

CHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY**DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH**

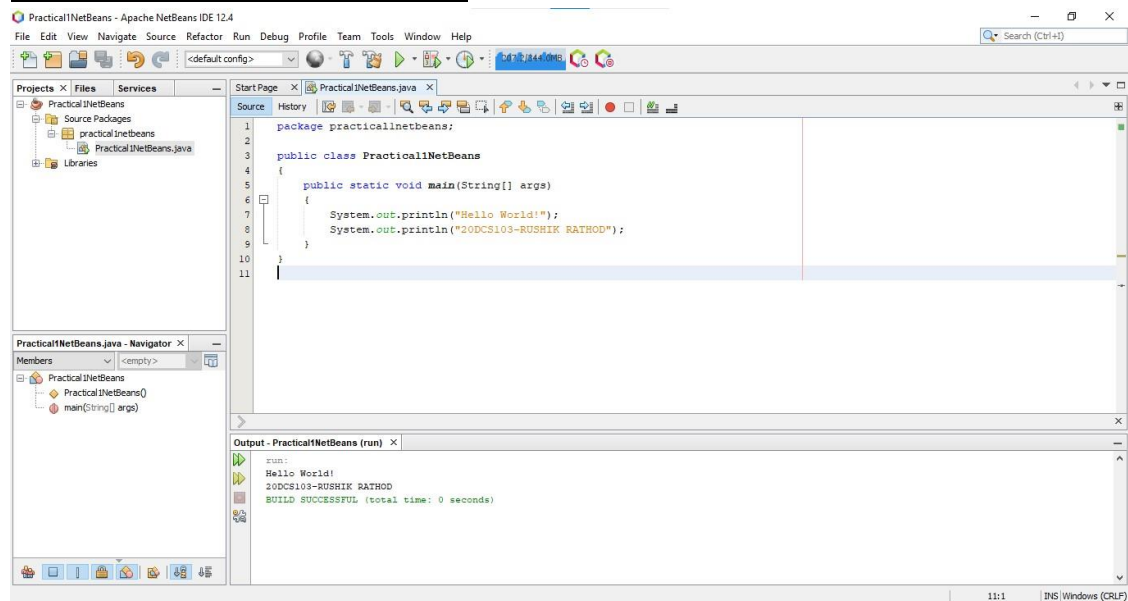
Department of Computer Engineering/Computer Science & Engineering/

Information Technology

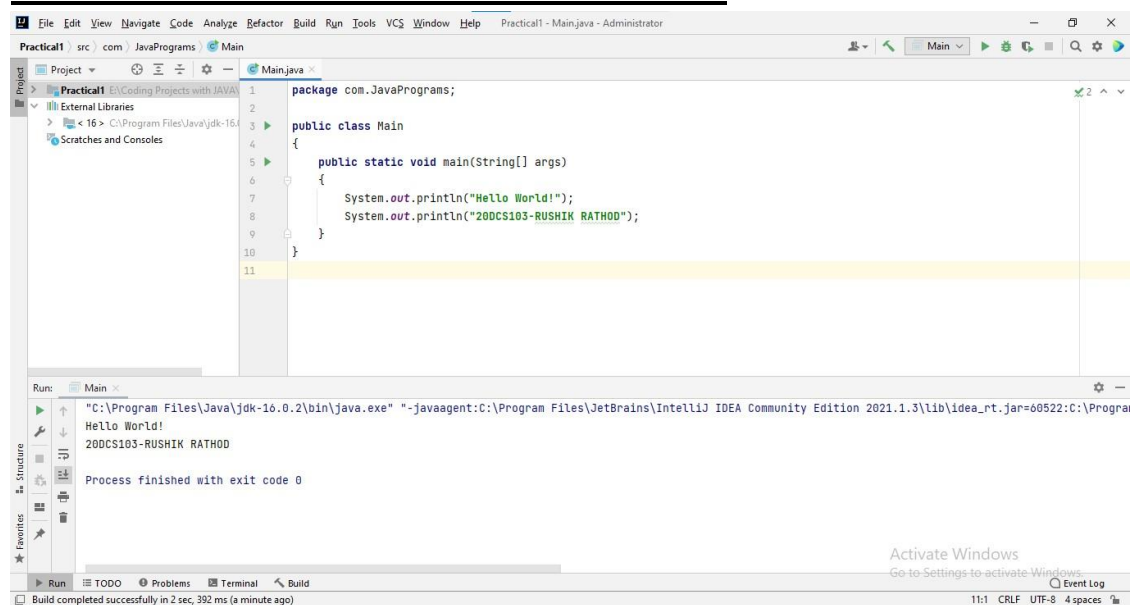
Subject Name: JAVA Programming**Semester: 3****Subject Code: CE251****Academic year: 2021-22****Practical List**

No.	Aim of the Practical
1.	<p style="text-align: center;"><u>PART - 1</u></p> <p>Introduction to Object Oriented Concepts, comparison of Java with other object oriented programming languages. Introduction to JDK, JRE, JVM, javadoc, command line argument. Introduction to Eclipse or Netbean IDE, or BlueJ and Console Programming.</p>

OUTPUT USING NetBeans:

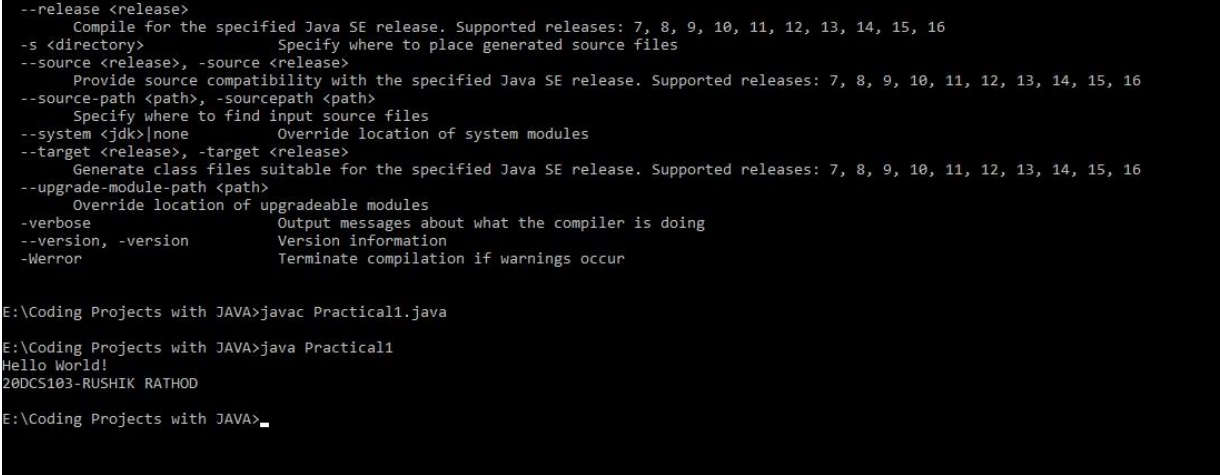


OUTPUT USING VISUAL STUDIO CODE:



PROGRAM CODE:

```
public class Practical1
{
    public static void main(String[] args)
    {
        System.out.println("Hello World!");
        System.out.println("20DCS103-RUSHIK RATHOD");
    }
}
```

OUTPUT:

```
Administrator: Command Prompt
--release <release>
    Compile for the specified Java SE release. Supported releases: 7, 8, 9, 10, 11, 12, 13, 14, 15, 16
-s <directory>
    Specify where to place generated source files
--source <release>, -source <release>
    Provide source compatibility with the specified Java SE release. Supported releases: 7, 8, 9, 10, 11, 12, 13, 14, 15, 16
--source-path <path>, -sourcepath <path>
    Specify where to find input source files
--system <jdk>|none
    Override location of system modules
--target <release>, -target <release>
    Generate class files suitable for the specified Java SE release. Supported releases: 7, 8, 9, 10, 11, 12, 13, 14, 15, 16
--upgrade-module-path <path>
    Override location of upgradeable modules
-verbose
    Output messages about what the compiler is doing
--version, -version
    Version information
-Werror
    Terminate compilation if warnings occur

E:\Coding Projects with JAVA>javac Practical1.java
E:\Coding Projects with JAVA>java Practical1
Hello World!
20DCS103-RUSHIK RATHOD
E:\Coding Projects with JAVA>_
```

CONCLUSION:

I learnt to create and run the programs in different IDE platforms such as NetBeans, Visual Studio Code as well as in the command prompt.

2. **Write a program that declares one integer variable called var1. Give value 10 to this variable and then, using one println() statement, display the value on the screen like this:**

“10 is the value of var1.”

PROGRAM CODE:

```
public class Practical2
{
    public static void main(String[] args)
    {
        int var1=10;
        System.out.println(var1 + " is the value of var1.");
        System.out.println("20DCS103 - RUSHIK RATHOD");
    }
}
```

OUTPUT:

The screenshot shows a Windows Command Prompt window titled "Administrator: Command Prompt". The text inside the window is as follows:

```
Microsoft Windows [Version 10.0.19043.1052]
(c) Microsoft Corporation. All rights reserved.

C:\Users\MIS>e:

E:\>cd/Coding Projects with JAVA

E:\Coding Projects with JAVA>javac Practical3.java

E:\Coding Projects with JAVA>java Practical3
var1 = 1234.5678
var2 = 1234
var3 = 5678
var2 + var3 = 1234.5678
20DCS103-RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

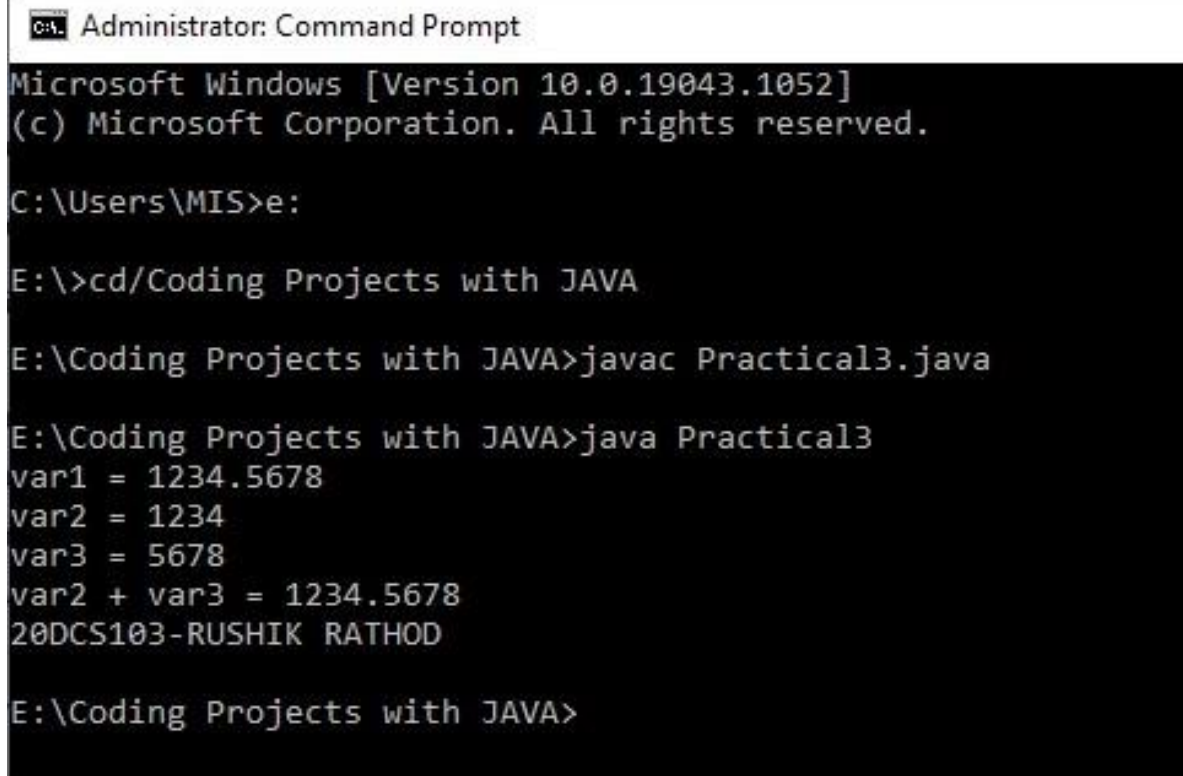
CONCLUSION:

I learnt to use the `println()` method and `+` operator to concatenate two strings in JAVA language.

3. **Write a console program to declare and initialize a double variable with some value such as 1234.5678. Then retrieve the integral part of the value and store it in a variable of type long, and the first four digits of the fractional part and store them in an integer of type short. Display the value of the double variable by outputting the two values stored as integers.**

SOURCE CODE:

```
public class Practical3
{
    public static void main(String[] args)
    {
        double var1 = 1234.5678;
        System.out.println("var1 = " + var1);
        long var2 = (long)var1;
        System.out.println("var2 = " + var2);
        short var3 = (short)((var1%1234)*10000);
        System.out.println("var3 = " + var3);
        System.out.println("var2 + var3 = " + var2 + "." + var3);
        System.out.println("20DCS103-RUSHIK RATHOD");
    }
}
```

OUTPUT:

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.19043.1052]
(c) Microsoft Corporation. All rights reserved.

C:\Users\MIS>e:

E:\>cd/Coding Projects with JAVA

E:\Coding Projects with JAVA>javac Practical3.java

E:\Coding Projects with JAVA>java Practical3
var1 = 1234.5678
var2 = 1234
var3 = 5678
var2 + var3 = 1234.5678
20DCS103-RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

From this program, I learnt type conversion in JAVA language, used “+” operator to concatenate two strings.

4. **Write an application that creates a two dimension array with int values. The first, second and third elements should be arrays with one, two and three numbers respectively. Display the length of each dimension.**

SOURCE CODE:

```
public class Practical4
{
    public static void main(String[] args)
    {
        int[][] arr = {{1}, {2,3}, {4,5,6}};
        int i;
        for(i=0; i<arr.length ; i++)
        {
            System.out.println("The length of dimension " + (i+1)+" is : " +
arr[i].length);
        }
        System.out.println("20DCS103-RUSHIK RATHOD");    }
}
```


OUTPUT:

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.19043.1052]
(c) Microsoft Corporation. All rights reserved.

C:\Users\MIS>e:

E:\>cd/Coding Projects with JAVA

E:\Coding Projects with JAVA>javac Practical4.java

E:\Coding Projects with JAVA>java Practical4
The length of dimension 1 is : 1
The length of dimension 2 is : 2
The length of dimension 3 is : 3
20DCS103-RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

From this program, I learnt to create jagged arrays and also learnt to count the length of the dimension of the array using for loop.

5. **An electric appliance shop assigns code 1 to motor,2 to fan,3 to tube and 4 for wires. All other items have code 5 or more. While selling the goods, a sales tax of 8% to motor,12% to fan,5% to tube light,7.5% to wires and 3% for all other items is charged. A list containing the product code and price in two different arrays.**

Write a java program using switch statement to prepare the bill.

SOURCE CODE:

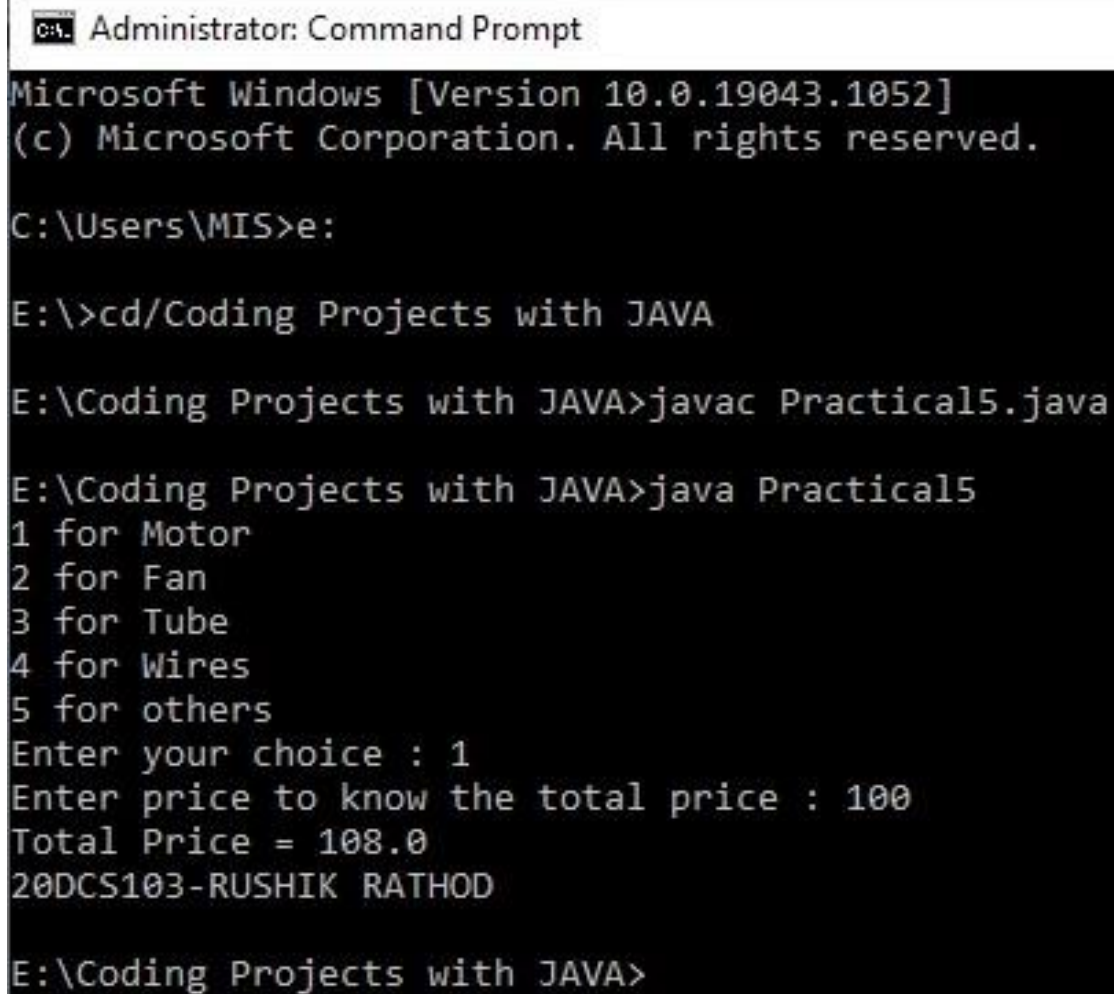
```
import java.util.Scanner;

public class Practical5
{
    public static void main(String[] args)
    {
        double[][] arr = {{1,8}, {2,12}, {3,5}, {4,7.5}, {5,3}};
        int item;
        double price;
        Scanner sc = new Scanner(System.in);

        System.out.println("1 for Motor\n2 for Fan\n3 for Tube\n4 for Wires\n5 for others");
        System.out.print("Enter your choice : ");
        item = sc.nextInt();
        System.out.print("Enter price to know the total price : ");
        price = sc.nextDouble();

        switch(item)
        {
            case 1:
                price = (price*arr[item-1][1]/100)+price;
                System.out.println("Total Price = " + price);
                break;
            case 2:
                price = (price*arr[item-1][1]/100)+price;
```

```
        System.out.println("Total Price = " + price);
        break;
    case 3:
        price = (price*arr[item-1][1]/100)+price;
        System.out.println("Total Price = " + price);
        break;
    case 4:
        price = (price*arr[item-1][1]/100)+price;
        System.out.println("Total Price = " + price);
        break;
    case 5:
        price = (price*arr[item-1][1]/100)+price;
        System.out.println("Total Price = " + price);
        break;
    default:
        System.out.println("Invalid Entry!");
        break;
    }
    System.out.println("20DCS103-RUSHIK RATHOD");
}
}
```

OUTPUT:

```
C:\> Administrator: Command Prompt

Microsoft Windows [Version 10.0.19043.1052]
(c) Microsoft Corporation. All rights reserved.

