LAB ACTIVITY 5

Name : Sarvagya Gupta
Sap Id : 500083195
Roll No. : R2142201047

TITLE: Interface

1. Write a program to create interface named test. In this interface the member function is square. Implement this interface in arithmetic class. Create one new class called ToTestInt. In this class use the object of arithmetic class.

```java

interface test{
    void square();
}

class arithmetic implements test{
    public void square(){
        System.out.println("Square");
    }
}

class ToTestInt{
    Run | Debug
    public static void main(String arg[]){
        test t = new arithmetic();
        t.square();
    }
}

```

```
PS E:\codes\java\lab 5> javac ToTestInt.java
PS E:\codes\java\lab 5> java ToTestInt
Square
PS E:\codes\java\lab 5>
```

2.Write a program to create interface A, in this interface we have two method meth1 and meth2. Implements this interface in another class named MyClass.

```java
interface A{
    void meth1();
    void meth2();
}

class MyClass implements A{
    public void meth1(){
        System.out.println("Method 1");
    }
    public void meth2(){
        System.out.println("Method 2");
    }
    Run | Debug
    public static void main(String arg[]){
        MyClass obj = new MyClass();
        obj.meth1();
        obj.meth2();
    }
}

```

```
PS E:\codes\java\lab 5> cd "e:\codes\java\lab 5\q2\" ; if ($?) { javac MyClass.java } ; if ($?) { java MyClass }
Method 1
Method 2
PS E:\codes\java\lab 5\q2>
```

3.Write a program in Java to show the usefulness of Interfaces as a place to keep constant value of the program

```java

interface compute{
    int operation(int x, int y);
}
class Addition implements compute{
    public int operation(int x, int y){
        return x+y;
    }
}
class Multiplication implements compute{
    public int operation(int x, int y){
        return x*y;
    }
}
class Exp3{
    Run | Debug
    public static void main(String arg[]){
        Addition a = new Addition();
        Multiplication m = new Multiplication();
        compute c;
        c = a;
        System.out.println("Addition: "+c.operation(1,2));
        c = m;
        System.out.println("Multiplication: "+c.operation(2,1));
    }
}

```

```
PS E:\codes\java\lab 5\q2> cd "e:\codes\java\lab 5\q3\" ; if ($?) { javac Exp3.java } ; if ($?) { java Exp3 }
Addition: 3
Multiplication: 2
PS E:\codes\java\lab 5\q3>
```

4. Write a program to create an Interface having two methods division and modules. Create a class, which overrides these methods.

```java


interface subject{
    void division(int x);
    void modules(int y);
}
class sub implements subject{
    int div, mod;
    public void division(int x){
        div = x;
    }
    public void modules(int y){
        mod = y;
    }
    void print(){
        System.out.println("Division: "+div);
        System.out.println("Modules: "+mod);
    }
}
class Exp4{
    Run | Debug
    public static void main(String arg[]){
        sub s = new sub();
        s.division(1);
        s.modules(12);
        s.print();
    }
}

```

```
PS E:\codes\java\lab 5\q3> cd "e:\codes\java\lab 5\q4\" ; if ($?) { javac Exp4.java } ; if ($?) { java Exp4 }
Division: 1
Modules: 12
PS E:\codes\java\lab 5\q4>
```

5. Write program to create an interface StackInterface having methods push (), pop () and display (). StackClass implements StackInterface. Class StackClass contains the main method which is having a switch case for selecting the particular operation of the stack.

```java
import java.io.*;
import java.util.Scanner;
class stack {
    static int ch;
    int element, maxsize, top;
    int[] st;
    public stack() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter stack size");
        maxsize = sc.nextInt();
        st = new int[maxsize];
        top = -1;
    }
    public void push(int element) {
        if(top ==maxsize-1) {
            System.out.println("\n----------Overflow----------\n");
        } else {
            try {
                st[++top] = element;
            } catch (ArrayIndexOutOfBoundsException e) {
                System.out.println(e);
            }
        }
    }
    public int pop() {
        if (top == -1) {
            System.out.println("\n----------UnderFlow----------\n");
            return (-1);
        }
        else {
            System.out.printf("Popped element is " +(st[top--]));
            return 0;
        }
    }

    public void display(int[] st, int max_size) {
        int i;
        System.out.println("Stack Elements:");
        for (i = 0; i <= max_size; i++)
            System.out.println(st[i]);
        new myStack();
    }
}
 class myStack {
    static int ch;

    Run | Debug
    public static void main(String[] args) {
        stack obj = new stack();
        while (true) {
            System.out.println("\nEnter your choice\n1.PUSH\n2.POP\n3.Display\n4..EXIT");
            Scanner integer = new Scanner(System.in);
            ch = integer.nextInt();
            switch (ch) {
                case 1:
                    System.out.println("Enter Element");
                    obj.element = integer.nextInt();
                    obj.push(obj.element);
                    break;
                case 2:
```

```
                    obj.pop();
                    break;
                case 3:
                    obj.display(obj.st, obj.top);
                    break;
                case 4:
                    System.exit(0);
                default:
                    System.out.println("Wrong option");
            }
        }
    }
}

```

```
PS E:\codes\java\lab 5\q5> javac myStack.java
PS E:\codes\java\lab 5\q5> java myStack
Enter stack size
4

Enter your choice
1.PUSH
2.POP
3.Display
4..EXIT
1
Enter Element
3

Enter your choice
1.PUSH
2.POP
3.Display
4..EXIT
1
Enter Element
5

Enter your choice
1.PUSH
2.POP
3.Display
4..EXIT
1
Enter Element
7

Enter your choice
1.PUSH
2.POP
3.Display
4..EXIT
1
Enter Element
9

Enter your choice
1.PUSH
2.POP
3.Display
```

```
4..EXIT
1
Enter Element
2

-----------Overflow------------


Enter your choice
1.PUSH
2.POP
3.Display
4..EXIT
3
Stack Elements:
3
5
7
9

Enter your choice
1.PUSH
2.POP
3.Display
4..EXIT
2
Popped element is 9
Enter your choice
1.PUSH
2.POP
3.Display
4..EXIT
2
Popped element is 7
Enter your choice
1.PUSH
2.POP
3.Display
4..EXIT
2
Popped element is 5
Enter your choice
1.PUSH
2.POP
3.Display
```

```
4..EXIT
2
Popped element is 3
Enter your choice
1.PUSH
2.POP
3.Display
4..EXIT
2

----------UnderFlow----------


Enter your choice
1.PUSH
2.POP
3.Display
4..EXIT
4
PS E:\codes\java\lab 5\q5>
```