# Practical 3

**Q.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:-**

**P1/Shape.java**

package P1;

public abstract class Shape {

protected abstract void getData() throws java.io.IOException; public abstract double area() throws java.io.IOException;

}

## P2/Rectangle.java

package P2; import java.io.\*; import P1.\*;

public class Rectangle extends Shape { private double length;

private double breadth;

protected void getData() throws IOException {

BufferedReader br = new BufferedReader(new InputStreamReader( System.in

));

System.out.print("Enter Length of Rectangle: "); length = Double.parseDouble(br.readLine()); System.out.print("Enter Breadth of Rectangle: "); breadth = Double.parseDouble(br.readLine());

}

public double area() throws IOException { getData();

return length \* breadth;

}

}

## P3/Circle.java

package P3; import java.io.\*; import P1.\*;

public class Circle extends Shape { private double radius;

protected void getData() throws IOException {

BufferedReader br = new BufferedReader(new InputStreamReader(

System.in

));

System.out.print("Enter Radius of Circle: "); radius = Double.parseDouble(br.readLine());

}

public double area() throws IOException { getData();

return Math.PI \* radius \* radius;

}

}

## practhree.java

package Dhruv\_Java**.**Dhruv\_Java; import java**.**io**.\***;

import Dhruv\_Java**.**P1**.\***; import Dhruv\_Java**.**P2**.\***; import Dhruv\_Java**.**P3**.\***; **public class** practhree {

**static int** getShapeType() **throws** IOException {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in)); System.out.println("==============\n SHAPE TYPE \n==============");

System.out.println(" (1) Rectangle\n (2) Circle"); System.out.print("Enter Choice: ");

return Integer.parseInt(br.readLine());

}

**public static void** main(String[] args) **throws** IOException { Shape ref;

**boolean** flag = false; while (!flag) {

switch (getShapeType()) { case 1:

flag = true;

ref = new Rectangle();

System.out.println("Area: " + ref.area() + " sq units"); break;

case 2:

flag = true;

ref = new Circle();

System.out.println("Area: " + ref.area() + " sq units"); break;

default:

System.err.println("Invalid Option"); break;

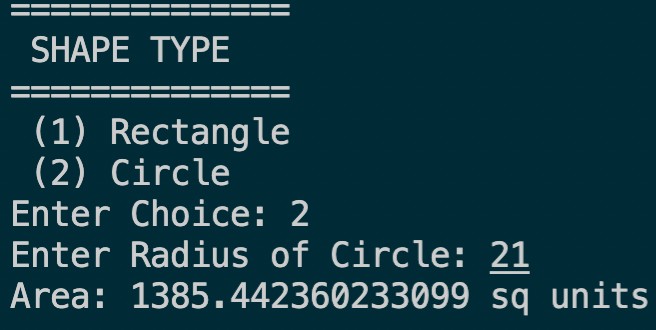
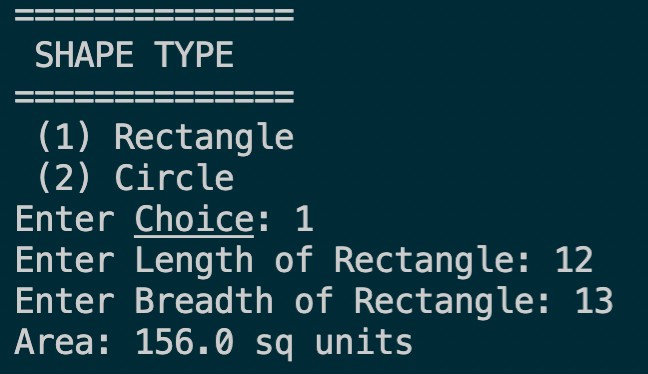
}

}

}

}

**Output:-**



# Practical 4

**Q.Create an Exception subclass UnderAge, which prints “Under Age” along with the age value when an object of UnderAge class is printed in the catch statement. Write a class exceptionDemo in which the method test() throws UnderAge exception if the variable age passed to it as argument is less than 18.**

**Write main() method also to show working of the program.**

**Code:-**

**Underage.java:**

**package Dhruv\_Java;**

**public class underage extends Exception { final private int age;**

**public underage(int age) { this.age = age;**

**}**

**@Override**

**public String getMessage() {**

**return "UnderAge: " + age + " is less than 18";**

**} }**

## exceptionDemo.java:

**package Dhruv\_Java;**

**import java.util.Scanner; class exceptionDemo {**

**static void test(int age) throws underage {**

**if (age < 18)**

**throw new underage(age);**

**}**

**public static void main(String[] args) { Scanner sc = new Scanner(System.in); System.out.print("Enter Age: ");**

**int age = sc.nextInt(); try {**

**test(age);**

**System.out.println("Test Successful");**

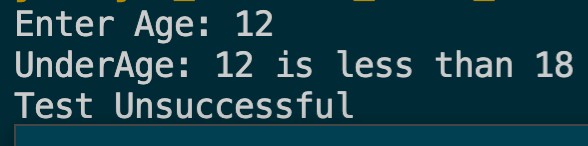
**} catch (underage e) { System.err.println(e.getMessage()); System.out.println("Test Unsuccessful");**

**}**

**finally { sc.close();**

**}**

**} }**

**Output:-**

# Practical 5

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

**Code:-**

**Stack.java:**

package Dhruv\_Java;

public class Stack { private int tos; private int[] array;

final private int size; public Stack(int size) { this.tos = -1;

this.size = size;

this.array = new int[this.size];

}

public void push(int e) throws StackException { if (tos == size - 1)

throw new StackException("Stack Overflow: could not push " + e); else

this.array[++this.tos] = e;

}

public int pop() throws StackException { if (this.tos < 0) {

throw new StackException("Stack Underflow: could not pop");

} else

return this.array[this.tos--];

}

public int getTOS() { return this.tos;

}

@Override

public String toString() {

return "Stack<size=" + this.size + ">";

} }

## StackException.java:

package Dhruv\_Java;

public class StackException extends Exception { final private String message;

public StackException(String message) { this.message = message;

}

@Override

public String getMessage() { return this.message;

}

}

## pracfive.java:

package Dhruv\_Java;

import java**.**util**.**Random;

**public class** pracfive {

**public static void** main(String[] args) {

**int** r;

Stack stack = new Stack(5);

Random random = new Random(1337);

System.out.println("Created stack of size 5..."); System.out.println("Pushing integers onto stack..."); while (true) {

r = random.nextInt(100); System.out.println("Pushing " + r + "..."); try {

stack.push(r); System.out.println(

"Elements in Stack = " + (stack.getTOS() + 1)

);

} catch (StackException e) { System.err.println(e.getMessage()); break;

}

}

System.out.println("Popping integers from stack..."); while (true) {

System.out.println(

"Elements in Stack = " + (stack.getTOS() + 1)

);

try {

System.out.println("Popped " + stack.pop() + "...");

} catch (StackException e) { System.err.println(e.getMessage()); break;

}

}

} }

## Output:-