

NAME	:	NITHISH G
REG NO.	:	19BCS0012
COURSE	:	JAVA PROGRAMMING
DATE	:	01.05.2021

1. a) Multilevel inheritance calculator

Source code :

```
import java.lang.Math;
import java.util.Scanner;

class calculator
{
    double x = 0.0,y =0.0, result =0.0;
    public void set_value(double a , double b)
    {
        x = a ;
        y = b ;
    }

    public void show_result()
    {
        System.out.println(result);
    }
}

class simple_calculator extends calculator
{
    public void add()
    {
        result = x + y ;
    }
    public void sub()
    {
        result = x - y ;
    }
    public void mul()
    {
        result = x * y ;
    }
}
```

```

    public void div()
    {
        result = x / y ;
    }
}

class Sceintfic_calculator extends simple_calculator{

    public void pow()
    {
        result = Math.pow(x, y);
    }
    public void sqrt()
    {
        result = Math.sqrt(x);
    }
    public void epow()
    {
        result = Math.exp(x);
    }

}

public class multilevel_eg1 {

    public static void main(String[] args) {

        double input1=0.00 , input2 = 0.00;
        Sceintfic_calculator obj = new Sceintfic_calculator();

        int choice = 0 ;
        System.out.print("\t Name : Nithish G \n\t Regno No.: 19BCS0012\n");
        System.out.print("\t -----");
        while (choice!= 999)
        {
            System.out.println("\n\n 1.Addtion  2.Subration  3.Multiplication");
            System.out.println(" 4.Division  5.Power    6. Square  7.Epower");

            System.out.print("\n Enter the Choice : ");
            Scanner read = new Scanner (System .in);
            choice = read .nextInt();
            if(choice <= 5)
            {
                System.out.print(" Enter Input 1 \t = ");
                input1 = read.nextDouble();
            }
        }
    }
}

```

```

        System.out.print(" Enter Input 2 \t = ");
        input2 = read.nextDouble();
    }
    else
    {
        System.out.print(" Enter Input    : ");
        input1 = read.nextDouble();
    }
}
switch(choice)
{
case 1:
    obj.set_value(input1,input2);
    obj.add();
    System.out.print(" Addition of "+ input1+" + " + input2 + " = ");
    obj.show_result();
    System.out.print(" -----");
    break;
case 2:
    obj.set_value(input1,input2);
    obj.sub();
    System.out.print(" Subraction of "+ input1+" - " + input2 + " = ");
    obj.show_result();
    System.out.print(" -----");
    break;
case 3:
    obj.set_value(input1,input2);
    obj.mul();
    System.out.print(" Multiplication of "+ input1+" x " + input2 + " = ");
    obj.show_result();
    System.out.print(" -----");
    break;
case 4:
    obj.set_value(input1,input2);
    obj.div();
    System.out.print(" Division of "+ input1+" / " + input2 + " = ");
    obj.show_result();
    System.out.print(" -----");
    break;
case 5:
    obj.set_value(input1,input2);
    obj.pow();
    System.out.print(" Power of "+ input1+" ^ " + input2 + " = ");
    obj.show_result();
    System.out.print(" -----");

```

```

        break;
    case 6:
        obj.set_value(input1,0);
        obj.sqrt();
        System.out.print(" Squart root of "+ +input1+" = ");
        obj.show_result();
        System.out.print(" -----");
        break;
    case 7:
        obj.set_value(input1,0);
        obj.epow();
        System.out.print(" EPower root of "+ input1 + " = ");
        obj.show_result();
        System.out.print(" -----");
        break;

    default :
        System.out.println(" invaild Input .....");
        System.out.print(" -----");

}
}
}

```

Output

```

<terminated> multilevel_eg1 [Java Application] C:\Program Files\Java\jdk1.7
Name      : Nithish G
Regno No.: 19BCS0012
-----

1.Addition  2.Subration  3.Multiplication
4.Division  5.Power     6. Square root  7.Epower

Enter the Choice : 1
Enter Input 1    = 23.42
Enter Input 2    = 43.21
Addition of 23.42 + 43.21 = 66.63
-----

1.Addition  2.Subration  3.Multiplication
4.Division  5.Power     6. Square root  7.Epower

Enter the Choice : 2
Enter Input 1    = 45.32
Enter Input 2    = 21.12
Subraction of 45.32 - 21.12 = 24.2

package (default package)
import java.util.Scanner;

public class Sceintfic {
    String Name;
    String Regno;

    public Sceintfic() {
        Name = "Nithish G";
        Regno = "19BCS0012";
    }

    public void show() {
        System.out.println("Name : " + Name);
        System.out.println("Regno : " + Regno);
    }

    public void menu() {
        System.out.println("1.Addition  2.Subration  3.Multiplication");
        System.out.println("4.Division  5.Power     6. Square root  7.Epower");
    }

    public void add(int a, int b) {
        System.out.println("Addition of " + a + " + " + b + " = " + (a + b));
    }

    public void sub(int a, int b) {
        System.out.println("Subraction of " + a + " - " + b + " = " + (a - b));
    }

    public void mul(int a, int b) {
        System.out.println("Multiplication of " + a + " * " + b + " = " + (a * b));
    }

    public void div(int a, int b) {
        System.out.println("Division of " + a + " / " + b + " = " + (a / b));
    }

    public void pow(int a, int b) {
        System.out.println("Power of " + a + " ^ " + b + " = " + (Math.pow(a, b)));
    }

    public void sqrt(int a) {
        System.out.println("Square root of " + a + " = " + (Math.sqrt(a)));
    }

    public void epow(int a, int b) {
        System.out.println("EPower root of " + a + " = " + (Math.pow(a, 1/b)));
    }

    public static void main(String[] args) {
        Sceintfic obj = new Sceintfic();
        obj.show();
        obj.menu();
        Scanner sc = new Scanner(System.in);
        int choice = sc.nextInt();
        int input1 = sc.nextInt();
        int input2 = sc.nextInt();

        switch (choice) {
            case 1: obj.add(input1, input2); break;
            case 2: obj.sub(input1, input2); break;
            case 3: obj.mul(input1, input2); break;
            case 4: obj.div(input1, input2); break;
            case 5: obj.pow(input1, input2); break;
            case 6: obj.sqrt(input1); break;
            case 7: obj.epow(input1, input2); break;
            default: System.out.println(" invaild Input .....");
        }
    }
}

