: ");
        double a = sc.nextDouble();
        double b = sc.nextDouble();

        arithmetic ans = new arithmetic();
        System.out.println("Addition = " + ans.addition(a,b));
        System.out.println("Subtraction = " + ans.subtraction(a,b));
        System.out.println("Multiplication = " + ans.multiplication(a,b));
        System.out.println("Division = " + ans.division(a,b));
    }
}
```

Output:

```
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3> cd Q2
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q2> javac main.java
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q2> java main
Enter two numbers: 3 4
Addition = 7.0
Subtraction = -1.0
Multiplication = 12.0
Division = 0.75
```

Q3) Write a specific scenario and then write a java program on that scenario like Question 1 and 2 that represent multiple inheritance can possible to implement using interface. Similarly, write a specific scenario and then write a java program on that scenario like Question 1 and 2 that represent interface extends another interface instead of implements. (Do

not use the examples which have been discussed in the lectures).

1. A scenario where multiple inheritance can be possible to implement using interface. A remote toy car is both battery operated and remote controlled. So, it will contain properties of both the classes and inherit their properties via multiple inheritance.

Code: part1.java

```
interface batteryOperatedToy{
    void checkBattery();
    void repairBattery();
}

interface remoteOperatedToys{
    void checkRemote();
    void repairRemote();
}

class toyCar implements batteryOperatedToy , remoteOperatedToys {
    toyCar(){
        System.out.println("Toy car has been created");
    }

    @java.lang.Override
    public void checkBattery() {
        System.out.println("Checking battery...");
        System.out.println();
    }

    @java.lang.Override
    public void repairBattery(){
        System.out.println("Repairing battery...");
        System.out.println("...Successfully repaired");
        System.out.println();
    }

    @java.lang.Override
    public void checkRemote(){
        System.out.println("Checking remote...");
        System.out.println();
    }

    @java.lang.Override
    public void repairRemote() {
        System.out.println("Repairing remote...");
        System.out.println("...Successfully repaired");
        System.out.println();
    }
}
```

```java
public class part1 {
    public static void main(String[] args){
        toyCar obj = new toyCar();
        obj.checkBattery();
        obj.repairBattery();
        obj.checkRemote();
        obj.repairRemote();
    }
}
```

Output:

```
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3> cd Q3
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q3> javac part1.java
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q3> java part1
Toy car has been created
Checking battery...

Repairing battery...
...Successfully repaired

Checking remote...

Repairing remote...
...Successfully repaired
```

2. A scenario that represents interface extends another interface. A smartphone can perform the same functions as a normal mobile. So smartPhone interface extends mobile interface instead of implementing.

part2.java

```java
interface mobile{
    void makeCall();
    void checkTime();
}

interface smartPhone extends mobile{
    void playGames();
    void watchMovie();
}

class myPhone implements smartPhone{

    @java.lang.Override
    public void makeCall() {
```

```
        System.out.println("Making a call ...");
        System.out.println();
    }

    @java.lang.Override
    public void checkTime() {
        System.out.println("Checking Time ...");
        System.out.println();
    }

    @java.lang.Override
    public void playGames() {
        System.out.println("Playing games ...");
        System.out.println();
    }

    @java.lang.Override
    public void watchMovie() {
        System.out.println("Playing movie ...");
        System.out.println();
    }
}

public class part2{
    public static void main(String[] args){
        myPhone obj = new myPhone();
        obj.makeCall();
        obj.checkTime();
        obj.playGames();
        obj.watchMovie();

    }
}
```

Output:

```
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q3> javac part2.java
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q3> java part2
Making a call ...

Checking Time ...

Playing games ...

Playing movie ...
```

Q4) Write a recursive definition in java that converts binary number to decimal number.

Code: Main.java

```java
import java.util.Scanner;
public class Main {
    static long binaryToDecimal(String bin , int i){
        int n = bin.length();
        if (i == n-1)
            return bin.charAt(i) - '0';

        return ((bin.charAt(i) - '0') << (n-i-1)) +
                binaryToDecimal(bin, i+1);
    }

    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter binary number: ");
        String input = sc.nextLine();
        System.out.println("Decimal = " + binaryToDecimal(input,0));
    }
}
```

Output:

```
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q4> javac Main.java
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q4> java Main
Enter binary number: 1110
Decimal = 14
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q4> java Main
Enter binary number: 1111
Decimal = 15
```

Q5) Write a specific scenario and then write a java program on that scenario like Question 1 and 2 that contains both method overloading and method overriding in the same program.

A scenario that contains both method overloading and method overriding in the same program. Football class inherits Sport class and play() function has been both overloaded and overridden.

Code: Sport.java

```java
public class Sport{
    public void play(){
        System.out.println("Playing sport");
    }
}
```

Football.java

```java
public class Football extends Sport{
    @java.lang.Override
    public void play() {
        System.out.println("Playing Football");
    }

    public void play(int a , int b){
        System.out.println("Playing football with score " + a + " - " + b
);
    }
}
```

Main.java

```java
public class Main {
    public static void main(String[] args){
        Football f = new Football();
        f.play();         // overriding
        f.play(3,4);      // overloading
    }
}
```

Output:

```
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3> cd Q5
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q5> javac Main.java
PS C:\Users\hp\Desktop\CLG\Third Sem\APL\Assignment-3\Q5> java Main
Playing Football
Playing football with score 3 - 4
```

Note: The code has also been sent with the zip file and is available on GitHub. Link for the repository:

https://github.com/Chinmay-Dorge/Advanced-Programming-Assignments