C:\Users\MIS>e:

E:\>cd/Coding Projects with JAVA

E:\Coding Projects with JAVA>javac Practical5.java

E:\Coding Projects with JAVA>java Practical5
1 for Motor
2 for Fan
3 for Tube
4 for Wires
5 for others
Enter your choice : 1
Enter price to know the total price : 100
Total Price = 108.0
20DCS103-RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

This program is created to display a bill of an electrical appliance shop using switch statements and two dimensional array.

6. **Write a program to show output like:**

* * * * *

* * * *


* * *

* *

*

SOURCE CODE:

```
public class Practical6
{
    public static void main(String[] args)
    {
        int i, j;
        for (i = 5; i >= 1; i--)
        {
            for (j = 1; j <= i; j++)
            {
                System.out.print("*");
            }
            System.out.println();
        }
        System.out.println("20DCS103-RUSHIK RATHOD");
    }
}
```

OUTPUT:

```
C:\> Administrator: Command Prompt

Microsoft Windows [Version 10.0.19043.1052]
(c) Microsoft Corporation. All rights reserved.

C:\Users\MIS>e:

E:\>cd/Coding PROjects with JAVA

E:\Coding Projects with JAVA>javac Practical6.java

E:\Coding Projects with JAVA>java Practical6
*****
*****
***
**
*
20DCS103-RUSHIK RATHOD

E:\Coding Projects with JAVA>_
```

CONCLUSION:

From this program, I learnt to create various types of shapes using for loop and by applying different logics.

PART - 2 : STRINGS

1. **Given a string and a non-negative int n, we'll say that the front of the string is the first 3 chars, or whatever is there if the string is less than length 3.**

Return n copies of the front.

front_times('Chocolate', 2) → 'ChoCho'

front_times('Chocolate', 3) → 'ChoChoCho'

front_times('Abc', 3) → 'AbcAbcAbc'

SOURCE CODE:

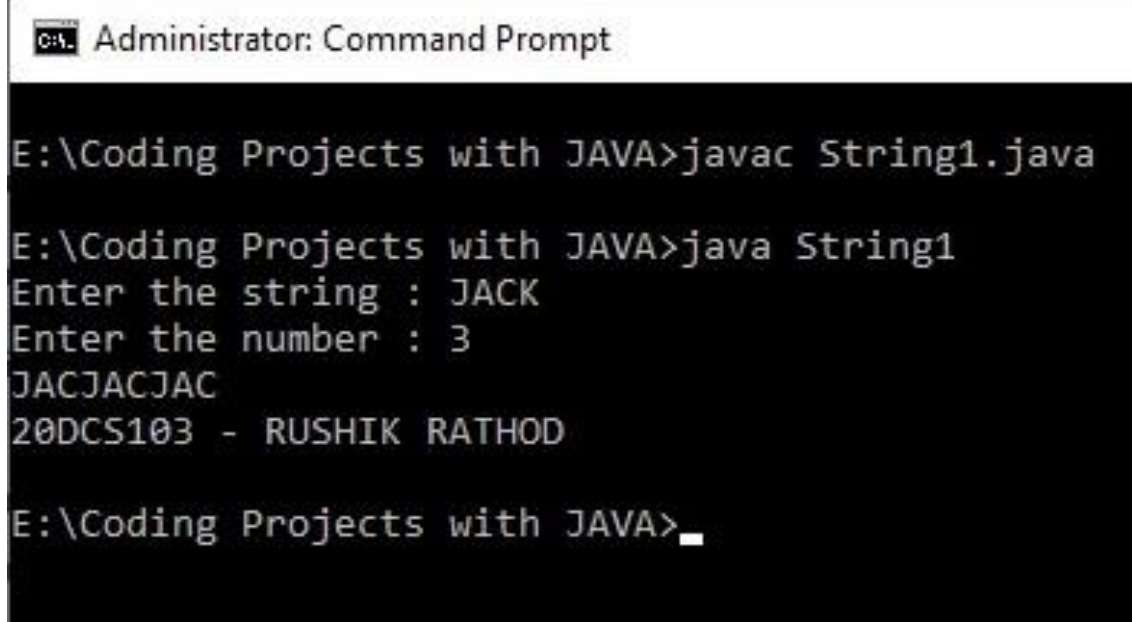
```
import java.util.Scanner;
```

```
public class String_1 {
```

```
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        System.out.print("Enter the string : ");  
        String str1 = sc.nextLine();  
        System.out.print("Enter the number : ");  
        int num = sc.nextInt();  
        front_times(str1, num);  
        sc.close();  
    }
```

```
    public static void front_times(String s1, int n) {  
        while (n > 0) {  
            if (s1.length() > 2) {  
                System.out.print(s1.substring(0, 3));  
                n--;  
            } else {  
                System.out.print("The string should have atleast 3 characters!");  
                n = 0;  
            }  
        }
```

```
    }  
    System.out.println("\n20DCS103 - RUSHIK RATHOD");  
    }  
}
```


OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac String1.java

E:\Coding Projects with JAVA>java String1
Enter the string : JACK
Enter the number : 3
JACJACJAC
20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>_
```

CONCLUSION:

The `nextLine()` is used to get the string, `nextInt` is used to get an integer from the user and I also learnt the use of `substring` and `length` methods.

2. **Given an array of ints, return the number of 9's in the array.**

array_count9([1, 2, 9]) → 1

array_count9([1, 9, 9]) → 2

array_count9([1, 9, 9, 3, 9]) → 3

SOURCE CODE:

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

```
import java.util.Arrays;
```

```
public class String_2
```

```
{
```

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

```
    {
```

```
        Scanner sc = new Scanner(System.in);
```

```
        System.out.print("Enter the size of an array : ");
```

```
        int num = sc.nextInt();
```

```
        int[] arr = new int[num];
```

```
        System.out.println("Enter the array elements...");
```

```
        for (int i = 0; i < arr.length; i++)
```

```
        {
```

```
            arr[i] = sc.nextInt();
```

```
        }
```

```
        array_count9(arr);
```

```
        sc.close();
```

```
    }
```

```
    public static void array_count9(int[] a)
```

```
    {
```

```
        int count = 0;
```

```
        for (int i = 0; i < a.length; i++)
```

```
        {
```

```
            if (a[i] == 9)
```

```
        {  
            count++;  
        }  
    }  
    System.out.print("---> 9's in the array : " + count);  
    System.out.println("\n20DCS103 - RUSHIK RATHOD");  
}  
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac String2.java

E:\Coding Projects with JAVA>java String2
Enter the size of an array : 4
Enter the array elements...
1
5
9
4
---> 9's in the array : 1
20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>_
```

CONCLUSION:

I learnt to call the method and to pass the array as an argument and by using this concept the number 9s in the array can be counted.

3. **Given an array of ints, return True if one of the first 4 elements in the array is a 9. The array length may be less than 4.**

array_front9([1, 2, 9, 3, 4]) → True

array_front9([1, 2, 3, 4, 9]) → False

array_front9([1, 2, 3, 4, 5]) → False

SOURCE CODE:

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

```
import java.util.Arrays;
```

```
public class String_3 {
```

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

```
        Scanner sc = new Scanner(System.in);
```

```
        System.out.print("Enter the size of an array : ");
```

```
        int num = sc.nextInt();
```

```
        int[] arr = new int[num];
```

```
        System.out.println("Enter the array elements...");
```

```
        for (int i = 0; i < arr.length; i++) {
```

```
            arr[i] = sc.nextInt();
```

```
        }
```

```
        boolean b = array_front9(arr);
```

```
        System.out.print(b);
```

```
        System.out.println("\n20DCS103 - RUSHIK RATHOD");
```

```
        sc.close();
```

```
    }
```

```
    public static boolean array_front9(int[] a) {
```

```
        for (int i = 0; i < a.length; i++)
```

```
        {
```

```
            if (a[i] == 9)
```

```
            {
```

```
        if(i<4)
            return true;
        else
            return false;
    }
}
return false;
}
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac String3.java

E:\Coding Projects with JAVA>java String3
Enter the size of an array : 5
Enter the array elements...
1
5
9
8
3
true
20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

I learnt to use Boolean data type which returns true or false and it is used to count the number of '9' in the first four elements of the array in the above program.

4. **Given a string, return a string where for every char in the original, there are two chars.**

double_char('The') → 'TThhee'

double_char('AAbb') → 'AAAAbbbb'

double_char('Hi-There') → 'HHii--TThheerree'

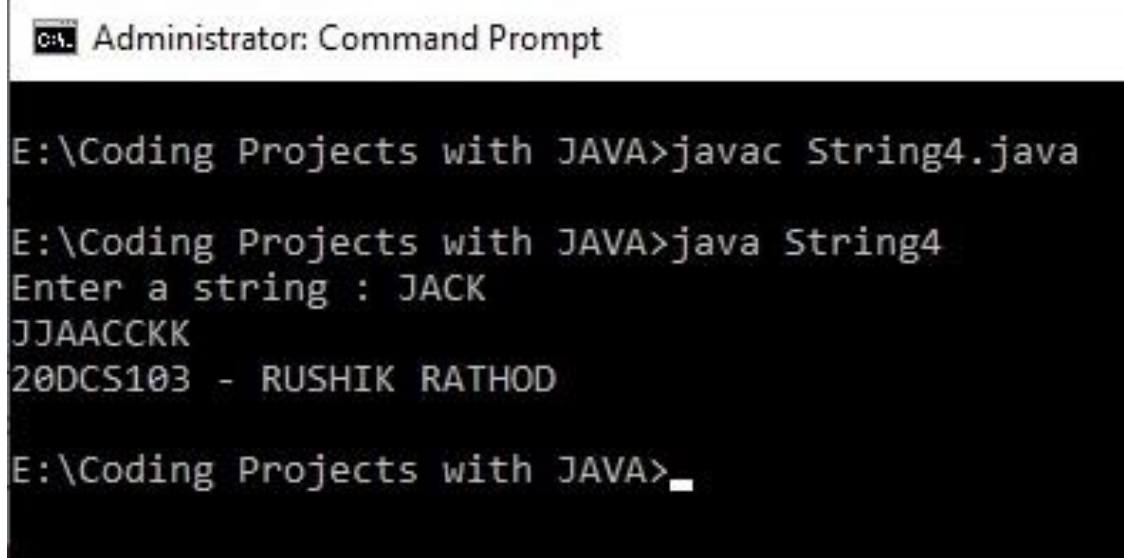
SOURCE CODE:

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

```
public class String_4 {
```

```
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        String str;  
        System.out.print("Enter a string : ");  
        str = sc.nextLine();  
        double_char(str);  
    }
```

```
    public static void double_char(String s) {  
        String temp = ""  
        for (int i = 0; i < s.length(); i++) {  
            temp = temp + s.charAt(i) + s.charAt(i);  
        }  
        System.out.print(temp);  
        System.out.println("\n20DCS103 - RUSHIK RATHOD");  
    }  
}
```


OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac String4.java

E:\Coding Projects with JAVA>java String4
Enter a string : JACK
JJAACCKK
20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

From this program, I learnt to pass the string as an argument and also learnt to use `charAt(int)` function.

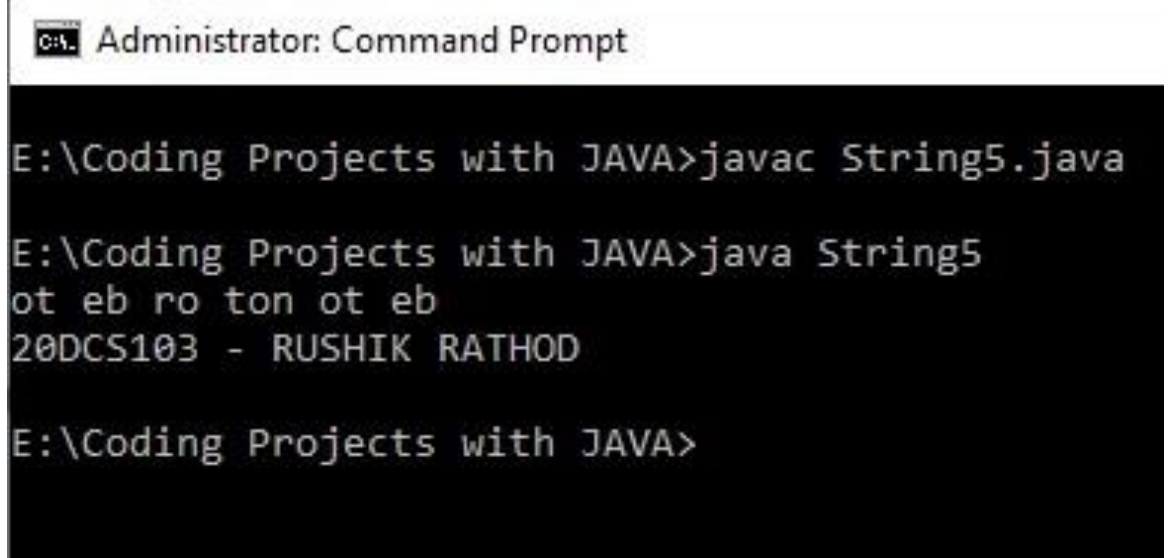
5. **Write a program that will reverse the sequence of letters in each word of your chosen paragraph. For instance, “To be or not to be” would become “oT eb ro ton ot eb”.**

SOURCE CODE:

```
import java.util.Arrays;

public class String5
{
    public static void main(String[] args)
    {
        String s = "to be or not to be";
        String[] str = s.split(" ");
        for(int i=0; i<str.length; i++)
        {
            reverse(str[i]);
            System.out.print(" ");
        }
        System.out.println("\n20DCS103 - RUSHIK RATHOD");
    }

    static void reverse(String str)
    {
        char[] ch = str.toCharArray();
        for(int i=(str.length()-1); i>=0; i--)
        {
            System.out.print(ch[i]);
        }
    }
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac String5.java

E:\Coding Projects with JAVA>java String5
ot eb ro ton ot eb
20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:


I learnt to use the split method to break the strings into the substrings. Moreover, I learnt to reverse the words of the strings by implementing various types of logics.

6. Perform following functionalities of the string:**Find Length of the String****Lowercase of the String****Uppercase of the String****Reverse String****Sort the string****SOURCE CODE:**

```
import java.util.Scanner;
import java.util.Arrays;

public class String6
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("\nEnter a string : ");
        String str1 = sc.nextLine();
        System.out.print("Length : " + str1.length());
        System.out.print("\nLowercase : " + str1.toLowerCase());
        System.out.print("\nUppercase : " + str1.toUpperCase());

        System.out.print("\nSort : ");
        char[] arr = str1.toCharArray();
        for(int i = (str1.length()-1); i>=0; i--)
        {
            System.out.print(arr[i]);
        }
        System.out.println("\n\n20DCS103 - RUSHIK RATHOD");
    }
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac String6.java

E:\Coding Projects with JAVA>java String6

Enter a string : Jack
Length : 4
Lowercase : jack
Uppercase : JACK
Sort : kcaJ

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

The length method is used to get the length of the string, toLowerCase() and toUpperCase() methods are used to get the entire string in the lower case and upper case respectively.

7. **Perform following Functionalities of the string: “CHARUSAT University”**
- **Find length**
 - **Replace ‘H’ by ‘N’**
 - **Convert all character in Uppercase**
 - **Extract and print “CHARUSAT” from given string**

SOURCE CODE:

```
import java.lang.String;
```

```
public class String7
```

```
{
```

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

```
    {
```

```
        String str = "CHARUSAT University";
```

```
        System.out.print("\nLength of the string : " + str.length());
```

```
        System.out.print("\nReplacing H by N    : " + str.replace('H','N'));
```

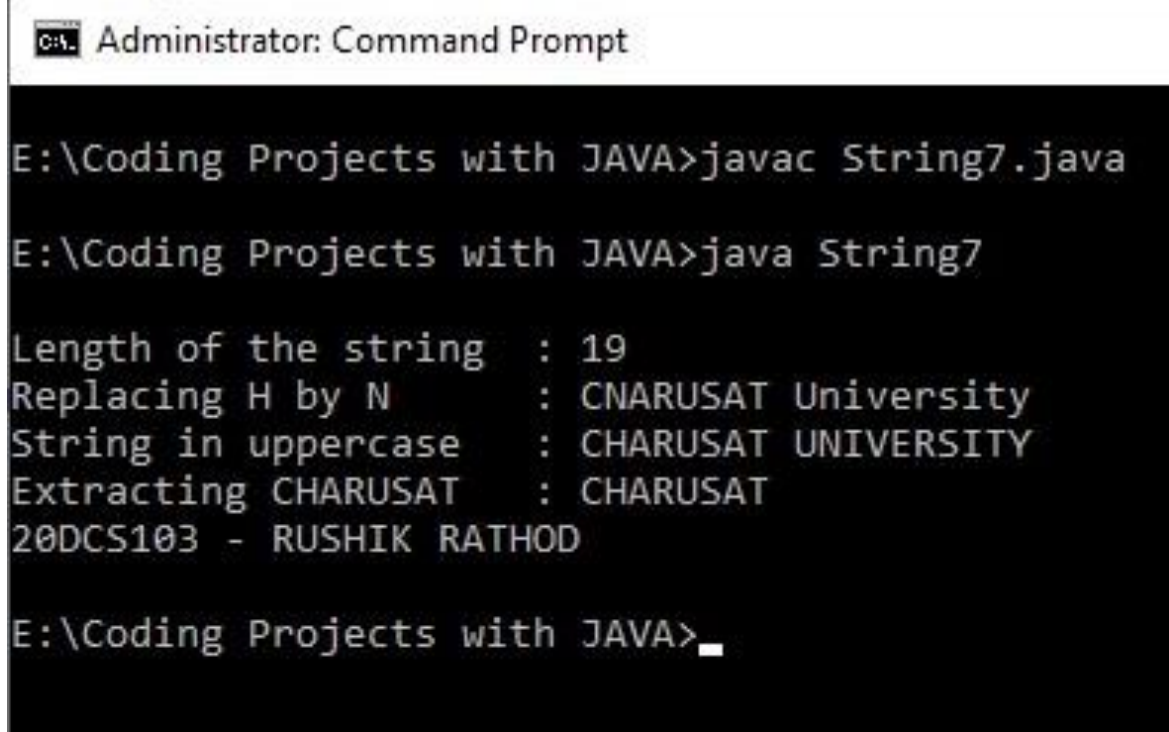
```
        System.out.print("\nString in uppercase : " + str.toUpperCase());
```

```
        System.out.print("\nExtracting CHARUSAT : " + str.substring(0,8));
```

```
        System.out.println("\n20DCS103 - RUSHIK RATHOD");
```

```
    }
```

```
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac String7.java

E:\Coding Projects with JAVA>java String7

Length of the string : 19
Replacing H by N : CNARUSAT University
String in uppcase : CHARUSAT UNIVERSITY
Extracting CHARUSAT : CHARUSAT
20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>_
```

CONCLUSION:

The `replace()` is used to replace a particular letter in a string, `toUpperCase()` method is used to get the entire string in the upper case, `substring()` method is used to get the word from the string.

