PRACTICAL 3

Aim:-

Define an abstract class Shape in package P1. Inherit two more classes: Rectangle in package P2 and Circle in package P3. Write a program to ask the user for the type of shape and then using the concept of dynamic method dispatch, display the area of the appropriate subclass. Also write appropriate methods to read the data. The main() function should not be in any package.

Code:-

```
/****Abstract Shape Class *****/
package P1;
public abstract class Shape {
public abstract void area();
}

/*****Rectangle.java*****/
package P2;
import P1.Shape;
public class Rectangle extends Shape {
double lenght;
double breadth;
public Rectangle(double lenght,double breadth) {
this.lenght=lenght;
this.breadth=breadth;
}
@Override
public void area() {
System.out.println("The area of
rectangle :"+lenght*breadth);
}
}

/*****Circle.java*****/
package P3;
import P1.Shape;
public class Circle extends Shape {
```

```
double radius;
public Circle(double radius) {
this.radius=radius;
}
@Override
public void area() {
System.out.println("The area of circle :"+
Math.pow(radius, 2)*3.14);
}
}

// main function class to ask the user for the type of shape
import java.util.Scanner;
import P1.Shape;
import P2.Rectangle;
import P3.Circle;
public class Main {
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
Rectangle R1 = new Rectangle(5,4);
Circle C1 = new Circle(5);
Shape ref = null;
System.out.println("Enter the type of the shape ( 1. Rectangle
2.Circle ) : ");
int no = sc.nextInt();
if(no == 1) {
ref = R1;
}
else if (no ==2){
ref = C1;
}
else {
System.out.println("Invalid Input");
}
ref.area();
}
```

}

Output:-

PRACTICAL 4

Aim:-

Create an exception subclass UnderAge, which prints “Under Age” along with the agevalue when an object of UnderAge class is printed in the catch statement. Write aclass exceptionDemo in which the method test() throws UnderAge exception if thevariable age passed to it as argument is less than 18. Write main() method also to show working of the program.

Code:-

```
/*****UnderAge.java*****/
package practical4;
public class UnderAge extends Exception{
int age;
public UnderAge(String s) {
super(s);
}
}
class Voter{
int age;
public Voter(int age) {
this.age = age;
}
```

```
void checkEligibity() {
try {
if(age<18) {
throw new UnderAge("Hey Kido , get out of
here.");
}
System.out.println("Congrates! You are eligible for
vote");
}
catch(UnderAge voter) {
System.out.println(voter.getMessage());
}
}
}
```

```
/*****Main.java*****/
package practical4;
import java.util.Scanner;
public class Main {
public static void main(String args[]) {
Scanner sc = new Scanner(System.in);
System.out.println("Enter your age : ");
int age = sc.nextInt();
Voter man = new Voter(age);
man.checkEligibity();
}
}
```

Output:-

PRACTICAL 5

Aim:-

Write a program to implement stack. Use exception handling to manage underflow and overflow conditions.

Code:-

```
/*****Stack.java*****/
```

```
package practical5;
class StackException extends Exception {
    public StackException(String s){
    super(s);
    }
}
public class Stack {
    public final int MAX = 100;
    public int top;
    public int arr[] = new int[MAX];
    Stack() {
        top = -1;
    }
    public void Push(int val) throws StackException{
    if(top == MAX-1) {
        throw new StackException("Stack Overflow");
    }
    arr[++top] = val;
    }
    public void Pop() throws StackException{
        if(top == -1) {
            throw new StackException("Stack Underflow");
        }
        top--;
    }
    public int Top() {
        return top;
    }
    public boolean isEmplty() {
        return top == -1;
    }
}

/*****Main.java*****/
package practical5;
public class Main {
```

```
    public static void main(String[] args) {
        Stack st = new Stack ();
        try {
        st.Push(0);
        st.Push(1);
        st.Push(2);
        System.out.println(st.Top());
        st.Pop();
        System.out.println(st.Top());
        st.Pop();
        st.Pop();
        System.out.println(st.isEmplty());
        st.Pop();
        }
        catch(StackException e) {
            System.out.println("Error detected :
            "+e.getMessage());
            System.exit(1);
        }
    }
}
```

Output:-

PRACTICAL 1

CODE

RESULT

PRACTICAL 2
CODE

RESULT

PRACTICAL 1

CODE

RESULT