```

Subtraction of 45.32 - 21.12 = 24.2

1.Addtion 2.Subration 3.Multiplication
4.Division 5.Power 6. Square root 7.Epower

Enter the Choice : 3

Enter Input 1 = 4.12

Enter Input 2 = 5.2

Multiplication of 4.12 x 5.2 = 21.4240000000000

1.Addtion 2.Subration 3.Multiplication
4.Division 5.Power 6. Square root 7.Epower

Enter the Choice : 4

Enter Input 1 = 91.9

Enter Input 2 = 8

Division of 91.9 / 8.0 = 11.4875

1.Addtion 2.Subration 3.Multiplication
4.Division 5.Power 6. Square root 7.Epower

Enter the Choice : 5

Enter Input 1 = 2

Enter Input 2 = 6

Power of 2.0 ^ 6.0 = 64.0

1.Addtion 2.Subration 3.Multiplication
4.Division 5.Power 6. Square root 7.Epower

Enter the Choice : 6

Enter Input : 2704

Squart root of 2704.0 = 52.0

1.Addtion 2.Subration 3.Multiplication
4.Division 5.Power 6. Square root 7.Epower

Enter the Choice : 7

Enter Input : 3

EPower root of 3.0 = 20.085536923187668

1.Addtion 2.Subration 3.Multiplication
4.Division 5.Power 6. Square root 7.Epower

Enter the Choice : 999

System Exit....

```
70 // Enter the Choice
71 // Scanner (System .i
72 xtInt();
73
74
75 int(" Enter Input 1
76 .nextDouble();
77 int(" Enter Input 2
78 .nextDouble();
79
```

```
57 String[] args) {
58
59 input2 = 0.00;
60 obj = new Sceintfic_
61
62
63 Name : Nithis
64 t -----
65
66
67 n("\n\n 1.Addtion
68 n(" 4.Division 5.Po
69
70 "\n Enter the Choice
71 // Scanner (System .i
72 xtInt();
73
74
75 int(" Enter Input 1
76 .nextDouble();
77 int(" Enter Input 2
78 .nextDouble();
79
```

```
68 n(" 4.Division 5.Po
69
70 "\n Enter the Choice
71 // Scanner (System .i
72 xtInt();
73
74
75 int(" Enter Input 1
76 .nextDouble();
77 int(" Enter Input 2
78 .nextDouble();
79
```

1 b). Hierarchical inheritance (Calculate Areas)

Source Code:

```
import java.util.Scanner;

abstract class graphic
{
    double input1 ,input2, input3;
    abstract void get_values();
    abstract void show_area();
    static Scanner read = new Scanner(System.in);
}

class Circle1 extends graphic
{
    void get_values()
    {
        System.out.println("      Area of Circle ");
        System.out.println("      ----- ");
        System.out.print(" Enter Radius      : ");
        this.input1 = read.nextDouble();
    }
    void show_area()
    {
        System.out.println(" Area of Circle      : " + ((3.14) * (input1 * input1))+"\n");
    }
}

class Rectangle extends graphic
{
    void get_values()
    {
        System.out.println("      Area of Rectangle ");
        System.out.println("      ----- ");
        System.out.print(" Enter length      : ");
        input1 = read.nextDouble();
        System.out.print(" Enter Width      : ");
        input2 = read.nextDouble();
    }
    void show_area()
    {
        System.out.print(" Area of Rectangle  : " + (input1*input2)+"\n\n");
    }
}
```

```

}
class Triangle extends graphic
{
    void get_values()
    {
        System.out.println("      Area of Triangle ");
        System.out.println("      ----- ");
        System.out.print(" Enter 1st side   : ");
        input1 = read.nextDouble();
        System.out.print(" Enter 2nd side   : ");
        input2 = read.nextDouble();
        System.out.print(" Enter 3rd side   : ");
        input3 = read.nextDouble();
    }
    void show_area()
    {
        double s = (input1+input2+input3)/2;

        System.out.print(" Area of Tringle   : " + Math.sqrt(s*((s-input1)*(s-input2)*(s-
input3))))+"\n\n");
    }
}
class Square extends graphic
{
    void get_values()
    {
        System.out.println("      Area of Square ");
        System.out.println("      ----- ");
        System.out.print(" Length of side   : ");
        input1 = read.nextDouble();
    }
    void show_area()
    {
        System.out.print(" Area of Square   : " + (input1*input1)+"\n");
    }
}
public class Hierarchical_area {

    public static void main(String[] args) {

        System.out.print("\tName   : Nithish G \n\tReg No. : 19BCS0012\n");
        System.out.print("-----\n\n");
    }
}

```

```

Circle1 circle= new Circle1();
Rectangle rectangle = new Rectangle();
Triangle triangle = new Triangle();
Square square = new Square();

```

```

circle.get_values();
circle.show_area();

```

```

rectangle.get_values();
rectangle.show_area();

```

```

triangle.get_values();
triangle.show_area();

```

```

square.get_values();
square.show_area();

```

```

}

```

```

}

```

Output

The screenshot displays a Java IDE with two windows. The left window, titled 'Hierarchical_area [Java Application]', shows the program's output. The right window, titled 'multilevel_eg1.java', shows the source code.

Output:

```

Name      : Nithish G
Reg No.   : 19BCS0012
-----
      Area of Circle
      -----
Enter Radius      : 5.3
Area of Circle    : 88.2026

      Area of Rectangle
      -----
Enter length      : 13.4
Enter Width       : 21
Area of Rectangle : 281.400000000000003

      Area of Triangle
      -----
Enter 1st side    : 13
Enter 2nd side    : 9
Enter 3rd side    : 15
Area of Tringle   : 58.16517428840044

      Area of Square
      -----
Length of side    : 15
Area of Square    : 225.0

```

Source Code (multilevel_eg1.java):

```

78
79 }
80 public class Hierarchical_area
81
82     public static void main(String[] args) {
83
84         System.out.print("\nName: ");
85         System.out.print("-----");
86
87         Circle1 circle= new Circle1();
88         Rectangle rectangle = new Rectangle();
89         Triangle triangle = new Triangle();
90         Square square = new Square();
91
92         circle.get_values();
93         circle.show_area();
94
95         rectangle.get_values();
96         rectangle.show_area();
97
98         triangle.get_values();
99         triangle.show_area();
100
101         square.get_values();
102         square.show_area();
103
104     }
105

```


2. Create a base class Fruit with name, taste and size as its attributes. Create a method called eat() which describes the name of the fruit and its taste. Inherit the same in 2 other classes Apple and Orange and override the eat() method to represent each fruit taste.