PART - 3 : CONSTRUCTORS

1. **Write a java program for converting Pound into Rupees. (Accept Pounds from command line argument and using scanner class also and take 1 Pound = 100 Rupees.)**

SOURCE CODE:

```
import java.util.Scanner;

public class Constructor1
{
    Constructor1(double x)
    {
        double r = (x*100);
        System.out.println("Rupees : " + r);
    }

    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter pound : ");
        double p = sc.nextDouble();
        Constructor1 c1 = new Constructor1(p);
        System.out.println("\n20DCS103 - RUSHIK RATHOD");
    }
}
```


OUTPUT:

```
C:\> Administrator: Command Prompt

E:\Coding Projects with JAVA>javac Constructor1.java

E:\Coding Projects with JAVA>java Constructor1
Enter pound : 5
Rupees : 500.0

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

When we create an object of a class, the constructor is called automatically.
When we pass the arguments, it is called parameterized constructor.

2. **Write a program that defines TriangleArea class with three constructor. The first form accept no arguments. The second accept one double value for radius. The third form accept any two arguments.**

SOURCE CODE:

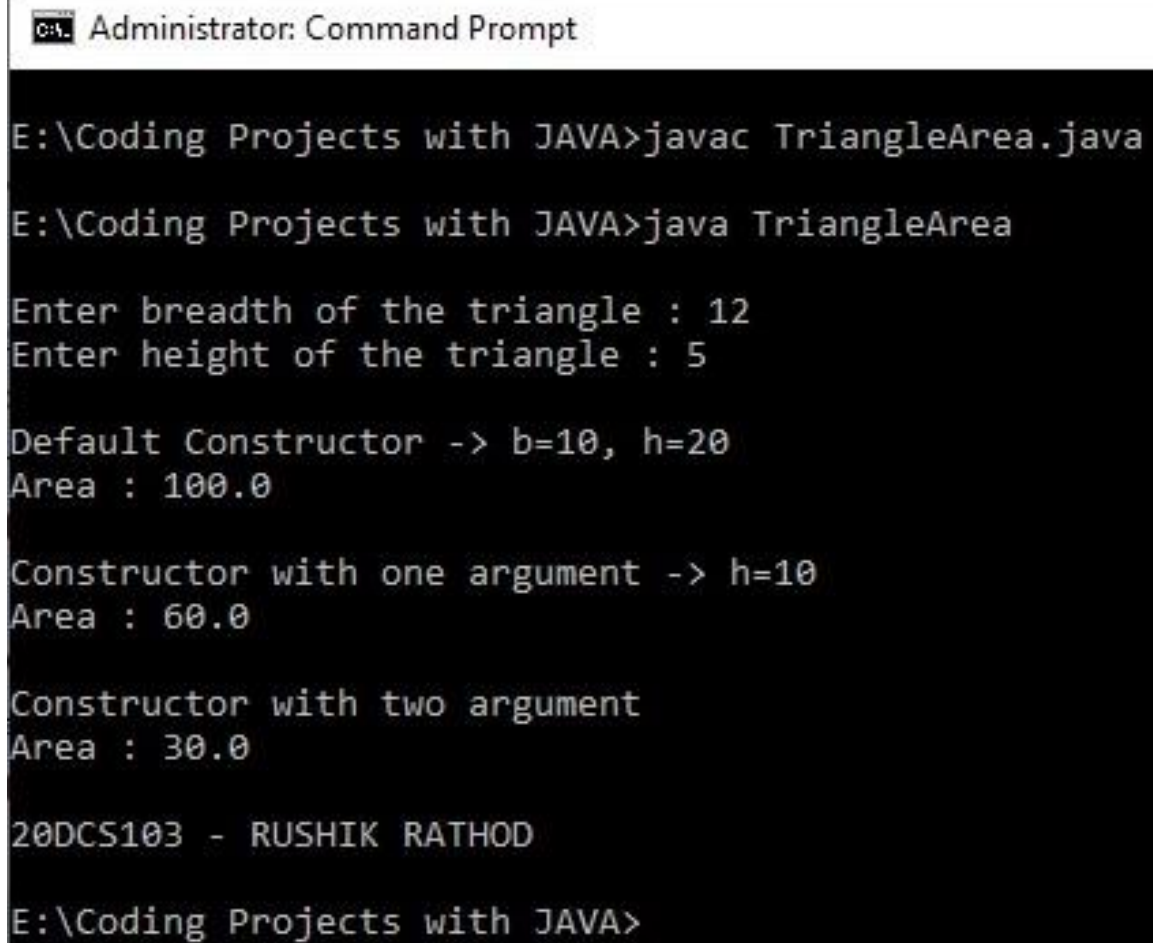
```
import java.util.Scanner;

public class TriangleArea
{
    TriangleArea()
    {
        double b = 10;
        double h = 20;
        System.out.print("\nDefault Constructor -> b=10, h=20\n");
        System.out.print("Area : " + ((b*h)/2) + "\n" );
    }
    TriangleArea(double b)
    {
        double h = 10;
        System.out.print("\nConstructor with one argument -> h=10\n");
        System.out.print("Area : " + ((b*h)/2) + "\n" );
    }
    TriangleArea(double x, double y)
    {
        System.out.print("\nConstructor with two argument\n");
        System.out.print("Area : " + ((x*y)/2) + "\n" );
    }

    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("\nEnter breadth of the triangle : ");
        double b = sc.nextDouble();
        System.out.print("Enter height of the triangle : ");
```

```
double h = sc.nextDouble();

TriangleArea t1 = new TriangleArea();
TriangleArea t2 = new TriangleArea(b);
TriangleArea t3 = new TriangleArea(b, h);
System.out.println("\n20DCS103 - RUSHIK RATHOD");
    }
}
```

OUTPUT:

```
C:\> Administrator: Command Prompt

E:\Coding Projects with JAVA>javac TriangleArea.java

E:\Coding Projects with JAVA>java TriangleArea

Enter breadth of the triangle : 12
Enter height of the triangle : 5

Default Constructor -> b=10, h=20
Area : 100.0

Constructor with one argument -> h=10
Area : 60.0

Constructor with two argument
Area : 30.0

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

When we create an object of a class, the constructor is called automatically.

When we pass the arguments to the constructors with the help of an object, it is called the parameterized constructor and all the types of constructors do not return anything.

3. **Create a class called Employee that includes three pieces of information as instance variables—a first name (type String), a last name (type String) and a monthly salary (double). Your class should have a constructor that initializes the three instance variables. Provide a set and a get method for each instance variable. If the monthly salary is not positive, set it to 0.0. Write a test application named EmployeeTest that demonstrates class Employee's capabilities. Create two Employee objects and display each object's yearly salary. Then give each Employee a 10% raise and display each Employee's yearly salary again.**

SOURCECODE:

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

```
class Employee
```

```
{
```

```
    String Name;
```

```
    String LastName;
```

```
    double Salary;
```

```
    Employee()
```

```
{
```

```
        Name = " ";
```

```
        LastName = " ";
```

```
        Salary = 0.0;
```

```
}
```

```
    Employee(String n, String l, Double s)
```

```
{
```

```
        Name = n;
```

```
        LastName = l;
```

```
        if(s>0)
```

```
            Salary = s;
```

```
        else
```

```
            Salary=0.0;
```

```
}

String getName()
{
    return Name;
}
String getLastName()
{
    return LastName;
}
Double getSalary()
{
    return Salary;
}

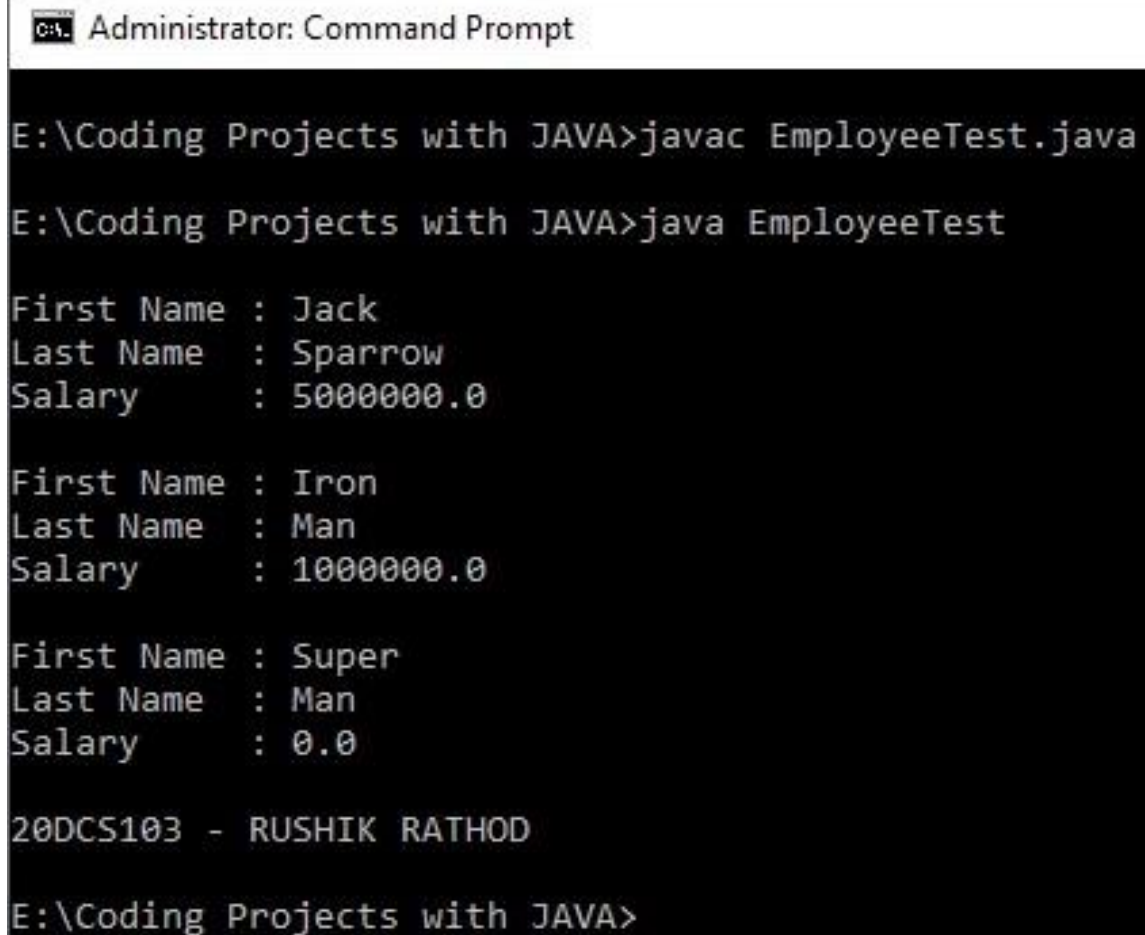
void setName(String n1)
{
    Name = n1;
}
void setLastName(String l1)
{
    LastName = l1;
}
void setSalary(Double s1)
{
    if(s1>0)
        Salary = s1;
    else
        Salary=0.0;
}

void display()
{
    System.out.println();
```

```
        System.out.println("First Name : " + getName());
        System.out.println("Last Name : " + getLastName());
        System.out.println("Salary    : " + getSalary());
    }
}

public class EmployeeTest
{
    public static void main(String[] args)
    {
        Employee e1 = new Employee("Jack", "Sparrow", 5000000.0);
        Employee e2 = new Employee("Iron", "Man", 1000000.0);

        e1.display();
        e2.display();
        e2.setName("Super");
        e2.setLastName("Man");
        e2.setSalary(-400.0);
        e2.display();
        System.out.println("\n20DCS103 - RUSHIK RATHOD");
    }
}
```

OUTPUT:

```
C:\> Administrator: Command Prompt

E:\Coding Projects with JAVA>javac EmployeeTest.java

E:\Coding Projects with JAVA>java EmployeeTest

First Name : Jack
Last Name  : Sparrow
Salary     : 5000000.0

First Name : Iron
Last Name  : Man
Salary     : 1000000.0

First Name : Super
Last Name  : Man
Salary     : 0.0

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

When we want to keep the data secure from the unauthorized personalities, the encapsulation concept plays crucial role in this program in which we have returned the safe value from a particular class.

4. **Create a class called Date that includes three pieces of information as instance variables—a month (type int), a day (type int) and a year (type int). Your class should have a constructor that initializes the three instance variables and assumes that the values provided are correct. Provide a set and a get method for each instance variable. Provide a method displayDate that displays the month, day and year separated by forward slashes (/). Write a test application named DateTest that demonstrates class Date's capabilities.**

SOURCE CODE:

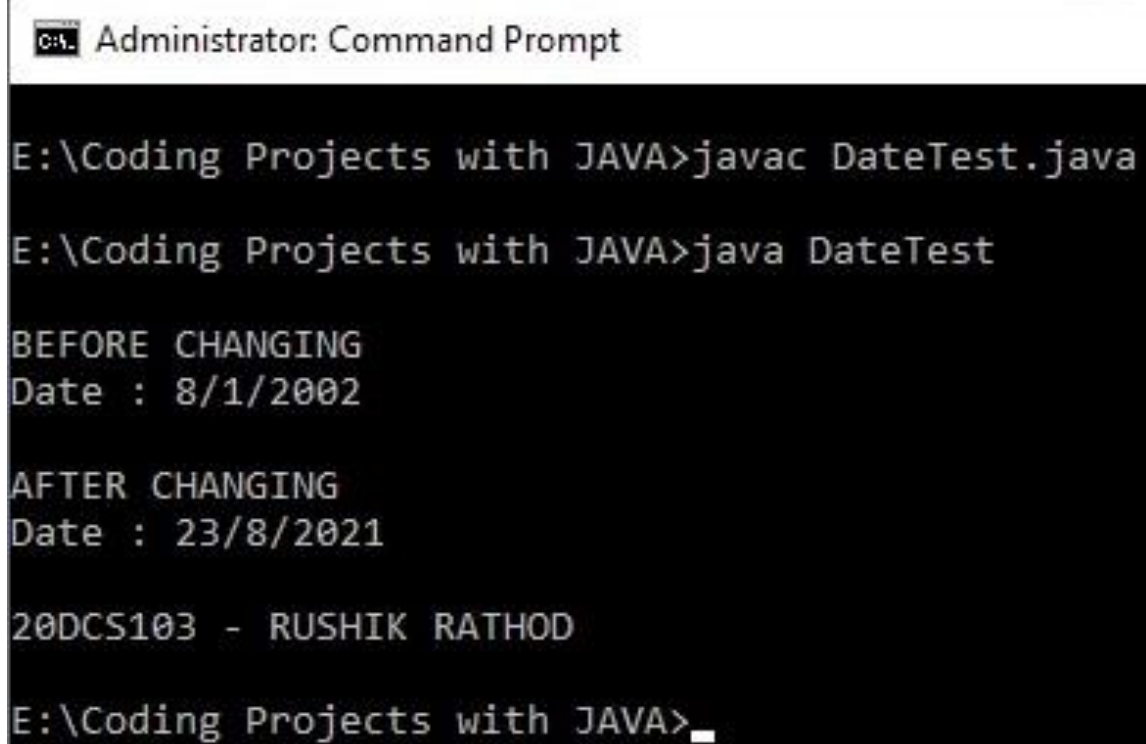
```
class Date
{
    int day;
    int month;
    int year;

    Date()
    {
        day = 0;
        month = 0;
        year = 0;
    }
    Date(int d, int m, int y)
    {
        if(d>0 && d<32)
            day = d;
        if(m<13 && m>0)
            month = m;
        if(y>0)
            year = y;
    }
    void setDay(int x)
    {
        day = x;
```

```
}  
void setMonth(int y)  
{  
    month = y;  
}  
void setYear(int z)  
{  
    year = z;  
}  
  
int getDay()  
{  
    return day;  
}  
int getMonth()  
{  
    return month;  
}  
int getYear()  
{  
    return year;  
}  
void DisplayDate()  
{  
    System.out.println("Date : " + getDay() + "/" + getMonth() + "/" +  
getYear());  
}  
}  
class DateTest  
{  
    public static void main(String[] args)  
    {  
        Date d1 = new Date(8,1,2002);  
        System.out.println("\nBEFORE CHANGING");  
    }  
}
```

```
d1.DisplayDate();

System.out.println("\nAFTER CHANGING");
d1.setDay(23);
d1.setMonth(8);
d1.setYear(2021);
d1.DisplayDate();
System.out.println("\n20DCS103 - RUSHIK RATHOD");
    }
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac DateTest.java

E:\Coding Projects with JAVA>java DateTest

BEFORE CHANGING
Date : 8/1/2002

AFTER CHANGING
Date : 23/8/2021

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>_
```

CONCLUSION:

From this program, I learnt to keep the data safe with the help of encapsulation and to access the data, return method is used.

5. **Write a program to print the area of a rectangle by creating a class named 'Area' taking the values of its length and breadth as parameters of its constructor and having a method named 'returnArea' which returns the area of the rectangle. Length and breadth of rectangle are entered through keyboard.**

SOURCECODE:

```
import java.util.Scanner;