Source Code

```
class fruit
{
    String fruit_name;
    String fruit_taste;
    float fruit_size;
    fruit(String name,String taste, float size)
    {
        fruit_name= name;
        fruit_taste= taste;
        fruit_size= size;
    }
}

class apple extends fruit
{
    apple(String n , String t , float s )
    {
        super(n,t,s);
    }
    public void eat ()
    {
        System.out.println("\n\nName of the Fruit : "+ fruit_name);
        System.out.println("Taste of the Fruit : "+ fruit_taste);
        System.out.println("Size of the Fruit : "+ fruit_size+" cm");
    }
}

class Orange extends fruit
{
    Orange(String n , String t , float s )
    {
        super(n,t,s);
    }
    public void eat ()
    {
        System.out.println("\n\nName of the Fruit : "+ fruit_name);
        System.out.println("Taste of the Fruit : "+ fruit_taste);
        System.out.println("Size of the Fruit : "+ fruit_size+"cm");
    }
}
```

```

public class method_ouerrading_eg {
    public static void main(String[] NITHISH) {

        System.out.print("\t Name      : Nithish G \n\t Regno No.: 19BCS0012\n");
        System.out.print("\t -----");

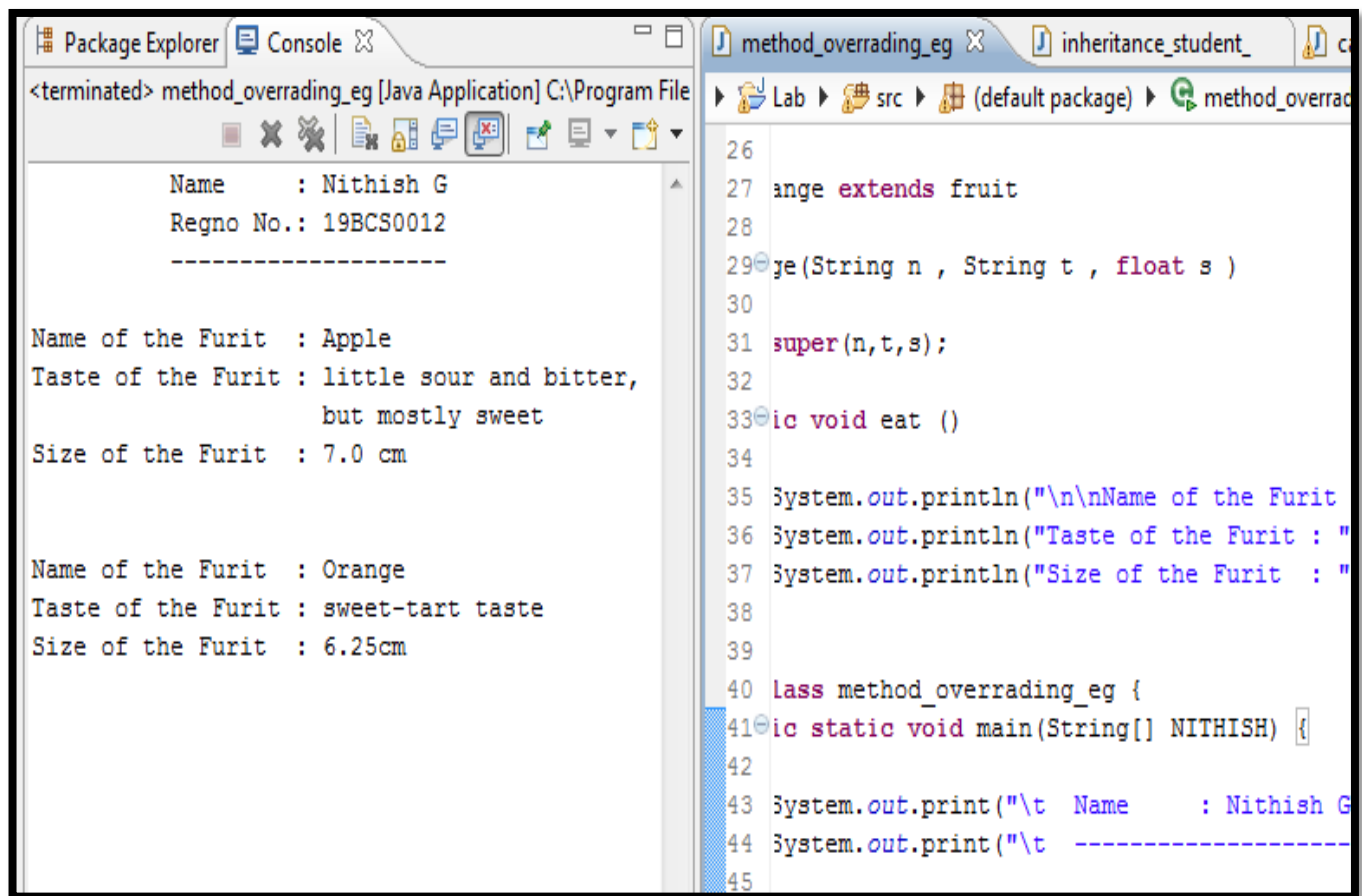
        apple obj1 = new apple("Apple","little sour and bitter,\n\t\t but mostly sweet" , 7);

        Orange obj2 = new Orange("Orange","sweet-tart taste" , 6.25f);
        obj1.eat();
        obj2.eat();

    }
}

```

Output



The screenshot shows an IDE with two windows. The left window, titled 'Console', displays the output of the Java application. The right window, titled 'method_ouerrading_eg', shows the source code of the program.

Console Output:

```

Name      : Nithish G
Regno No.: 19BCS0012
-----

Name of the Furit : Apple
Taste of the Furit : little sour and bitter,
                  but mostly sweet
Size of the Furit  : 7.0 cm

Name of the Furit : Orange
Taste of the Furit : sweet-tart taste
Size of the Furit  : 6.25cm

```

Source Code (method_ouerrading_eg.java):

```

26
27 ange extends fruit
28
29 ge(String n , String t , float s )
30
31 super(n,t,s);
32
33 ic void eat ()
34
35 System.out.println("\n\nName of the Furit
36 System.out.println("Taste of the Furit : "
37 System.out.println("Size of the Furit  : "
38
39
40 lass method_ouerrading_eg {
41 ic static void main(String[] NITHISH) {
42
43 System.out.print("\t Name      : Nithish G
44 System.out.print("\t -----
45

```

3. Create a class named 'Animal' which includes methods like eat() and sleep(). Create a child class of Animal named 'Bird' and override the parent class methods. Add a new method named fly(). Create an instance of Animal class and invoke the eat and sleep methods using this object. Create an instance of Bird class and invoke the eat, sleep and fly methods using this object.

Source Code:

```
class Animal{

    public void Eat()
    {
        System.out.println(" Class : Animal \n method : Eat \n");
    }
    public void sleep(){
        System.out.println(" Class : Animal \n method : Sleep \n");
    }

}
class Bird{

    public void Eat()
    {
        System.out.println(" Class : Bird \n method : Eat \n");
    }
    public void sleep(){
        System.out.println(" Class : Bird \n method : Sleep \n");
    }
    public void Fly(){
        System.out.println(" Class : Bird \n method : Fly \n");
    }

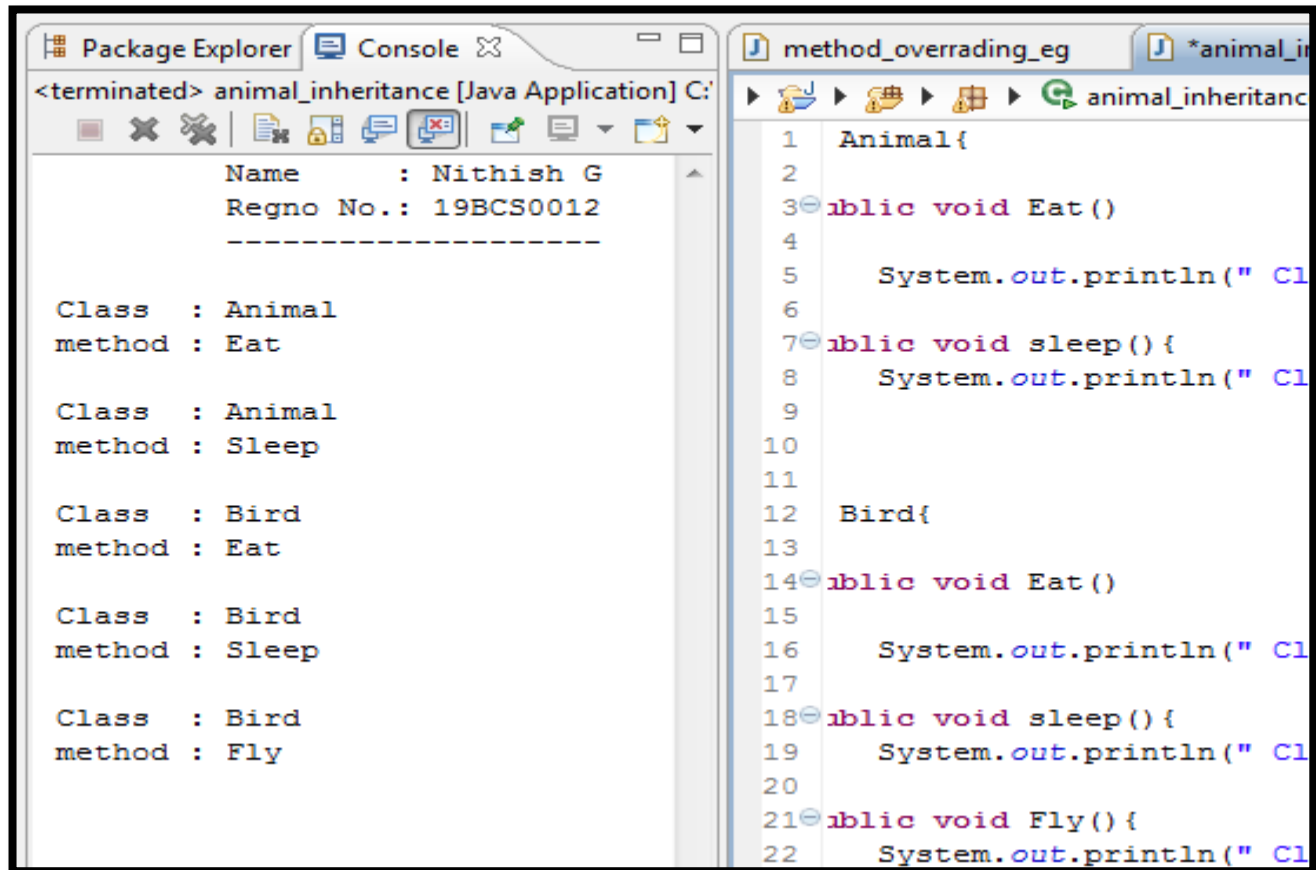
}
public class animal_inheritance
{

    public static void main(String[] args) {
        System.out.print("\t Name      : Nithish G \n\t Regno No.: 19BCS0012\n");
        System.out.print("\t ----- \n\n");
        Animal ob1 = new Animal();
        Bird ob2 = new Bird();

        ob1.Eat();
        ob1.sleep();
        ob2.Eat();
        ob2.sleep();
        ob2.Fly();
    }

}
```

Output



4. Create a superclass, Student, and two subclasses, Undergrad and Grad. The superclass Student should have the following data members: name, ID, grade, age, and address. The superclass, Student should have one method: boolean isPassed (double grade) The purpose of the isPassed method is to take one parameter, grade (value between 0 and 100) and check whether the grade has passed the requirement for passing a course. In the Student class this method should be empty as an abstract method. The two subclasses: Grad and Undergrad, will inherit all data members of the Student class and override the method isPassed. For the UnderGrad class, if the grade is above 70.0, then isPassed returns true, otherwise it returns false. For the Grad class, if the grade is above 80.0, then isPassed returns true, otherwise returns false. Create a test class for your three classes. In the test class, create one Grad object and one Undergrad object. For each object, provide a grade and display the results of the isPassed method.