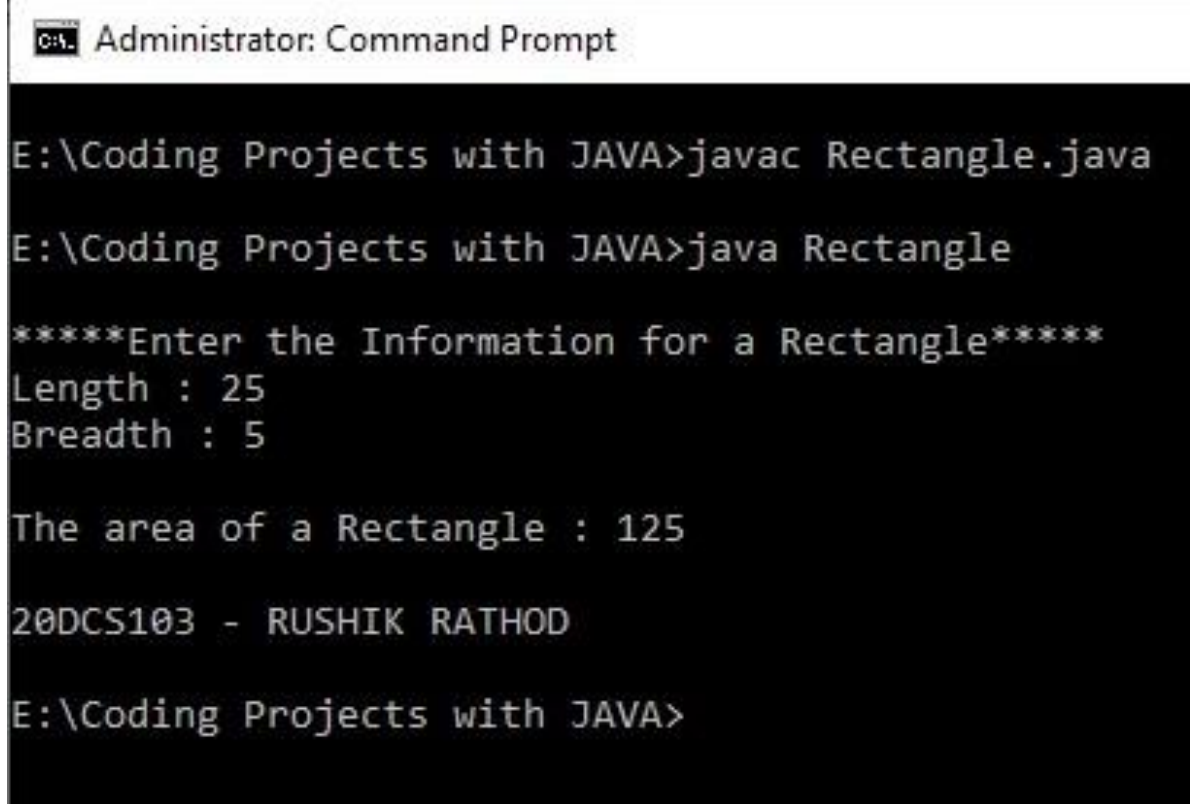
class Area
{
    int length;
    int breadth;

    Area(int l, int b)
    {
        length = l;
        breadth = b;
    }

    public int returnArea()
    {
        return length*breadth;
    }
}

public class Rectangle
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("\n*****Enter the Information for a Rectangle*****");
        System.out.print("Length : ");
        int l = sc.nextInt();
```

```
System.out.print("Breadth : ");  
int b = sc.nextInt();  
  
Area a1 = new Area(1, b);  
System.out.println("\nThe area of a Rectangle : " + a1.returnArea() );  
System.out.println("\n20DCS103 - RUSHIK RATHOD");  
}  
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac Rectangle.java

E:\Coding Projects with JAVA>java Rectangle

*****Enter the Information for a Rectangle*****
Length : 25
Breadth : 5

The area of a Rectangle : 125

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

From this practical, I learnt to create an object of a class and to pass the arguments to the constructor by calling them. Also, I learnt to keep the data secure from the unauthorized people and to access that data returnArea() method is used.

6. **Print the sum, difference and product of two complex numbers by creating a class named 'Complex' with separate methods for each operation whose real and imaginary parts are entered by user.**

SOURCECODE:

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

```
public class Complex
```

```
{
```

```
    float real, imaginary;
```

```
    void sum(Complex c1, Complex c2)
```

```
    {
```

```
        System.out.println("\nSum : " + (c1.real + c2.real) + " + " + (c1.imaginary  
+ c2.imaginary) + "i");
```

```
    }
```

```
    void difference(Complex c1, Complex c2)
```

```
    {
```

```
        System.out.println("Difference : " + (c2.real - c1.real) + " + " +  
(c2.imaginary - c1.imaginary) + "i");
```

```
    }
```

```
    void product(Complex c1, Complex c2)
```

```
    {
```

```
        System.out.println("Product : " + ((c1.real*c2.real)-  
(c1.imaginary*c2.imaginary)) + " + " + ((c1.real*c2.imaginary) +  
(c1.imaginary*c2.real)) + "i");
```

```
    }
```

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

```
    {
```

```
        Scanner sc = new Scanner(System.in);
```



```
Complex c1 = new Complex();
System.out.println("\nEnter the 1st number...");

System.out.print("Real : ");
float r1 = sc.nextFloat();
c1.real = r1;

System.out.print("Imaginary : ");
float i1 = sc.nextFloat();
c1.imaginary = i1;

Complex c2 = new Complex();
System.out.println("\nEnter the 2nd number...");

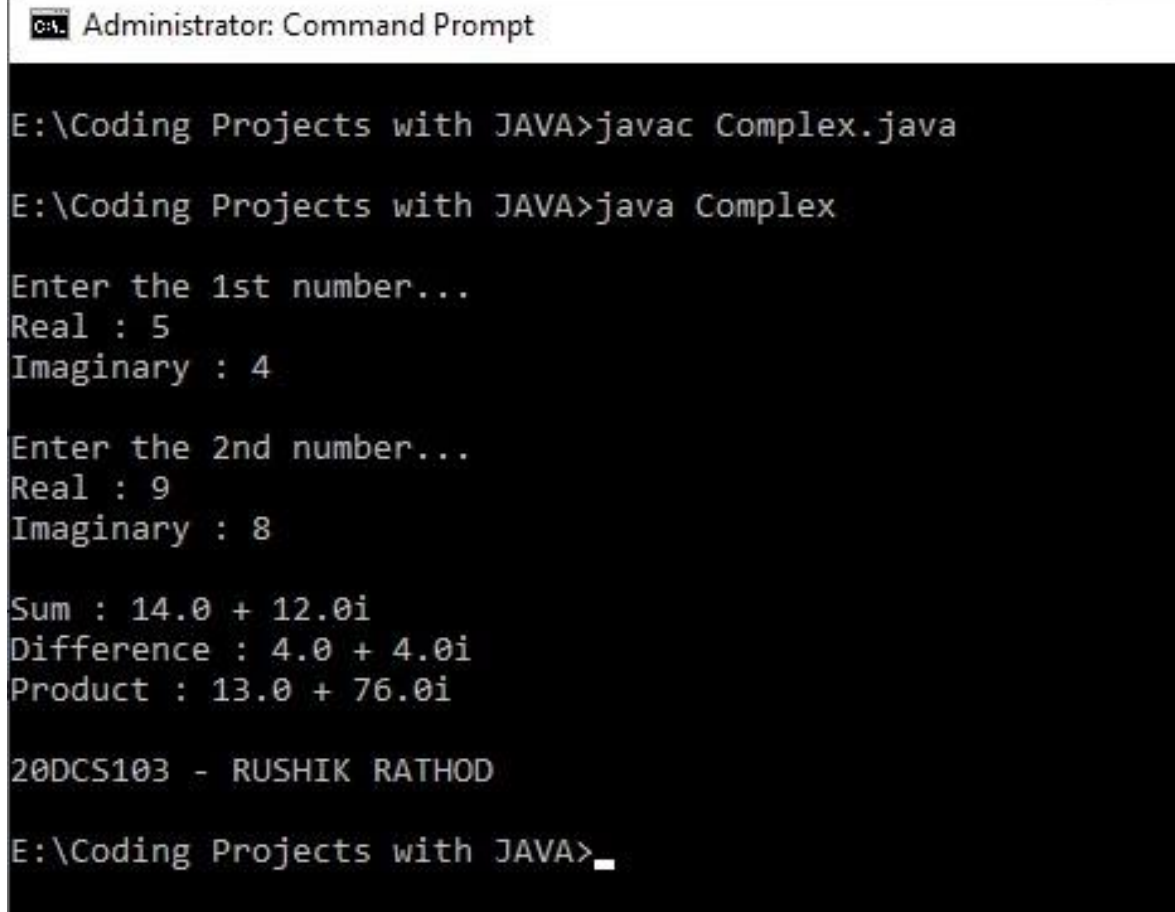
System.out.print("Real : ");
float r2 = sc.nextFloat();
c2.real = r2;

System.out.print("Imaginary : ");
float i2 = sc.nextFloat();
c2.imaginary = i2;

Complex c3 = new Complex();

c3.sum(c1,c2);
c3.difference(c1,c2);
c3.product(c1,c2);

System.out.println("\n20DCS103 - RUSHIK RATHOD");
    }
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac Complex.java

E:\Coding Projects with JAVA>java Complex

Enter the 1st number...
Real : 5
Imaginary : 4

Enter the 2nd number...
Real : 9
Imaginary : 8

Sum : 14.0 + 12.0i
Difference : 4.0 + 4.0i
Product : 13.0 + 76.0i

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>_
```

CONCLUSION:

I learnt to call the methods using the objects of a particular class and also learnt to print the summation, subtraction and product of a complex numbers with the JAVA language.

7. **Complete the code and write main () method to execute program.**

SOURCECODE:

```
class MethodOverloading
{
    private void methodOverloaded()
    {
        System.out.println("\nPrivate Method with no argument.");
    }

    private int methodOverloaded(int i)
    {
        System.out.println("\nPrivate Method with one argument.");
        return i;
    }

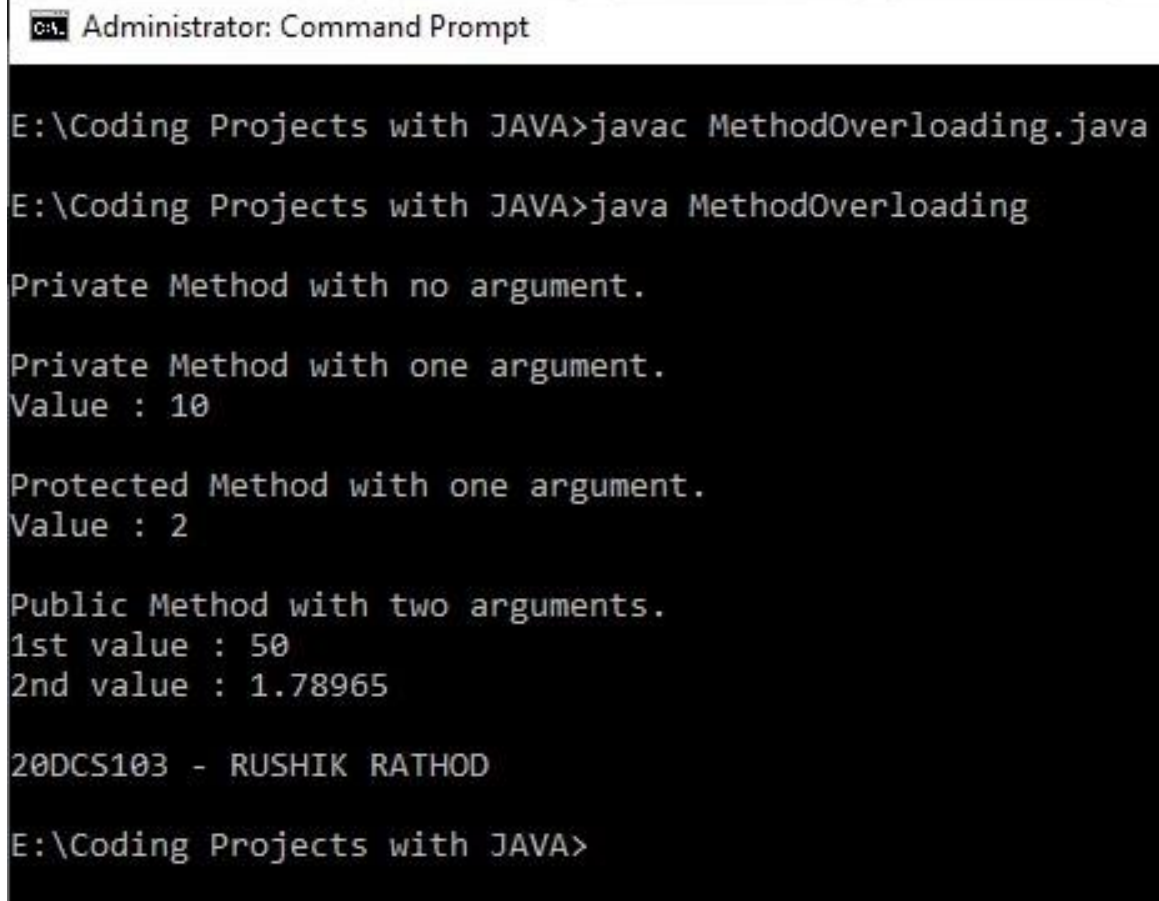
    protected int methodOverloaded(double d)
    {
        System.out.println("\nProtected Method with one argument.");
        int i = (int)d;
        return i;
    }

    public void methodOverloaded(int i, double d)
    {
        System.out.println("\nPublic Method with two arguments.");
        System.out.println("1st value : " + i + "\n2nd value : " + d );
    }

    public static void main(String[] args)
    {
        MethodOverloading m1 = new MethodOverloading();

        m1.methodOverloaded();
    }
}
```

```
System.out.println("Value : " + m1.methodOverloaded(10));  
System.out.println("Value : " + m1.methodOverloaded(2.3456));  
  
m1.methodOverloaded(50, 1.78965);  
  
System.out.println("\n20DCS103 - RUSHIK RATHOD");  
}  
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac MethodOverloading.java
E:\Coding Projects with JAVA>java MethodOverloading

Private Method with no argument.

Private Method with one argument.
Value : 10

Protected Method with one argument.
Value : 2

Public Method with two arguments.
1st value : 50
2nd value : 1.78965

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

From this program, I learnt the method overloading concept by calling them with the help of the object of a class. Moreover, I learn the type conversion from double to integer.

PART - 4 : INHERITANCE, INTERFACE, PACKAGE

1. **Create a class with a method that prints "This is parent class" and its subclass with another method that prints "This is child class".**

Now, create an object for each of the class and call

1 - method of parent class by object of parent class

2 - method of child class by object of child class

3 - method of parent class by object of child class

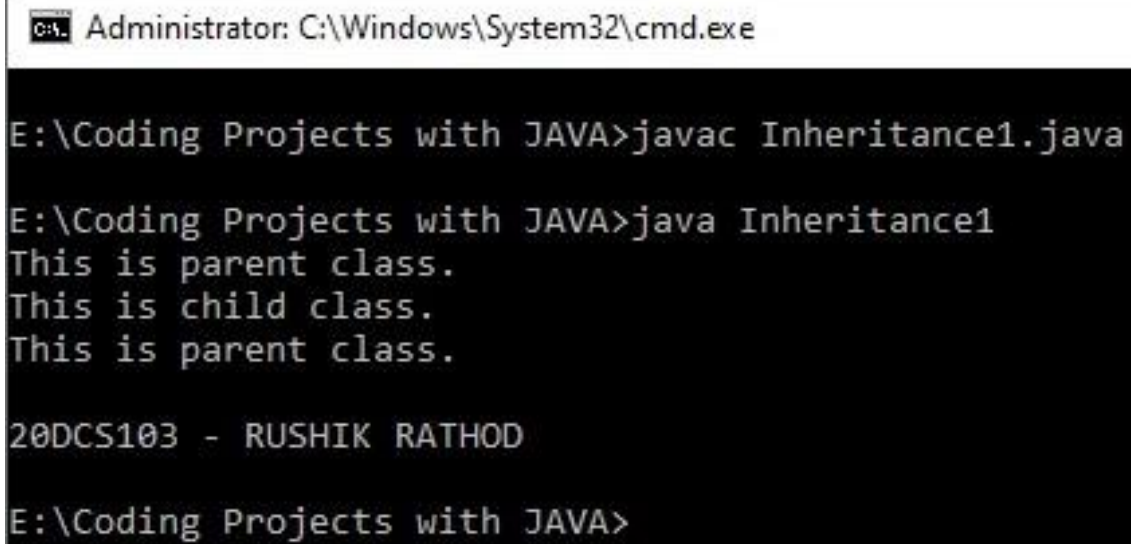
SOURCECODE:

```
class Parent {
    void print_parent() {
        System.out.println("This is parent class.");
    }
}

class Child extends Parent {
    void print_child() {
        System.out.println("This is child class.");
    }
}

public class Inheritance1 {
    public static void main(String[] args) {

        Parent p1 = new Parent();
        p1.print_parent();
        Child c1 = new Child();
        c1.print_child();
        c1.print_parent();
        System.out.println("\n20DCS103 - RUSHIK RATHOD");
    }
}
```

OUTPUT:

The screenshot shows a Windows command prompt window titled "Administrator: C:\Windows\System32\cmd.exe". The prompt is at "E:\Coding Projects with JAVA>". The user has entered "javac Inheritance1.java" and "java Inheritance1". The output of the program is displayed as follows:

```
E:\Coding Projects with JAVA>javac Inheritance1.java
E:\Coding Projects with JAVA>java Inheritance1
This is parent class.
This is child class.
This is parent class.

20DCS103 - RUSHIK RATHOD
E:\Coding Projects with JAVA>
```

CONCLUSION:

From this program, I learnt to implement the inheritance concept. Additionally, I learnt to call the method of a parent class with the help of child class's object and with the respective object as well.

2. **Create a class named 'Member' having the following members:**
Data members 1 – Name
2 – Age
3 - Phone number
4 – Address
5 – Salary
It also has a method named 'printSalary' which prints the salary of the members.
Two classes 'Employee' and 'Manager' inherits the 'Member' class. The 'Employee' and 'Manager' classes have data members 'specialization' and 'department' respectively. Now, assign name, age, phone number, address and salary to an employee and a manager by making an object of both of these classes and print the same.

SOURCECODE:

```
class Member {  
    String name;  
    int age;  
    String phoneNumber;  
    String address;  
    float salary;  
  
    void printSalary() {  
        System.out.print("Salary : " + salary);  
    }  
}  
  
class Employee extends Member {  
    String specialization;  
}  
  
class Manager extends Member {  
    String department;  
}
```



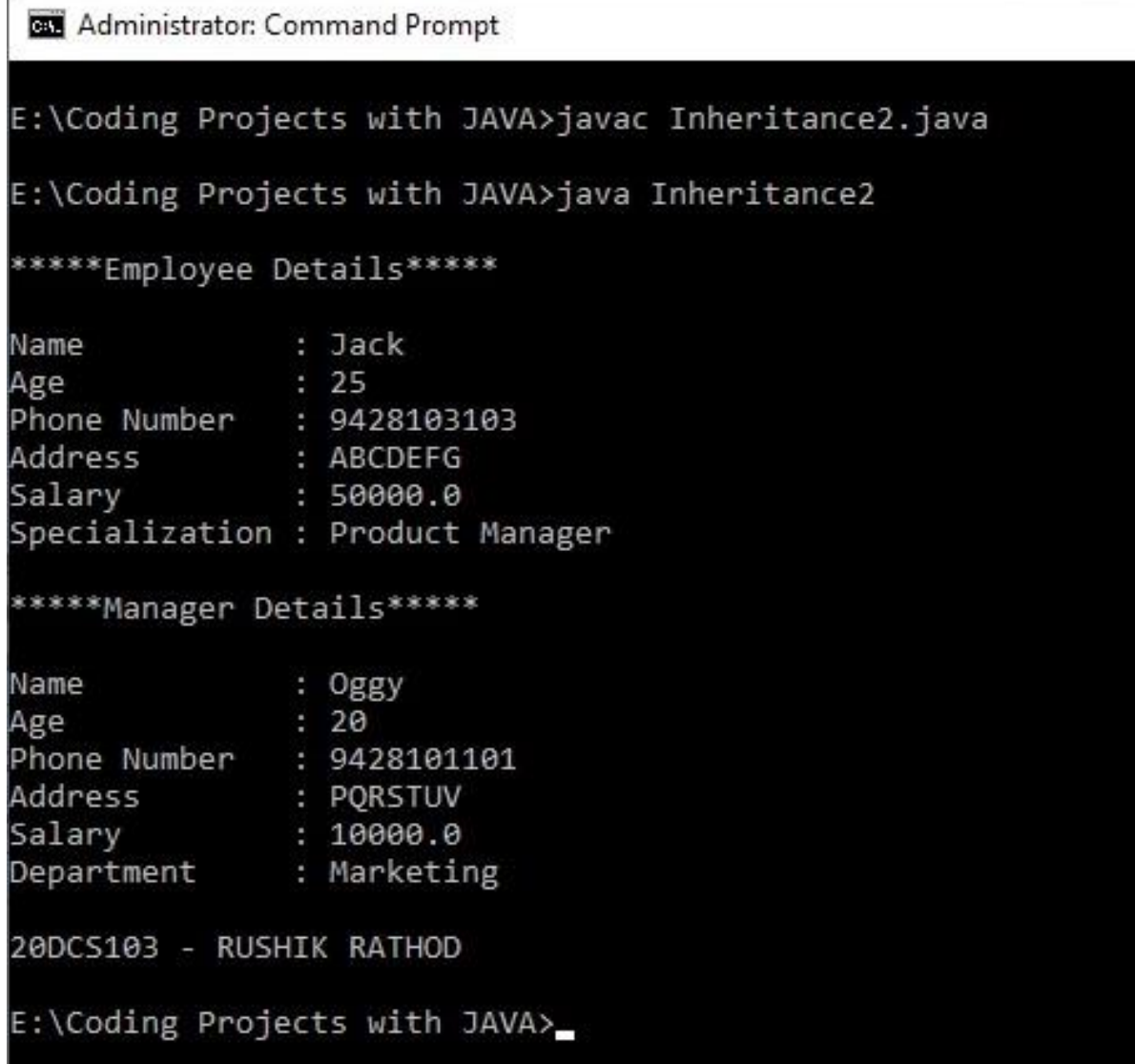
```
public class Inheritance2 {
    public static void main(String[] args)
    {
        Employee e1 = new Employee();
        e1.name = "Jack";
        e1.age = 25;
        e1.phoneNumber = "9428103103";
        e1.address = "ABCDEFGF";
        e1.salary = 50000;
        e1.specialization = "Product Manager";

        Manager m1 = new Manager();
        m1.name = "Oggy";
        m1.age = 20;
        m1.phoneNumber = "9428101101";
        m1.address = "PQRSTUV";
        m1.salary = 10000;
        m1.department = "Marketing";

        System.out.println("\n*****Employee Details*****\n");
        System.out.println("Name      : " + e1.name);
        System.out.println("Age      : " + e1.age);
        System.out.println("Phone Number : " + e1.phoneNumber);
        System.out.println("Address    : " + e1.address);
        System.out.println("Salary     : " + e1.salary);
        System.out.println("Specialization : " + e1.specialization);

        System.out.println("\n*****Manager Details*****\n");
        System.out.println("Name      : " + m1.name);
        System.out.println("Age      : " + m1.age);
        System.out.println("Phone Number : " + m1.phoneNumber);
        System.out.println("Address    : " + m1.address);
        System.out.println("Salary     : " + m1.salary);
```

```
        System.out.println("Department    : " + m1.department);  
        System.out.println("\n20DCS103 - RUSHIK RATHOD");  
    }  
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac Inheritance2.java

E:\Coding Projects with JAVA>java Inheritance2

*****Employee Details*****

Name           : Jack
Age            : 25
Phone Number   : 9428103103
Address        : ABCDEFG
Salary         : 50000.0
Specialization : Product Manager

*****Manager Details*****

Name           : Oggy
Age            : 20
Phone Number   : 9428101101
Address        : PQRSTUV
Salary         : 10000.0
Department     : Marketing

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>_
```

CONCLUSION:

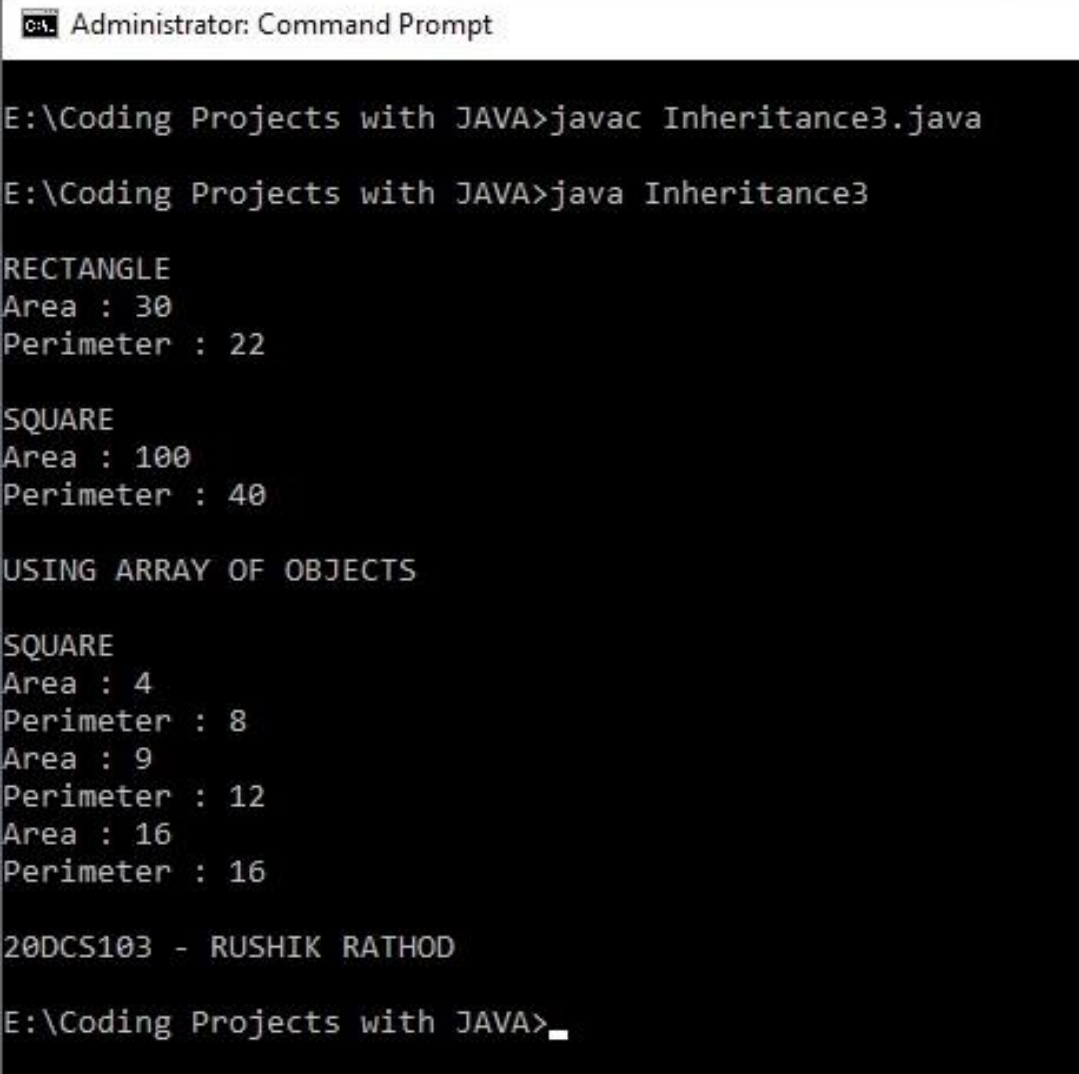
The 'extends' keyword is used after class name to make that class inherited from the parent class. Also, I learn to call the methods of the parent class with the help of the object of the child class.

3. **Create a class named 'Rectangle' with two data members 'length' and 'breadth' and two methods to print the area and perimeter of the rectangle respectively. Its constructor having parameters for length and breadth is used to initialize length and breadth of the rectangle. Let class 'Square' inherit the 'Rectangle' class with its constructor having a parameter for its side (suppose s) calling the constructor of its parent class as 'super(s,s)'. Print the area and perimeter of a rectangle and a square. Also use array of objects.**

SOURCECODE:

```
class Rectangle
{
    int length;
    int breadth;
    public Rectangle(int l, int b)    //Constructor
    {
        length = l;
        breadth = b;
    }
    public void area()
    {
        System.out.println("Area : " + length*breadth);
    }
    public void perimeter()
    {
        System.out.println("Perimeter : " + 2*(length + breadth));
    }
}
class Square extends Rectangle
{
    int side;
    public Square(int s)    //Constructor
    {
        super(s,s);
    }
}
```

```
    }  
}  
  
public class Inheritance3 {  
    public static void main(String[] args)  
    {  
        Rectangle r1 = new Rectangle(5, 6);  
        System.out.println("\nRECTANGLE");  
        r1.area();  
        r1.perimeter();  
  
        Square s1 = new Square(10);  
        System.out.println("\nSQUARE");  
        s1.area();  
        s1.perimeter();  
  
        Square[] sq=new Square[5];  
        int k=2;  
        System.out.println("\nUSING ARRAY OF OBJECTS");  
        System.out.println("\nSQUARE");  
        for(int i=0;i<3;i++)  
        {  
            sq[i]=new Square(k);  
            k++;  
        }  
  
        for(int i=0;i<3;i++)  
        {  
            sq[i].area();  
            sq[i].perimeter();  
        }  
        System.out.println("\n20DCS103 - RUSHIK RATHOD");  
    }  
}
```

OUTPUT:

```
C:\> Administrator: Command Prompt

E:\Coding Projects with JAVA>javac Inheritance3.java
E:\Coding Projects with JAVA>java Inheritance3

RECTANGLE
Area : 30
Perimeter : 22

SQUARE
Area : 100
Perimeter : 40

USING ARRAY OF OBJECTS

SQUARE
Area : 4
Perimeter : 8
Area : 9
Perimeter : 12
Area : 16
Perimeter : 16

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>_
```

CONCLUSION:

From this program, I learn to inherit the class from another class. The `super()` keyword is used to invoke the immediate parent class method.

4. **Create a class named 'Shape' with a method to print "This is This is shape". Then create two other classes named 'Rectangle', 'Circle' inheriting the Shape class, both having a method to print "This is rectangular shape" and "This is circular shape" respectively. Create a subclass 'Square' of 'Rectangle' having a method to print "Square is a rectangle". Now call the method of 'Shape' and 'Rectangle' class by the object of 'Square' class.**

SOURCECODE:

```
class Shape
{
    public void printShape()
    {
        System.out.println("\nThis is shape.");
    }
}

class Circle extends Shape
{
    public void printCircle()
    {
        System.out.println("This is circular shape.");
    }
}

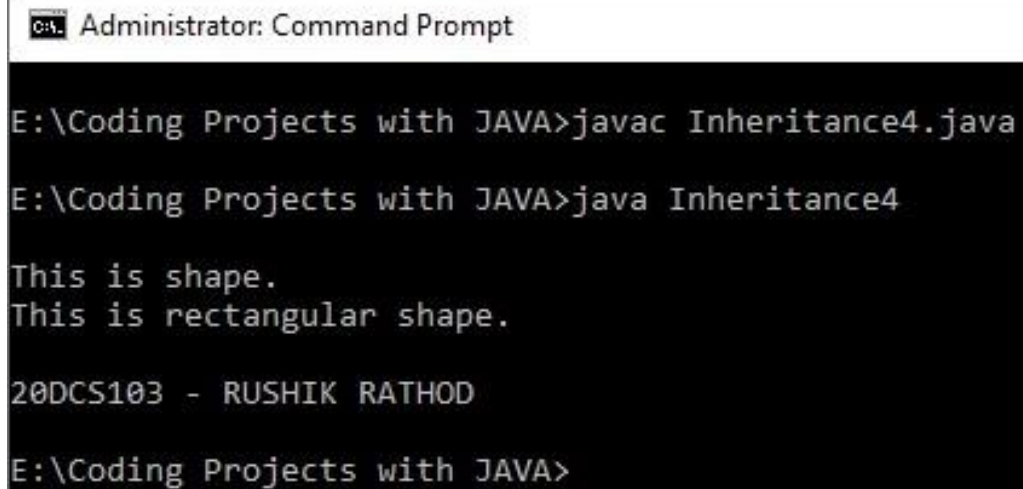
class Rectangle extends Shape
{
    public void printRectangle()
    {
        System.out.println("This is rectangular shape.");
    }
}

class Square extends Rectangle
{

```

```
public void printSquare()
{
    System.out.println("Square is a rectangle.");
}

public class Inheritance4
{
    public static void main(String[] args)
    {
        Square s1 = new Square();
        s1.printShape();
        s1.printRectangle();
        System.out.println("\n20DCS103 - RUSHIK RATHOD");
    }
}
```


OUTPUT:

The screenshot shows a Windows Command Prompt window titled "Administrator: Command Prompt". The command prompt is at the directory "E:\Coding Projects with JAVA". The user has entered the command "javac Inheritance4.java" and then "java Inheritance4". The output of the program is displayed as follows:

```
E:\Coding Projects with JAVA>javac Inheritance4.java
E:\Coding Projects with JAVA>java Inheritance4

This is shape.
This is rectangular shape.

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

In this program 'extends' keyword is used to inherit the class from another class. Further, I learnt to call the methods of parent class with the help of the objects of the super sub class.

5. **Create a class 'Degree' having a method 'getDegree' that prints "I got a degree". It has two subclasses namely 'Undergraduate' and 'Postgraduate' each having a method with the same name that prints "I am an Undergraduate" and "I am a Postgraduate" respectively. Call the method by creating an object of each of the three classes.**

SOURCECODE:

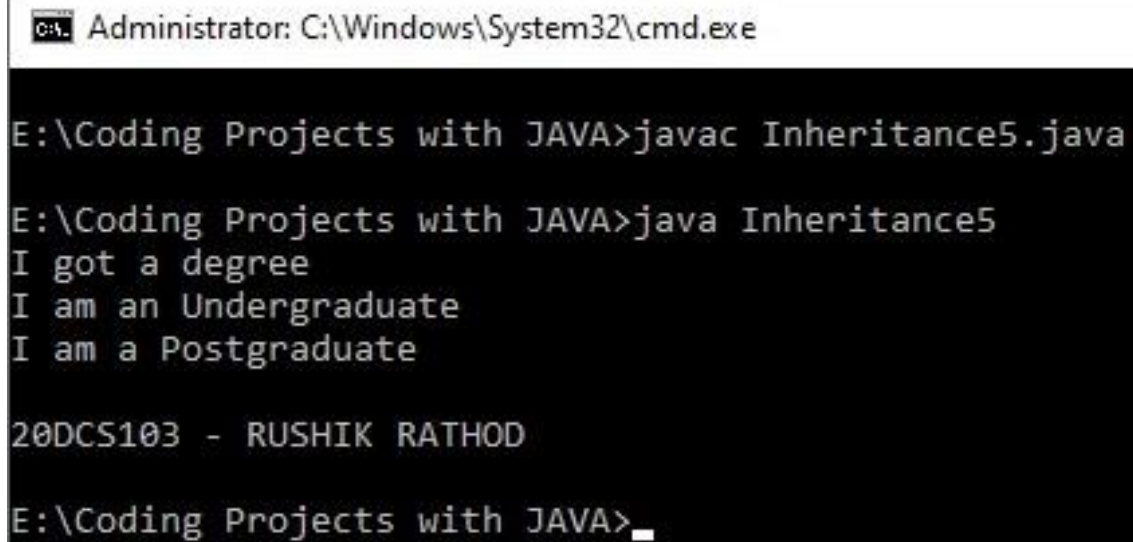
```
class Degree
{
    void getDegree()
    {
        System.out.println("I got a degree");
    }
}

class Undergraduate extends Degree
{
    void getDegree()
    {
        System.out.println("I am an Undergraduate");
    }
}

class Postgraduate extends Degree
{
    void getDegree()
    {
        System.out.println("I am a Postgraduate");
    }
}

public class Inheritance5
{
    public static void main(String[] args)
```

```
{  
    Degree d = new Degree();  
    d.getDegree();  
    Undergraduate u = new Undergraduate();  
    u.getDegree();  
    Postgraduate p = new Postgraduate();  
    p.getDegree();  
    System.out.println("\n20DCS103 - RUSHIK RATHOD");  
}  
}
```

OUTPUT:

The screenshot shows a Windows command prompt window titled "Administrator: C:\Windows\System32\cmd.exe". The prompt is at "E:\Coding Projects with JAVA>". The user enters "javac Inheritance5.java" and then "java Inheritance5". The program outputs three lines: "I got a degree", "I am an Undergraduate", and "I am a Postgraduate". Below this, the text "20DCS103 - RUSHIK RATHOD" is displayed. The prompt returns to "E:\Coding Projects with JAVA>".

```
Administrator: C:\Windows\System32\cmd.exe

E:\Coding Projects with JAVA>javac Inheritance5.java

E:\Coding Projects with JAVA>java Inheritance5
I got a degree
I am an Undergraduate
I am a Postgraduate

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>_
```

CONCLUSION:

The 'extends' keyword is used to inherit the class from another class in JAVA language. From this program, I learnt to call the methods of parent class and two sub classes by their respective objects.