Source Code

```
abstract class Student {
```

```
    String student_name = "", student_address = "";
```

```
    int student_ID, student_age;
```

```
    double student_grade;
```

```
    abstract boolean isPassed (double grade);
```

```
}  
class Undergrad extends Student {  
  
    Undergrad(String name ,int id, int age, String addr)  
    {  
  
        student_name = name ;  
        student_ID = id;  
        student_age = age;  
        student_address= addr;  
  
        System.out.println("Student_Name    : " +student_name );  
        System.out.println("Student_ID      : " +student_ID );  
        System.out.println("Student_Age     : " +student_age );  
        System.out.println("Student_Address : " +student_address );  
  
    }  
}
```

```
String result = "";
```

```
public boolean isPassed ( double grade )  
{  
    student_grade = grade;  
    System.out.println("Student_grad    : " +student_grade );  
  
    if (student_grade>70)  
    {  
        return true;  
    }  
    else  
  
        return false;  
}
```

```
}  
class Grad extends Student {  
  
    Grad(String name ,int id, int age, String addr)  
    {
```

```
student_name = name ;  
student_ID = id;  
student_age = age;  
student_address= addr;
```

```
System.out.println("Student_Name    : " +student_name );  
System.out.println("Student_ID      : " +student_ID );  
System.out.println("Student_Age     : " +student_age );  
System.out.println("Student_Address : " +student_address );
```

```
}
```

```
String result = "";
```

```
public boolean isPassed ( double grade )
```

```
{
```

```
    student_grade = grade;
```

```
System.out.println("Student_grad    : " +student_grade );
```

```
if (student_grade>80)
```

```
{
```

```
    return true;
```

```
}
```

```
else
```

```
    return false;
```

```
}
```

```
}
```

```
public class inheritance_student_grade {
```

```
    static public void passed()
```

```
{
```

```
        System.out.println("Result          : Passed" );
```

```
}
```

```
    static public void failed()
```

```
{
```

```
        System.out.println("Result          : Failed" );
```

```
}
```

```
public static void main(String[] NITHISH) {
```

```
    System.out.print("\t Name          : Nithish G \n\t Regno No.: 19BCS0012\n");
```

```
    System.out.print("\t ----- \n");
```

```
System.out.println("\t\t Grad");
System.out.print("\t\t-----\n");

Grad[] ob2 = new Grad[2];
ob2[0] = new Grad("Nithish" ,12 , 18, "vellore");

if(ob2[0].isPassed(88.8)) passed();

else failed();
System.out.println();

ob2[1] = new Grad("Kumar" ,13 , 18,"chennai");
if(ob2[1].isPassed(62.2)) {    passed(); }

else    {failed();}
System.out.print("-----\n");

System.out.println("\t\t UnderGrad");
System.out.print("\t\t ----- \n");
Undergrad[] ob1 = new Undergrad[2];

ob1[0] = new Undergrad("Viknesh" ,14 , 17, "vellore");
if(ob1[0].isPassed(72.8)) {passed();}

else    {failed();}

System.out.println();

ob1[1] = new Undergrad("Ramesh" ,15 , 17, "vellore");

if(ob1[1].isPassed(68.8)) { passed(); }

else    {failed();}

}

}
```

Output:

The screenshot shows a Java IDE with two windows. The left window displays the output of a Java application titled "inheritance_student_grade". The output shows the details of three students: Nithish G, Kumar, and Viknesh. The right window shows the source code of the program, "inheritance_student_grade.java".

Output:

```
<terminated> inheritance_student_grade [Java Application] C:\Pr
Name      : Nithish G
Regno No.: 19BCS0012
-----
Grad
-----
Student_Name      : Nithish
Student_ID        : 12
Student_Age       : 18
Student_Address   : vellore
Student_grad      : 88.8
Result            : Passed

Student_Name      : Kumar
Student_ID        : 13
Student_Age       : 18
Student_Address   : chennai
Student_grad      : 62.2
Result            : Failed
-----
UnderGrad
-----
Student_Name      : Viknesh
Student_ID        : 14
Student_Age       : 17
Student_Address   : vellore
Student_grad      : 72.8
Result            : Passed

Student_Name      : Ramesh
Student_ID        : 15
Student_Age       : 17
Student_Name      : Ramesh
Student_ID        : 15
Student_Age       : 17
Student_Address   : vellore
Student_grad      : 68.8
Result            : Failed
```

Source Code:

```
79
80
81
82
83 inheritance_student_grade
84
85 void main(String[]
86
87 t.print("\t Name
88 t.print("\t -----
89
90 t.println("\t\t Grad
91 t.print("\t\t-----
92
93 2 = new Grad[2];
94 new Grad("Nithish"
95 Passed(88.8);
96 t.println();
97
98 new Grad("Kumar" ,1
99 Passed(62.2);
100 t.print("-----
101 t.println("\t
102 t.print("\t ---
103 [] ob1 = new Underg
104
105 new Undergrad("Vikn
106 Passed(72.8);
107
108 t.println();
```

5. Write a Java program which has Interface class for Stack Operations.

(i) A Class that implements the Stack Interface and creates a fixed length Stack

Source Code:

```
import java.util.Scanner;
import java.util.NoSuchElementException;
```



```
interface Stack {

    void push(int data) throws Exception;

    int pop() throws Exception;

    int top() throws Exception;

    boolean isEmpty();

    int size();
}

class FixedArrayStack implements Stack {

    protected int[] stack;

    protected int top = -1;
    public static final int CAPACITY = 3;

    public FixedArrayStack() {
        stack = new int [FixedArrayStack.CAPACITY];
    }

    public void push(int data) throws Exception {
        if (size() == CAPACITY){
            throw new IndexOutOfBoundsException("StackOrverflowException.");
        } else {
            stack[++top] = data;
        }
    }

    public int top() throws Exception {
        if (isEmpty())
            throw new NoSuchElementException("Stack is empty.");
        return stack[top];
    }

    public int pop() throws Exception {
        int data;
        if (isEmpty())
            throw new NoSuchElementException("Stack is empty.");
```

```
data = stack[top];
```

```
stack[top--] = Integer.MIN_VALUE;
```

```
return data;
```

```
}
```

```
public int size() {
```

```
    return (top + 1);
```

```
}
```

```
public boolean isEmpty() {
```

```
    return (top < 0);
```

```
}
```

```
public String toString() {
```

```
    String s = "[";
```

```
    if (size() > 0)
```

```
        s += stack[0];
```

```
    if (size() > 1)
```

```
        for (int i = 1; i <= size() - 1; i++) {
```

```
            s += ", " + stack[i];
```

```
        }
```

```
    return s + "]";
```

```
}
```

```
}
```

```
public class Stack_interface_fixed_length {
```

```
    static Scanner scanner = new Scanner(System.in);
```

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

```
        System.out.print("\t Name      : Nithish G \n\t Regno No.: 19BCS0012\n");
```

```
        System.out.print("\t ----- \n");
```

```
        Stack stack = new FixedArrayStack();
```

```
        System.out.println("    Stack created with size : " + FixedArrayStack.CAPACITY);
```

```
        System.out.println("    -----");
```

```
        while(true){
```

```
            System.out.println("\n 1    2    3    4    5 ");
```

```
            System.out.print(" push() pop() top() iterate() exit(1)\n Choice : ");
```

```
int choice = scanner.nextInt();
```

```
try {
```

```
    switch(choice) {
```

```
    case 1:
```

```
        System.out.println(" Enter a number to push : ");
```

```
        stack.push(scanner.nextInt());
```

```
        System.out.print(" -----");
```

```
        break;
```

```
    case 2:
```

```
        System.out.println(" popped element : " + stack.pop());
```

```
        System.out.print(" -----");
```

```
        break;
```

```
    case 3:
```

```
        System.out.println(" top element : " + stack.top());
```

```
        System.out.print(" -----");
```

```
        break;
```

```
    case 4:
```

```
        System.out.println(" elements are : " + stack.toString());
```

```
        System.out.print(" -----");
```

```
        break;
```

```
    case 5:
```

```
        System.out.print(" exit the sytem: ");
```

```
        System.exit(1);
```

```
    break;
```

```
    default :
```

```
        System.out.println("invalid option.....");
```

```
}
```

```
} catch (Exception e) {
```

```
    System.err.println(" error message : " + e.getMessage());
```

```
}
```

```
}
```

```
}
```

```
}
```

Output:

<terminated> Stack_interface_fixed_length [Java Application] C:\Prog

Name : Nithish G
Regno No.: 19BCS0012

Stack created with size : 3

1 2 3 4 5
push() pop() top() iterate() exit(1)
Choice : 1
Enter a number to push : 12

1 2 3 4 5
push() pop() top() iterate() exit(1)
Choice : 1
Enter a number to push : 13

1 2 3 4 5
push() pop() top() iterate() exit(1)
Choice : 1
Enter a number to push : 14

1 2 3 4 5
push() pop() top() iterate() exit(1)
Choice : 1
Enter a number to push : 15
error message : StackOverFlowException.

1 2 3 4 5
push() pop() top() iterate() exit(1)
Choice : 4
elements are : [12, 13, 14]

1 2 3 4 5
push() pop() top() iterate() exit(1)
Choice : 3
top element : 14

1 2 3 4 5
push() pop() top() iterate() exit(1)
Choice : 2
popped element : 14

1 2 3 4 5
push() pop() top() iterate() exit(1)
Choice : 2
popped element : 13

1 2 3 4 5
push() pop() top() iterate() exit(1)
Choice : 2
popped element : 12

(default package)

```
81
82
83
84 public static void
85
86     System.out.pri
87     System.out.pri
88
89     Stack stack = n
90     System.out.prin
91     System.out.prin
92
93     while(true) {
94     System.out.printl
95     System.out.print (
96
97     int choice = scan
98
99     try {
100
101         switch(choice)
102         case 1:
103             System.out.pr
104             stack.push(sc
105             System.out.pr
106             break;
107
108         case 2:
109             System.out.pr
110             System.out.pr
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
```

```
-----
1      2      3      4      5
push() pop() top() iterate() exit(1)
Choice : 2
error message : Stack is empty.
-----
1      2      3      4      5
push() pop() top() iterate() exit(1)
Choice : 5
exit the Stack :

103      System.out.pr
104      stack.push(sc
105      System.out.pr
106      break;
107
108      case 2:
109      System.out.pr
110      System.out.pr
Problems @ Javadoc Decl
```

(ii) A Class that implements the Stack Interface and creates a Dynamic length Stack.

Source code

```
import java.util.Scanner;
import java.util.NoSuchElementException;

interface dynamicStack {

    void push(int data) throws Exception;

    int pop() throws Exception;

    int top() throws Exception;

    boolean isEmpty();

    int size();
}

class dynamicArrayStack implements Stack {

    protected int capacity;

    protected int[] stack;

    protected int top = -1;

    public dynamicArrayStack(int length) {
        this.capacity = length;
        stack = new int[this.capacity];
    }
}
```

```
public void push(int data) throws Exception {  
    if (size() == capacity){  
        throw new IndexOutOfBoundsException("StackOrverflowException.");  
    } else {  
        stack[++top] = data;  
    }  
}
```

```
public int top() throws Exception {  
    if (isEmpty())  
        throw new NoSuchElementException("Stack is empty.");  
    return stack[top];  
}
```

```
public int pop() throws Exception {  
    int data;  
    if (isEmpty())  
        throw new NoSuchElementException("Stack is empty.");  
    data = stack[top];  
  
    stack[top--] = Integer.MIN_VALUE;  
    return data;  
}
```

```
public int size() {  
    return (top + 1);  
}
```

```
public boolean isEmpty() {  
    return (top < 0);  
}
```

```

public String toString() {
    String s = "[";
    if (size() > 0)
        s += stack[0];
    if (size() > 1)
        for (int i = 1; i <= size() - 1; i++) {
            s += ", " + stack[i];
        }
    return s + "]";
}

```

```

public class Stack_interface_dynamic_length {
    static Scanner read = new Scanner(System.in);

    public static void main(String[] args) {

        System.out.print("\t Name      : Nithish G \n\t Regno No.: 19BCS0012\n");
        System.out.print("\t ----- \n");

        System.out.print("Enter the Size of Stack : ");
        int size = read.nextInt();
        Stack stack = new dynamicArrayStack(size);
        System.out.println("    Stack created with size : " + FixedArrayStack.CAPACITY);
        System.out.println("    -----");

        while(true){
            System.out.println("\n 1    2    3    4    5 ");
            System.out.print("  push() pop() top() iterate() exit(1)\n Choice : ");

```

```
int choice = read.nextInt();
```

```
try {
```

```
    switch(choice) {
```

```
    case 1:
```

```
        System.out.print(" Enter a number to push : ");
```

```
        stack.push(read.nextInt());
```

```
        System.out.print(" -----");
```

```
        break;
```

```
    case 2:
```

```
        System.out.println(" popped element : " + stack.pop());
```

```
        System.out.print(" -----");
```

```
        break;
```

```
    case 3:
```

```
        System.out.println(" top element : " + stack.top());
```

```
        System.out.print(" -----");
```

```
        break;
```

```
    case 4:
```

```
        System.out.println(" elements are : " + stack.toString());
```

```
        System.out.print(" -----");
```

```
        break;
```

```
    case 5:
```

```
        System.out.print(" exit the sytem: ");
```

```
        System.exit(1);
```

```
    break;
```


default :

```
System.out.println("invalid option.....");
```

```
}
```

```
} catch (Exception e) {
```

```
System.err.println(" error message : " + e.getMessage());
```