6. **Write a java that implements an interface AdvancedArithmetic which contains amethod signature int divisor_sum(int n). You need to write a class calledMyCalculator which implements the interface. divisorSum function just takes an integer as input and return the sum of all its divisors. For example divisors of 6 are 1, 2, 3 and 6, so divisor_sum should return 12. The value of n will be at most 1000.**

SOURCECODE:

```
import java.util.Scanner;

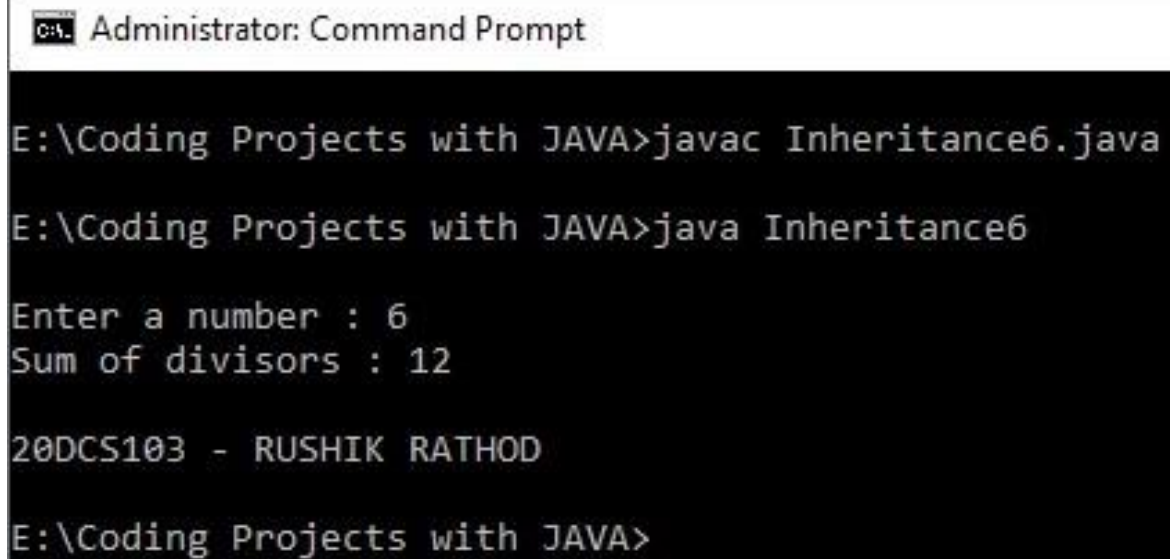
interface AdvancedArithmetic {
    int divisor_sum(int n);
}

class MyCalculator implements AdvancedArithmetic
{

    public int divisor_sum(int x)
    {
        int result = 0;
        for (int i = 1; i <= x; i++)
        {
            if (x % i == 0)
            {
                result = result + i;
            }
        }
        return result;
    }
}

public class Inheritance6
{
    public static void main(String[] args)
```

```
{  
    Scanner sc = new Scanner(System.in);  
    System.out.print("\nEnter a number : ");  
    int x = sc.nextInt();  
  
    MyCalculator m = new MyCalculator();  
    int result = m.divisor_sum(x);  
    System.out.println("Sum of divisors : " + result);  
    System.out.println("\n20DCS103 - RUSHIK RATHOD");  
}  
}
```

OUTPUT:

```
C:\> Administrator: Command Prompt

E:\Coding Projects with JAVA>javac Inheritance6.java

E:\Coding Projects with JAVA>java Inheritance6

Enter a number : 6
Sum of divisors : 12

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>
```

CONCLUSION:

Here, interface is used to calculate the sum of all the divisors of a given number. An interface is a reference type and it is a collection of abstract methods which has no body.

7. Assume you want to capture shapes, which can be either circles (with a radius and a color) or rectangles (with a length, width, and color). You also want to be able to create signs (to post in the campus center, for example), each of which has a shape (for the background of the sign) and the text (a String) to put on the sign.

SOURCECODE:

```
interface Shape_intf {  
    void Shape();  
    void Area();  
}
```

```
interface Sign_intf {  
    void sign(String S);  
}
```

```
class Circle implements Shape_intf {  
    int radius;  
    String color;  
  
    Circle() {  
        radius = 50;  
        color = "Green";  
    }  
  
    public void Shape() {  
        System.out.println("\nShape : Circle " + "\nColor : " + color + " \nRadius :  
" + radius);  
    }  
  
    public void Area() {  
        System.out.println("Area : " + (3.14 * radius * radius));  
    }  
}
```



```
}

class Rectangle implements Shape_intf {
    int length;
    int width;
    String color;

    Rectangle() {
        length = 20;
        width = 10;
        color = "Red";
    }

    public void Shape() {
        System.out.println("\nShape : Rectangle " + "\nColor : " + color + "
\nLength : " + length + " \nWidth : " + width);
    }

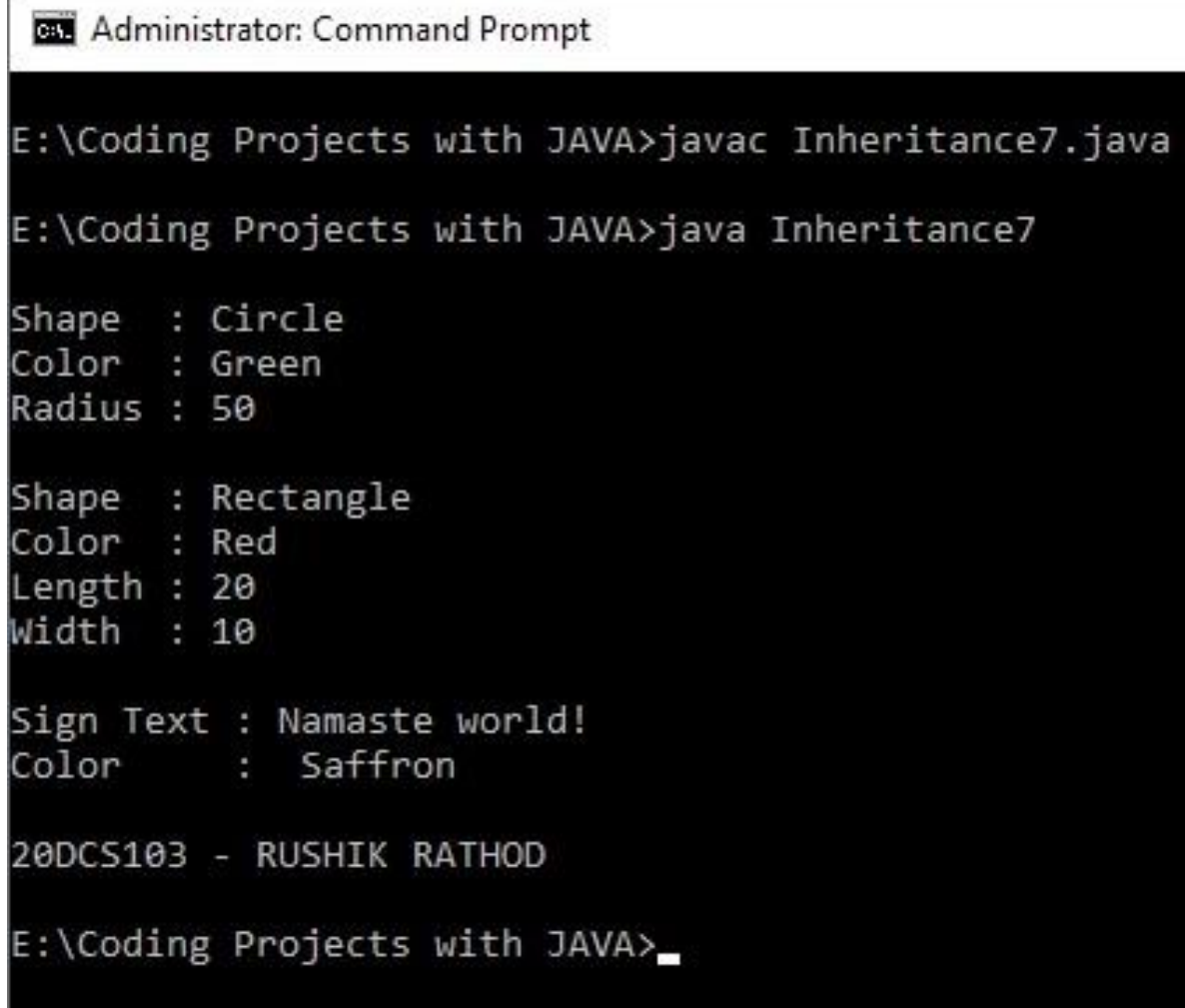
    public void Area() {
        System.out.println("Area : " + (length * width));
    }
}

class Sign implements Sign_intf {
    String Color = "Saffron";
    String Text = "Namaste";

    public void sign(String Text) {
        System.out.println("\nSign Text : " + Text + " \nColor : " + Color);
    }
}

public class Inheritance7 {
    public static void main(String[] args) {
```

```
Circle c1 = new Circle();  
c1.Shape();  
Rectangle r1 = new Rectangle();  
r1.Shape();  
Sign s1 = new Sign();  
s1.sign("Namaste world!");  
System.out.println("\n20DCS103 - RUSHIK RATHOD");  
}  
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac Inheritance7.java

E:\Coding Projects with JAVA>java Inheritance7

Shape : Circle
Color : Green
Radius : 50

Shape : Rectangle
Color : Red
Length : 20
Width : 10

Sign Text : Namaste world!
Color : Saffron

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>_
```

CONCLUSION:

An interface is a collection of abstract methods which is used and implemented on the different classes in this program. We can override the abstract method by just defining again in the implemented classes.

8. **Create classes and interfaces for circles, rectangles, shapes, and signs. Write a program that illustrates the significance of interface default method.**

SOURCECODE:

```
interface circle_interface
{
    default void show()
    {
        System.out.println("DEFAULT METHOD FROM circle");
    }

    void display();
}

interface rectangle_interface
{
    default void show()
    {
        System.out.println("DEFAULT METHOD FROM rectangle");
    }

    void display();
}

interface shape_interface
{
    default void show()
    {
        System.out.println("DEFAULT METHOD FROM shape");
    }

    void display();
}
```

```
interface sign_interface
{
    default void show()
    {
        System.out.println("DEFAULT METHOD FROM sign");
    }

    void display();
}

class Circle implements circle_interface
{
    public void display()
    {
        System.out.println("METHOD OF CLASS circle WHICH IMPLEMENTS
circle_interface");
    }
}

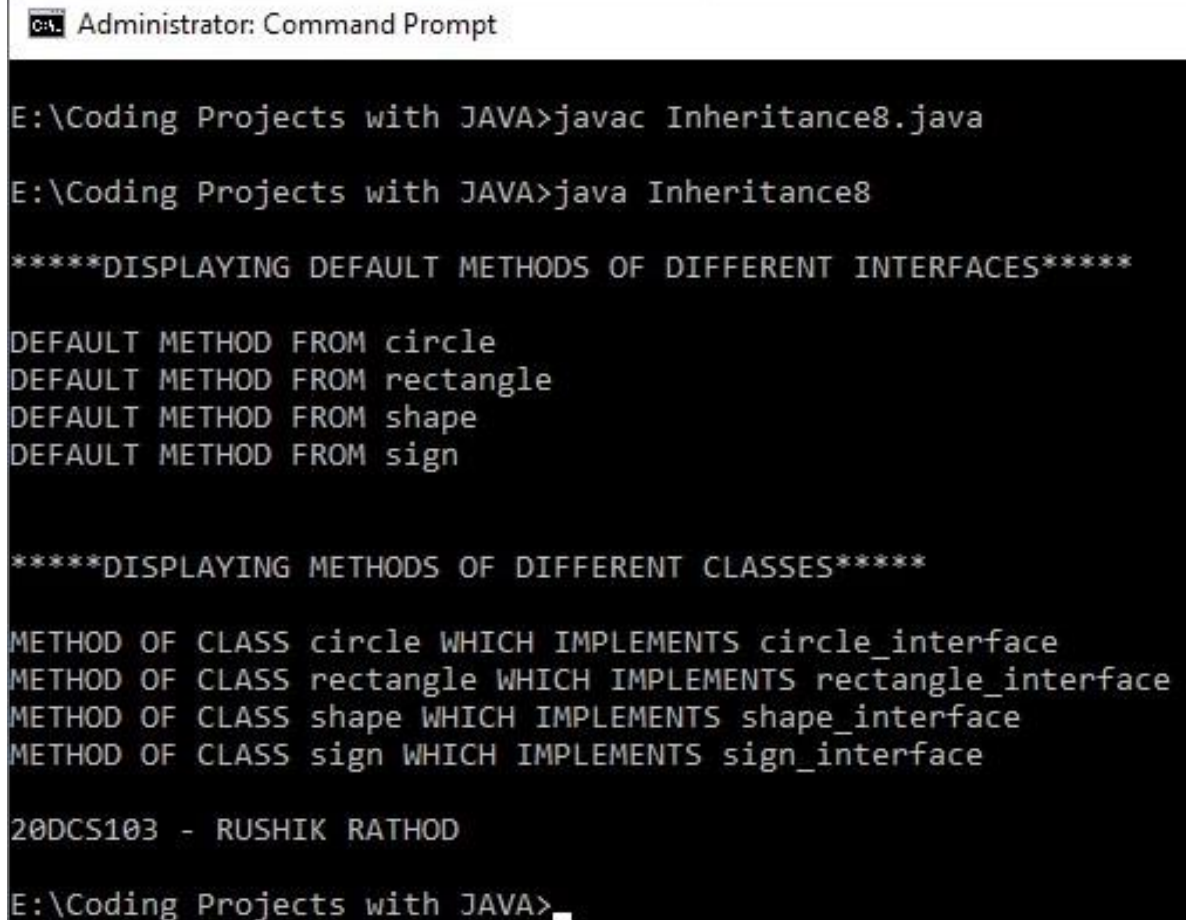
class Rectangle implements rectangle_interface
{
    public void display()
    {
        System.out.println("METHOD OF CLASS rectangle WHICH
IMPLEMENTS rectangle_interface");
    }
}

class Shape implements shape_interface
{
    public void display()
    {
        System.out.println("METHOD OF CLASS shape WHICH IMPLEMENTS
shape_interface");
    }
}
```

```
class Sign implements sign_interface
{
    public void display()
    {
        System.out.println("METHOD OF CLASS sign WHICH IMPLEMENTS
sign_interface");
    }
}
class Inheritance8
{
    public static void main(String[] args)
    {
        Circle c1 = new Circle();
        Rectangle r1 = new Rectangle();
        Shape s1 = new Shape();
        Sign sn1 = new Sign();

        System.out.println("\n*****DISPLAYING DEFAULT METHODS OF
DIFFERENT INTERFACES*****\n");
        c1.show();
        r1.show();
        s1.show();
        sn1.show();
        System.out.println();

        System.out.println("\n*****DISPLAYING METHODS OF DIFFERENT
CLASSES*****\n");
        c1.display();
        r1.display();
        s1.display();
        sn1.display();
        System.out.println("\n20DCS103 - RUSHIK RATHOD");
    }
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA>javac Inheritance8.java

E:\Coding Projects with JAVA>java Inheritance8

*****DISPLAYING DEFAULT METHODS OF DIFFERENT INTERFACES*****

DEFAULT METHOD FROM circle
DEFAULT METHOD FROM rectangle
DEFAULT METHOD FROM shape
DEFAULT METHOD FROM sign

*****DISPLAYING METHODS OF DIFFERENT CLASSES*****

METHOD OF CLASS circle WHICH IMPLEMENTS circle_interface
METHOD OF CLASS rectangle WHICH IMPLEMENTS rectangle_interface
METHOD OF CLASS shape WHICH IMPLEMENTS shape_interface
METHOD OF CLASS sign WHICH IMPLEMENTS sign_interface

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA>_
```

CONCLUSION:

From this practical, I learnt to make an interface for a program and also learnt to implement it on a class to get the quick and easy creation of the entire program.

9. **Write a java program which shows importing of classes from other user define packages.**

SOURCECODE:

```
//importing package
```

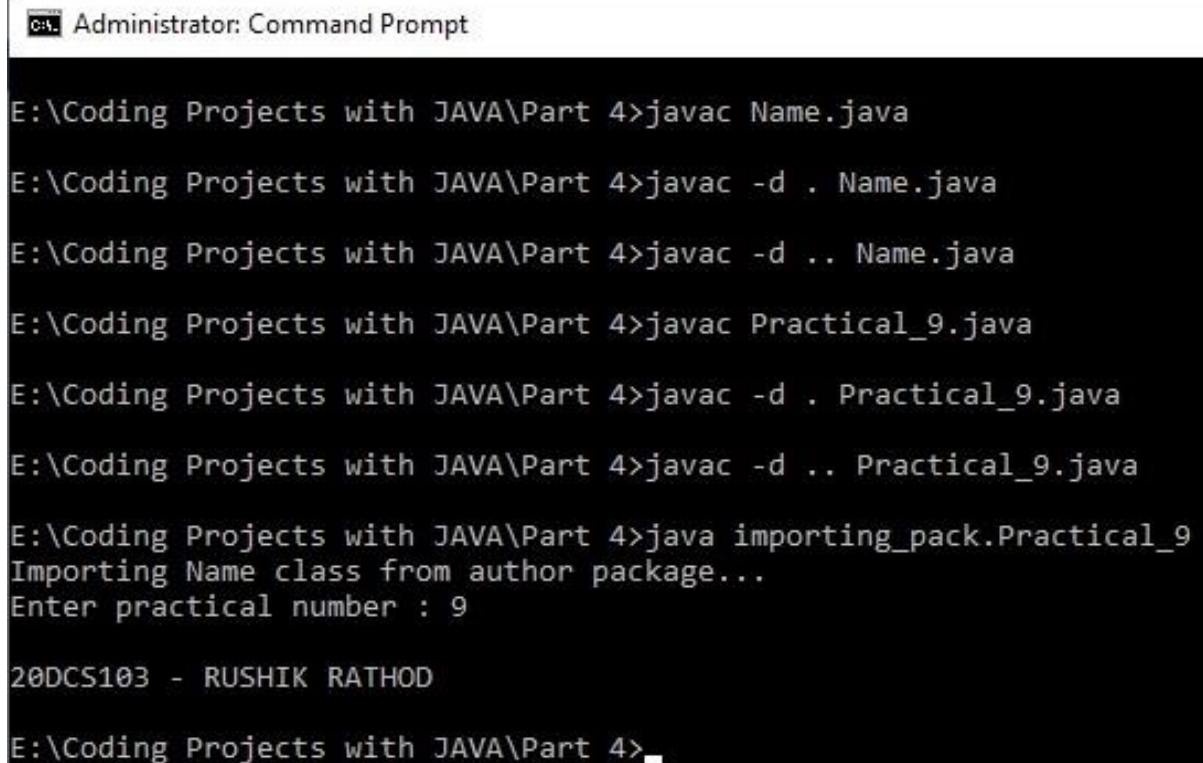
```
package importing_pack;  
import author.Name;
```

```
class Practical_9  
{  
    public static void main(String args[])  
    {  
        System.out.println("Importing Name class from author package...");  
        Name obj = new Name();  
        obj.ID();  
        System.out.println("\n20DCS103 - RUSHIK RATHOD");  
    }  
}
```

```
//user defined package
```

```
package author;  
import java.util.Scanner;
```

```
public class Name  
{  
    public void ID()  
    {  
        Scanner sc = new Scanner(System.in);  
        System.out.print("Enter practical number : ");  
        int n = sc.nextInt();  
    }  
}
```


OUTPUT:

```
C:\> Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 4>javac Name.java
E:\Coding Projects with JAVA\Part 4>javac -d . Name.java
E:\Coding Projects with JAVA\Part 4>javac -d .. Name.java
E:\Coding Projects with JAVA\Part 4>javac Practical_9.java
E:\Coding Projects with JAVA\Part 4>javac -d . Practical_9.java
E:\Coding Projects with JAVA\Part 4>javac -d .. Practical_9.java
E:\Coding Projects with JAVA\Part 4>java importing_pack.Practical_9
Importing Name class from author package...
Enter practical number : 9

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 4>_
```

CONCLUSION:

From this practical, I learnt to import classes from user define packages with the help of command prompt.