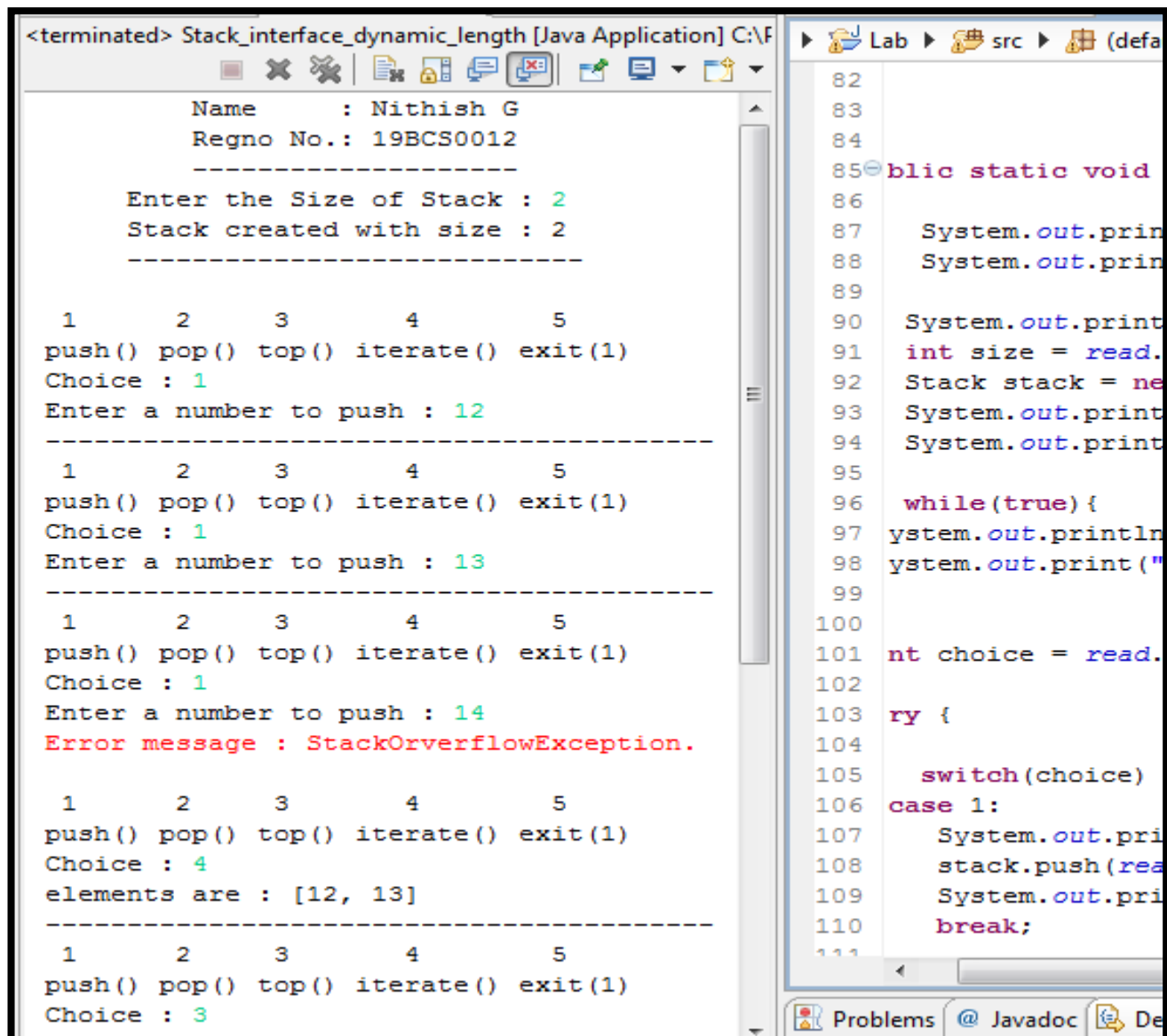
```
}
```

```
}
```

```
}
```

```
}
```

Output



```
<terminated> Stack_interface_dynamic_length [Java Application] C:\F
Name      : Nithish G
Regno No.: 19BCS0012
-----
Enter the Size of Stack : 2
Stack created with size : 2
-----

 1      2      3      4      5
push() pop() top() iterate() exit(1)
Choice : 1
Enter a number to push : 12
-----

 1      2      3      4      5
push() pop() top() iterate() exit(1)
Choice : 1
Enter a number to push : 13
-----

 1      2      3      4      5
push() pop() top() iterate() exit(1)
Choice : 1
Enter a number to push : 14
Error message : StackOverflowException.

 1      2      3      4      5
push() pop() top() iterate() exit(1)
Choice : 4
elements are : [12, 13]
-----

 1      2      3      4      5
push() pop() top() iterate() exit(1)
Choice : 3
```

```
82
83
84
85 public static void
86
87     System.out.prin
88     System.out.prin
89
90     System.out.print
91     int size = read.
92     Stack stack = ne
93     System.out.print
94     System.out.print
95
96     while(true){
97     ystem.out.println
98     ystem.out.print("
99
100
101     nt choice = read.
102
103     ry {
104
105         switch(choice)
106     case 1:
107         System.out.pri
108         stack.push(rea
109         System.out.pri
110         break;
111
```

```
top element : 13
-----
1      2      3      4      5
push() pop() top() iterate() exit(1)
Choice : 2
popped element : 13
-----
1      2      3      4      5
push() pop() top() iterate() exit(1)
Choice : 2
popped element : 12
-----
1      2      3      4      5
push() pop() top() iterate() exit(1)
Choice : 2
Error message : Stack is empty.
-----
1      2      3      4      5
push() pop() top() iterate() exit(1)
Choice : 5
Exit the Stack....

93 System.out.println
94 System.out.println
95
96 while(true){
97     System.out.println
98     System.out.println
99
100
101     int choice = read
102
103     try {
104
105         switch(choice)
106         case 1:
107             System.out.println
108             stack.push(re
109             System.out.println
110             break;
111
```