PART - 5 : EXCEPTION HANDLING

1. **Write a java program which takes two integers x & y as input, you have to compute x/y. If x and y are not integers or if y is zero, exception will occur and you have to report it.**

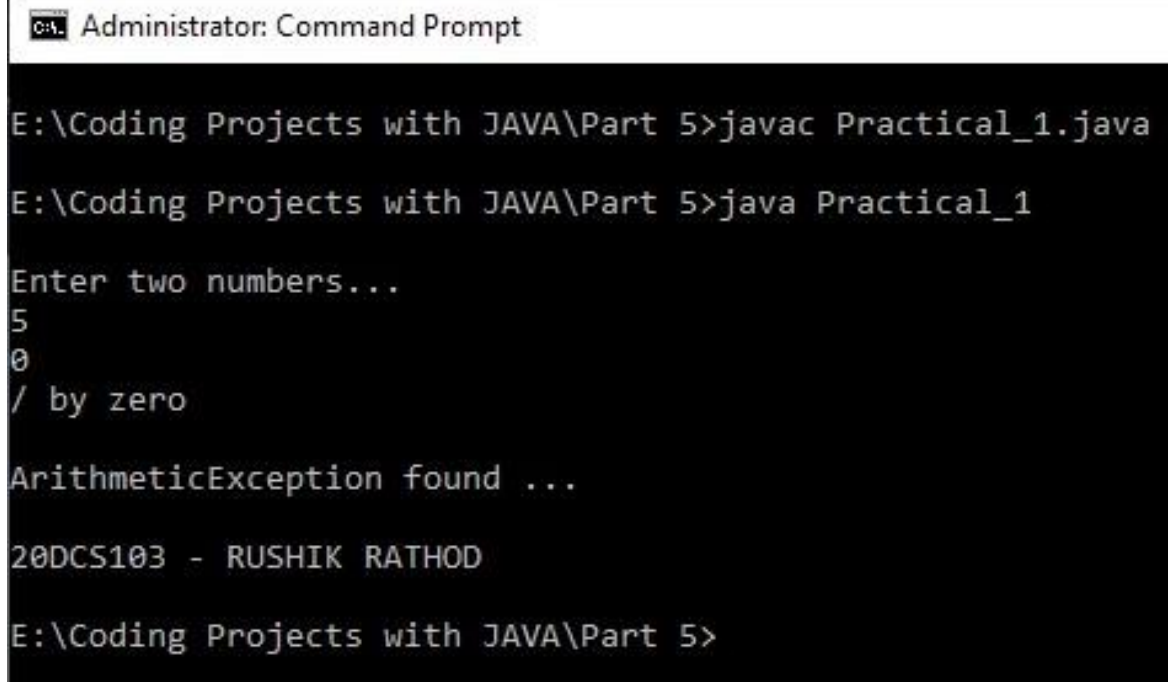
SOURCECODE:

```
import java.util.*;

public class Practical_1 extends Exception
{
    public static void main(String[] args)
    {
        int x, y, answer;
        Scanner sc = new Scanner(System.in);

        try
        {
            System.out.println("\nEnter two numbers...");
            x = sc.nextInt();
            y = sc.nextInt();
            answer = x / y;
            System.out.println("Answer : " + answer);
        }
        catch (ArithmeticException e)
        {
            System.out.println(e.getMessage());
            System.out.println("\nArithmeticException found ...");
        }
        catch (InputMismatchException e)
        {
            System.out.println(e.getMessage());
            System.out.println("\nInputMismatchException found...");
        }
    }
}
```

```
    }  
    System.out.println("\n20DCS103 - RUSHIK RATHOD");  
    }  
}
```

OUTPUT:

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 5>javac Practical_1.java
E:\Coding Projects with JAVA\Part 5>java Practical_1

Enter two numbers...
5
0
/ by zero

ArithmeticException found ...

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 5>
```

CONCLUSION:

From this practical, I learnt to find the exceptions using the concept of exception handling in JAVA. Further, I learnt the uses of try and catch.

2. **A piece of Java code is given below. You have to complete the code by writing down the handlers for exceptions thrown by the code. The exceptions the code may throw along with the handler message are listed below:**
- Division by zero:** Print "Invalid division".
- String parsed to a numeric variable:** Print "Format mismatch".
- Accessing an invalid index in string:** Print "Index is invalid".
- Accessing an invalid index in array:** Print "Array index is invalid".
- MyException:** This is a user defined Exception which you need to create. It takes a parameter param. When an exception of this class is encountered, the handler should print "MyException[param]", here param is the parameter passed to the exception class.
- Exceptions other than mentioned above:** Any other exception except the above ones fall in this category. Print "Exception encountered".
- Finally, after the exception is handled, print "Exception Handling Completed".**
- Example:** For an exception of MyException class if the parameter value is 5, the message will look like
MyException[5].
- SOURCECODE:**
- ```
import java.util.*;
import java.lang.*;

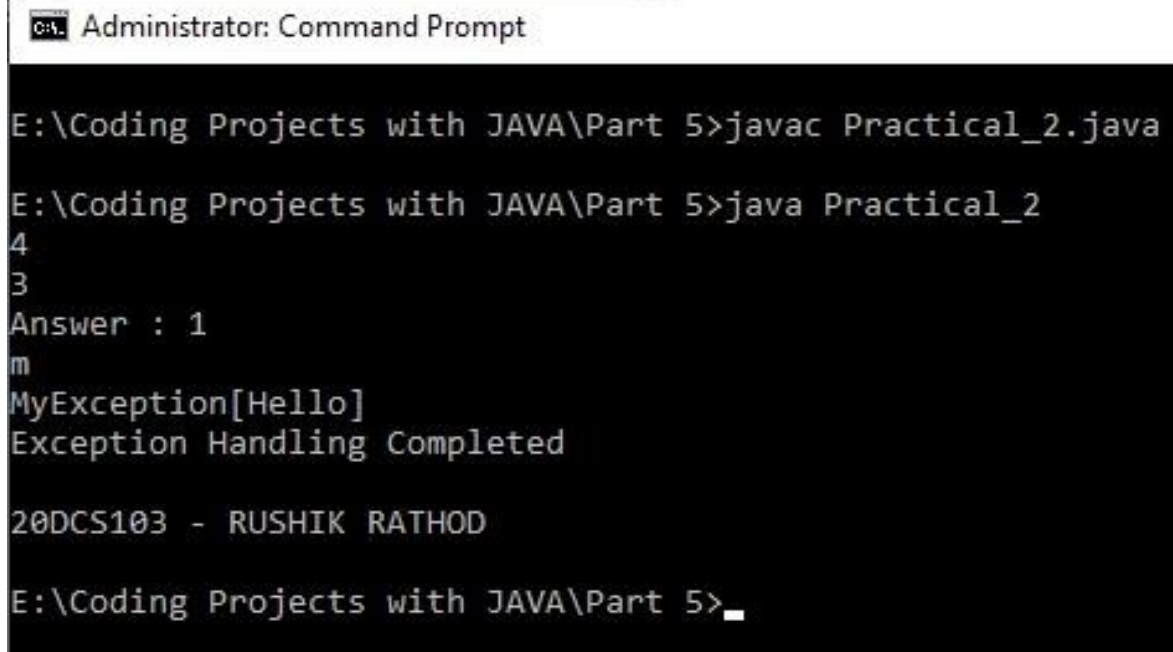
class MyException extends Exception
{
 public MyException(String s)
 {
 super(s);
 }
}

public class Practical_2 extends Exception
{
```

```
public static void main (String[] args)
{
 int a,b,answer;
 Scanner sc = new Scanner(System.in);
 try
 {
 a = sc.nextInt();
 b = sc.nextInt();
 answer = a / b;
 String s = new String();
 s = "Name";
 System.out.println("Answer : " + answer);
 System.out.println(s.charAt(2));
 throw new MyException("Hello");
 }
 catch (MyException E)
 {
 //System.out.println(E.getMessage());
 System.out.println("MyException[" + E.getMessage() + "]);
 }
 catch(ArithmeticException e)
 {
 System.out.println("Invalid division");
 }
 catch(InputMismatchException e)
 {
 System.out.println("Format mismatch");
 }
 catch(StringIndexOutOfBoundsException e)
 {
 System.out.println("Index is invalid");
 }
 catch(ArrayIndexOutOfBoundsException e)
 {

```

```
 System.out.println("Array index is invalid");
 }
 finally
 {
 System.out.println("Exception Handling Completed");
 }
 System.out.println("\n20DCS103 - RUSHIK RATHOD");
}
}
```

**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 5>javac Practical_2.java

E:\Coding Projects with JAVA\Part 5>java Practical_2
4
3
Answer : 1
m
MyException[Hello]
Exception Handling Completed

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 5>_
```

**CONCLUSION:**

From this practical, I learnt the concept of exception handling in JAVA. Further, I learnt the uses of try, catch and finally.



3. **Write a java program to generate user defined exception using “throw” and “throws” keyword.**

**Also Write a java that differentiate checked and unchecked exceptions.  
(Mention at least two checked and two unchecked exception in program).**

**SOURCECODE:**

```
import java.util.*;
import java.io.*;
```

```
class Practical_3
```

```
{
```

```
 void m1() throws Exception
```

```
 {
```

```
 m2();
```

```
 }
```

```
 void m2() throws Exception, ClassNotFoundException
```

```
 {
```

```
 try
```

```
 {
```

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

```
 System.out.println("\tChecked Exception 1");
```

```
 Class a = Class.forName("xyz"); // Checked Exception
```

```
 System.out.println("Class Found...");
```

```
 }
```

```
 catch (ClassNotFoundException e)
```

```
 {
```

```
 e.getMessage();
```

```
 System.out.println("Class does not exist ! Please verify the class name
mentioned.");
```

```
 }
```

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

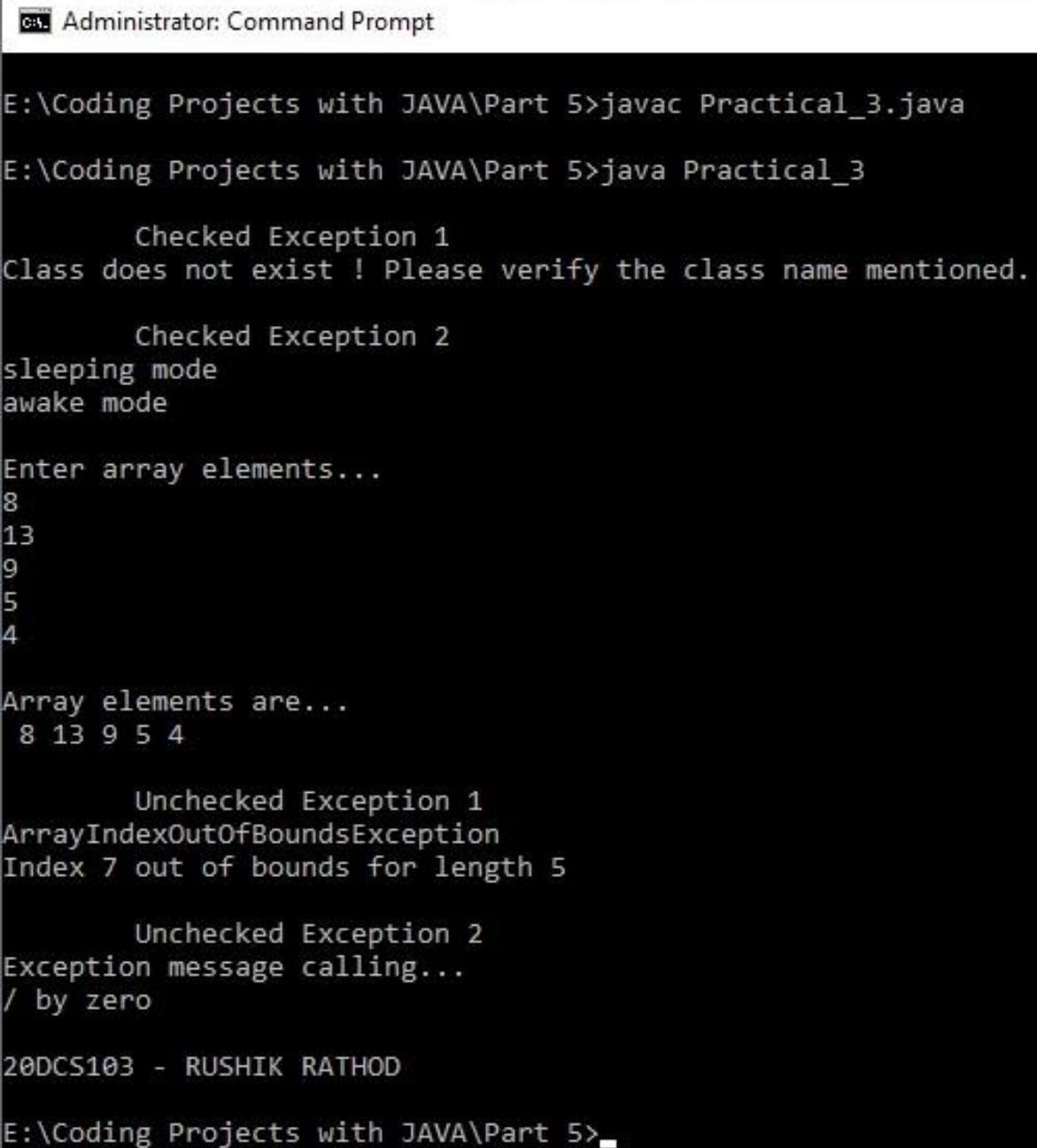
```
System.out.println("\tChecked Exception 2");
System.out.println("sleeping mode");
Thread.sleep(3000); // Checked Exception
System.out.println("awake mode");

try
{
 int arr[] = new int[5];
 int count;
 Scanner sc = new Scanner(System.in);
 System.out.println("\nEnter array elements...");
 for (count = 0; count < arr.length; count++)
 {
 arr[count] = sc.nextInt();
 }
 System.out.println("\nArray elements are...");
 for (count = 0; count < arr.length; count++)
 {
 System.out.print(" " + arr[count]);
 }
 System.out.println();
 System.out.println("\n\tUnchecked Exception 1");
 System.out.print(arr[7]); // Unchecked Exception
}

catch (ArrayIndexOutOfBoundsException e)
{
 System.out.println("ArrayIndexOutOfBoundsException");
 System.out.println(e.getMessage());
}

System.out.println();
System.out.println("\tUnchecked Exception 2");
int a = 5, b = 0;
int ans = a / b; // Unchecked Exception
```

```
 System.out.println(ans);
 }
 public static void main (String[] args)
 {
 try
 {
 new Practical_3().m1();
 }
 catch(Exception E)
 {
 System.out.println("Exception message calling...");
 System.out.println(E.getMessage());
 }
 System.out.println("\n20DCS103 - RUSHIK RATHOD");
 }
}
```

**OUTPUT:**

```
C:\> Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 5>javac Practical_3.java

E:\Coding Projects with JAVA\Part 5>java Practical_3

 Checked Exception 1
Class does not exist ! Please verify the class name mentioned.

 Checked Exception 2
sleeping mode
awake mode

Enter array elements...
8
13
9
5
4

Array elements are...
8 13 9 5 4

 Unchecked Exception 1
ArrayIndexOutOfBoundsException
Index 7 out of bounds for length 5

 Unchecked Exception 2
Exception message calling...
/ by zero

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 5>_
```

**CONCLUSION:**

From this practical, I learnt to find the user defined exceptions in JAVA. Also, I learnt the uses of throw and throws.

**PART - 6 : FILE HANDLING & STREAMS**

1. **Write a program that will count the number of lines in each file that is specified on the command line. Assume that the files are text files. Note that multiple files can be specified, as in "java LineCounts file1.txt file2.txt file3.txt".**
- Write each file name, along with the number of lines in that file, to standard output. If an error occurs while trying to read from one of the files, you should print an error message for that file, but you should still process all the remaining files.**

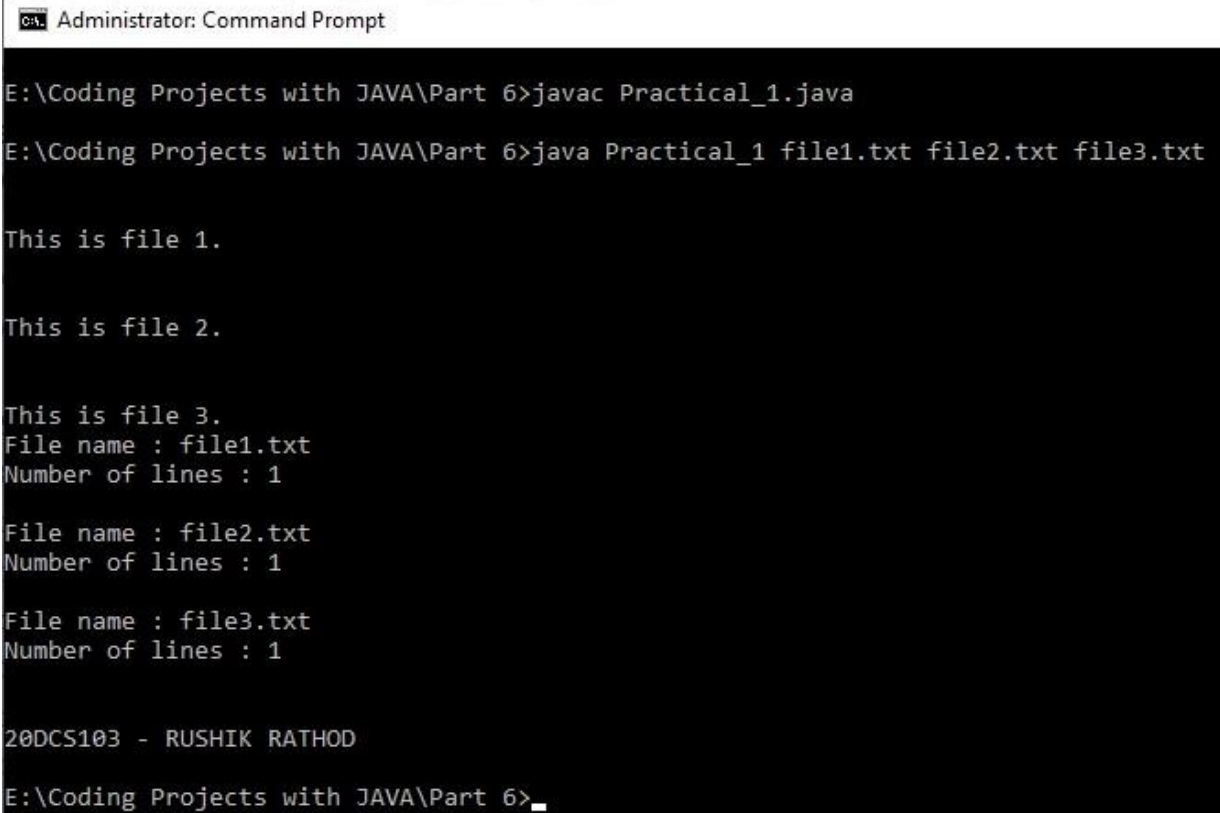
**SOURCECODE:**

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class Practical_1
{
 public static void main(String[] args)
 {
 try
 {
 int[] line = new int[4];
 for (int i = 0; i < args.length; i++)
 {
 File obj = new File(args[i]);
 Scanner sc = new Scanner(obj);
 System.out.println();
 int line_in_file = 0;
 while (sc.hasNextLine())
 {
 String info = sc.nextLine();
 System.out.println();
 }
 }
 }
 catch (FileNotFoundException e)
 {
 System.out.println(e.getMessage());
 }
 }
}
```

```
 System.out.println(info);
 line_in_file++;
 line[i] = line_in_file;
 }
 sc.close();
}

for (int i = 0; i < args.length; i++)
{
 File obj = new File(args[i]);
 Scanner sc = new Scanner(obj);
 System.out.println("File name : " + obj.getName());
 System.out.println("Number of lines : " + line[i]);
 System.out.println();
}
System.out.println("\n20DCS103 - RUSHIK RATHOD");
}
catch (FileNotFoundException e)
{
 System.out.print("Exception found ");
 e.printStackTrace();
}
}
```

**OUTPUT:**

```
C:\> Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 6>javac Practical_1.java
E:\Coding Projects with JAVA\Part 6>java Practical_1 file1.txt file2.txt file3.txt

This is file 1.