6. Write a Java program using Synchronized Threads, which demonstrates Producer Consumer concepts.

Source Code:

```
class Shop1
{

    int materials;

    boolean spaceavailable = false;

    public synchronized void put(int materials)
    {

        while (spaceavailable)
        {

            try{ wait(); } catch( Exception e ) {}

        }

        System.out.println("Produced value --> put method : " + materials);

        this.materials = materials;
```

```

spaceavailable = true;
notify();
}

public synchronized void get()
{
while (!spaceavailable)
{
try { wait (); } catch( Exception e ) {}
}
System.out.println("Consumer value --> put method : " + materials);
System.out.println("");
spaceavailable = false;
notify();
}
}

class Producer1 implements Runnable
{

Shop1 shop;

public Producer1(Shop1 shop)
{
this.shop =shop;
Thread t = new Thread(this, "Producer");
t.start();
}

public void run()
{
int i =0;
while(true)
{
shop.put(i++);
try { Thread.sleep(1000) ; } catch(Exception e) {}
}
}
}

```

```
class Consumer1 implements Runnable
{
    Shop1 shop;

    public Consumer1(Shop1 shop)
    {
        this.shop = shop;
        Thread t = new Thread (this, "Consumer1");
        t.start();
    }

    public void run()
    {
        while ( true)
        {
            shop.get();

            try{ Thread.sleep(1000);} catch( Exception e) {}

        }
    }
}

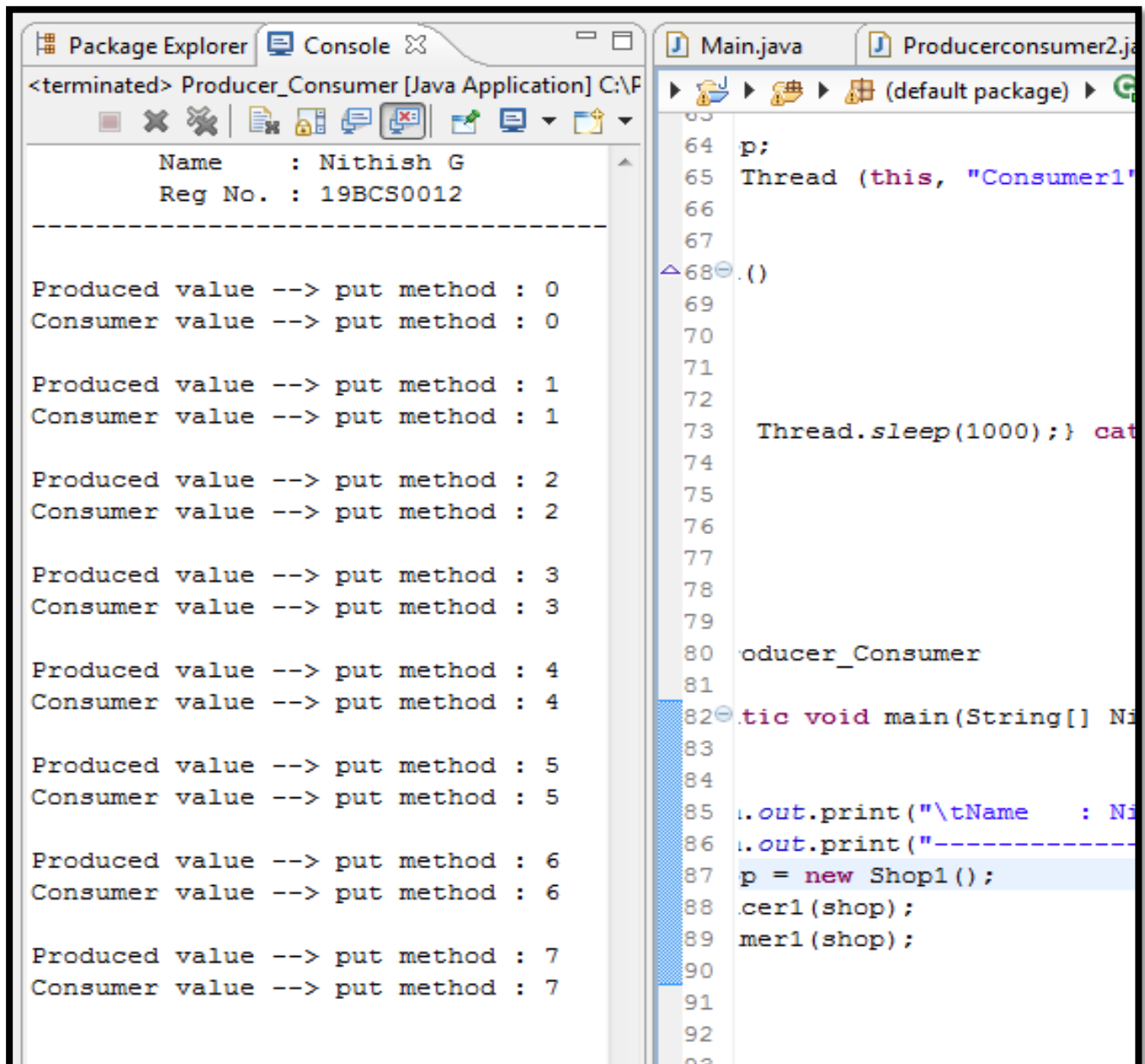
public class Producer_Consumer
{
    public static void main(String[] Nithish)
    {
        System.out.print("\tName   : Nithish G \n\tReg No. : 19BCS0012\n\n");
        Shop1 shop = new Shop1();

        new Producer1(shop);

        new Consumer1(shop);

    }
}
```

Output:



The screenshot displays an IDE with two main panels. The left panel, titled 'Console', shows the output of a Java application named 'Producer_Consumer'. The output consists of eight pairs of lines, each pair representing a state where both the producer and consumer have processed a value from 0 to 7. Each line follows the format 'Produced value --> put method : X' and 'Consumer value --> put method : X'. The right panel shows the source code of 'Main.java' and 'Producerconsumer2.java'. The 'Main.java' file contains the following code:

```
64 p;  
65 Thread (this, "Consumer1"  
66  
67  
68 .()  
69  
70  
71  
72  
73 Thread.sleep(1000);} cat  
74  
75  
76  
77  
78  
79  
80 oducer_Consumer  
81  
82 tic void main(String[] Ni  
83  
84  
85 l.out.print("\tName : Ni  
86 l.out.print("-----  
87 p = new Shop1();  
88 cer1(shop);  
89 mer1(shop);  
90  
91  
92  
93
```

----- Thank You Madam! -----