This is file 2.

This is file 3.
File name : file1.txt
Number of lines : 1

File name : file2.txt
Number of lines : 1

File name : file3.txt
Number of lines : 1

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 6>_
```

**CONCLUSION:**

From this practical, I learnt to create an object and opening and closing of the files. In addition, I learnt to handle different files using the command prompt.

2. **Write an example that counts the number of times a particular character, such as e, appears in a file. The character can be specified at the command line. You can use xanadu.txt as the input file.**

**SOURCECODE:**

```
import java.io.*;
import java.util.*;

public class Practical_2
{
 public static void main(String[] args) throws Exception
 {
 File file = new File("20DCS103.txt");
 FileReader file_r = new FileReader(file);
 BufferedReader obj = new BufferedReader(file_r);

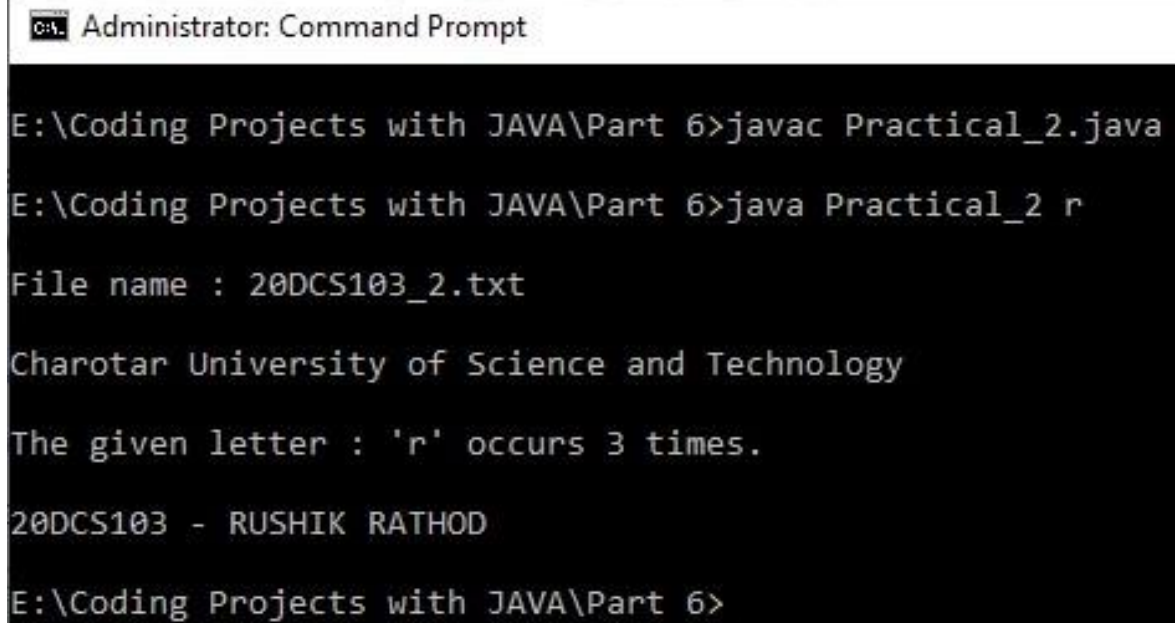
 Scanner sc = new Scanner(file);

 while (sc.hasNextLine())
 {
 String info = sc.nextLine();
 System.out.println("\nFile name : " + file.getName());
 System.out.println();
 System.out.println(info);
 }

 int i, no_of_times = 0;
 while ((i = obj.read()) != -1)
 {
 if (i == args[0].charAt(0))
 {
 no_of_times++;
 }
 }
 }
}
```



```
 System.out.println();
 System.out.println("The given letter : '" + args[0].charAt(0) + "' occurs " +
no_of_times + " times.");
 System.out.println("\n20DCS103 - RUSHIK RATHOD");
 file_r.close();
 }
}
```

**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 6>javac Practical_2.java
E:\Coding Projects with JAVA\Part 6>java Practical_2 r

File name : 20DCS103_2.txt

Charotar University of Science and Technology

The given letter : 'r' occurs 3 times.

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 6>
```

**CONCLUSION:**

From this practical, I learnt to find the number of the number of characters in the line using the different file handling functions.

3. **Write a Java Program to Search for a given word in a File. Also show use of Wrapper Class with an example.**

**SOURCECODE:**

```
import java.io.*;
import java.util.*;

public class Practical_3 {
 public static void main(String[] args) throws Exception
 {
 File file = new File("20DCS103_3.txt");
 FileReader file_r = new FileReader(file);
 BufferedReader obj = new BufferedReader(file_r);
 Scanner sc = new Scanner(file);

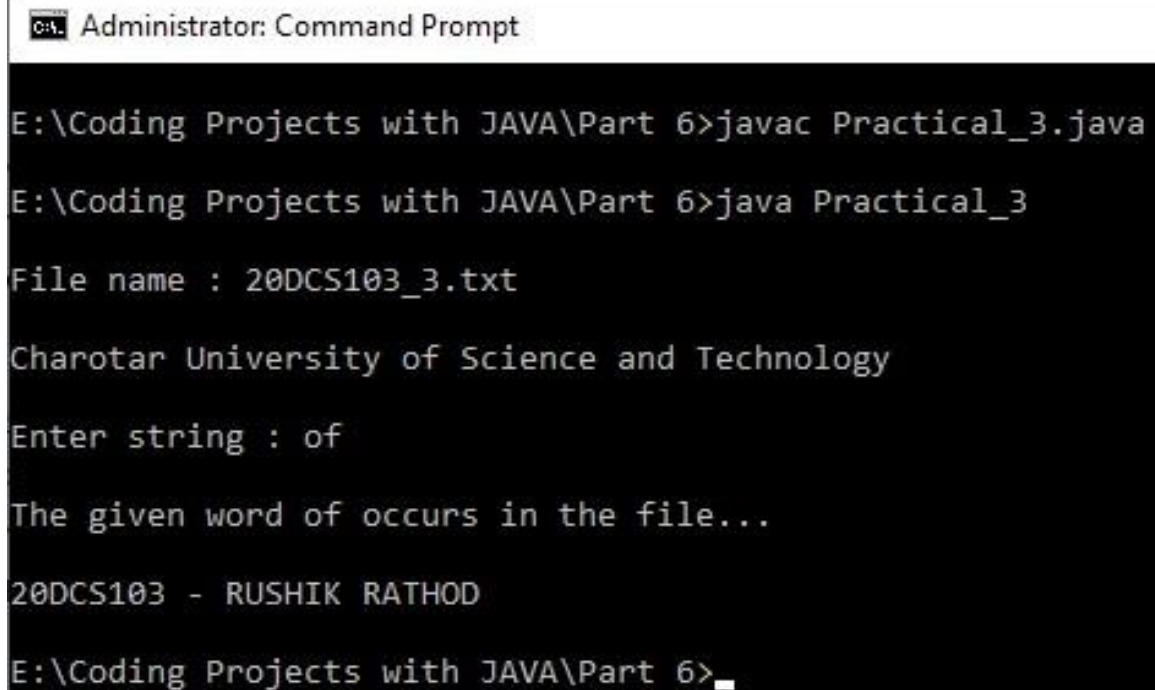
 while (sc.hasNextLine())
 {
 String info = sc.nextLine();
 System.out.println("\nFile name : " + file.getName());
 System.out.println();
 System.out.println(info);
 }

 String words[] = null;
 System.out.print("\nEnter string : ");
 Scanner gett = new Scanner(System.in);
 String a = gett.nextLine();
 String i;
 boolean occur = false;

 while ((i = obj.readLine()) != null)
 {
 words = i.split(" ");
 for (String word : words)
```

```
 {
 if (word.equals(a))
 {
 occur = true;
 }
 }

 if (occur)
 {
 System.out.println("\nThe given word " + a + " occurs in the file...");
 }
 else
 {
 System.out.println("\nThe given word " + a + " does not occur in the file
!!!");
 }
 System.out.println("\n20DCS103 - RUSHIK RATHOD");
 file_r.close();
}
}
```

**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 6>javac Practical_3.java
E:\Coding Projects with JAVA\Part 6>java Practical_3
File name : 20DCS103_3.txt
Charotar University of Science and Technology
Enter string : of
The given word of occurs in the file...
20DCS103 - RUSHIK RATHOD
E:\Coding Projects with JAVA\Part 6>_
```

**CONCLUSION:**

From this practical, I learnt to check whether the file contains the entered string or not using the file handling concept.

4. **Write a program to copy data from one file to another file. If the destination file does not exist, it is created automatically.**

**SOURCECODE:**

```
import java.io.*;

public class Practical_4 {
 public static void main(String[] args) throws IOException {
 try {
 // Creating FileReader object
 File file_1 = new File("P4_file1.txt");

 FileReader read_file = new FileReader(file_1);
 char data[] = new char[(int) file_1.length()];

 // Reading data from the file
 read_file.read(data);

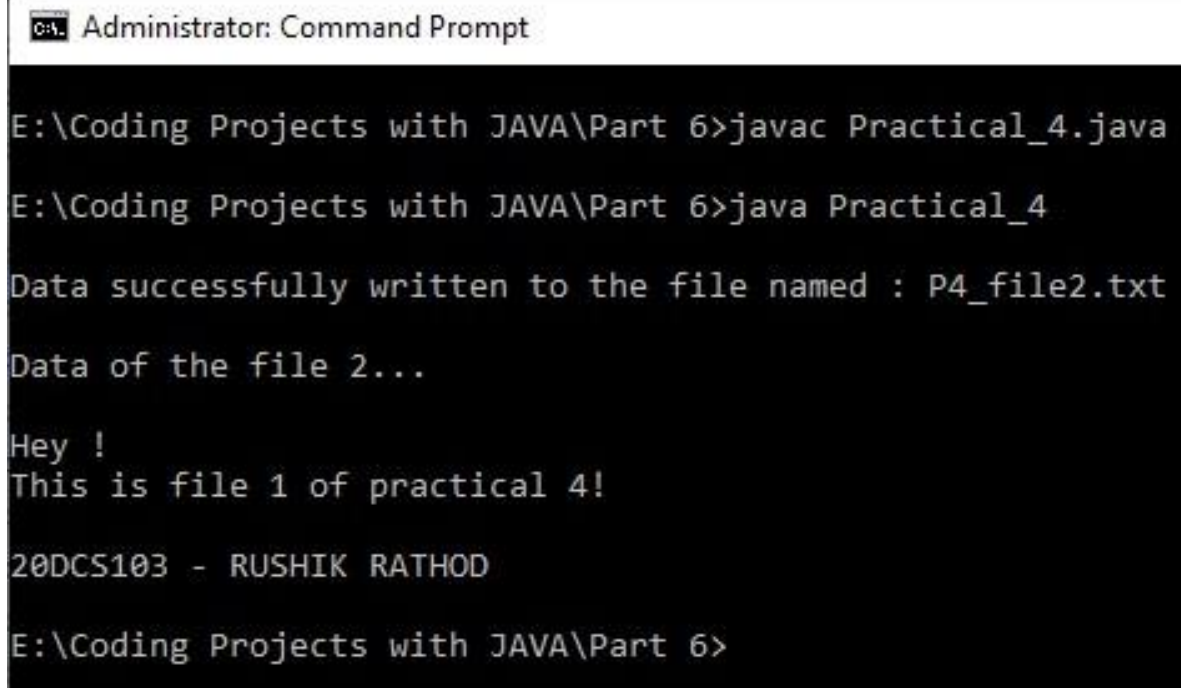
 // Writing data to another file

 File file2 = new File("P4_file2.txt");
 FileWriter write_file = new FileWriter(file2);

 // Writing data to the file
 write_file.write(data);
 write_file.flush();
 write_file.close();
 read_file.close();
 System.out.println("\nData successfully written to the file named : " + file
2.getName());
 System.out.println("\nData of the file 2...");
 System.out.println();
 System.out.println(data);
 }
 }
}
```

```
 }

 catch (IOException e) {
 System.out.print("Exception" + e.getMessage());
 }
 System.out.println("\n20DCS103 - RUSHIK RATHOD");
}
}
```

**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 6>javac Practical_4.java
E:\Coding Projects with JAVA\Part 6>java Practical_4

Data successfully written to the file named : P4_file2.txt

Data of the file 2...

Hey !
This is file 1 of practical 4!

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 6>
```

**CONCLUSION:**

From this practical, I learnt to copy the data of one file into another file using the concept of file handling in JAVA language. Further, I also learnt to create a file automatically when the file is not already made.



5. **Write a program to show use of character and byte stream. Also show use of `BufferedReader` / `BufferedWriter` to read console input and write them into a file.**

**SOURCECODE:**

```
import java.io.*;
import java.util.*;

public class Practical_5 {
 public static void main(String[] args) throws Exception {

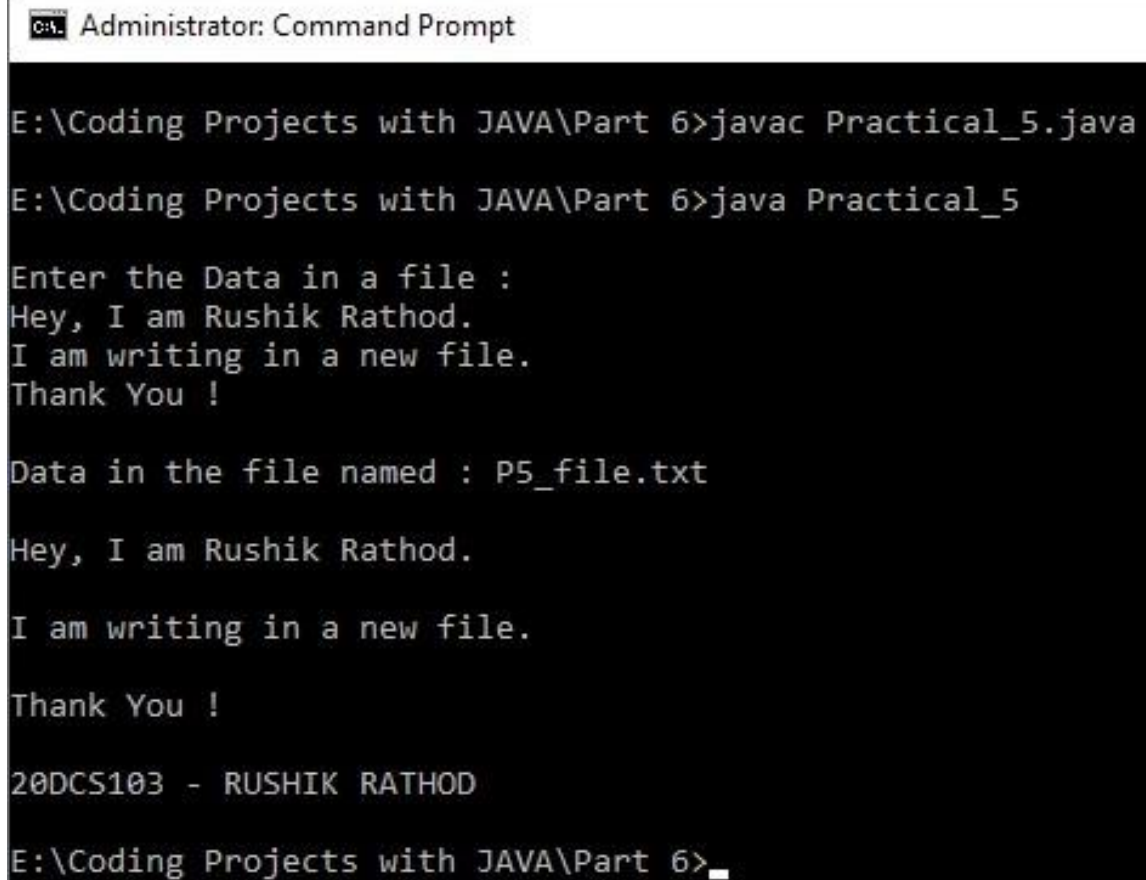
 File file = new File("P5_file.txt");
 FileWriter f_w = new FileWriter(file, true);
 BufferedWriter obj_w = new BufferedWriter(f_w);
 Scanner sc = new Scanner(System.in);

 System.out.println("\nEnter the Data in a file : ");
 int i = 3;
 while (i > 0) {
 String data = sc.nextLine() + "\n";
 obj_w.write(data);
 i--;
 }
 obj_w.close();
 sc.close();

 FileReader file_r = new FileReader(file);
 BufferedReader obj_r = new BufferedReader(file_r);
 Scanner sc2 = new Scanner(file);

 System.out.println("\nData in the file named : " + file.getName());
 while (sc2.hasNextLine()) {
 String info = sc2.nextLine();
 System.out.println();
 }
 }
}
```

```
 System.out.println(info);
 }
 System.out.println("\n20DCS103 - RUSHIK RATHOD");
}
}
```

**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 6>javac Practical_5.java

E:\Coding Projects with JAVA\Part 6>java Practical_5

Enter the Data in a file :
Hey, I am Rushik Rathod.
I am writing in a new file.
Thank You !

Data in the file named : P5_file.txt

Hey, I am Rushik Rathod.

I am writing in a new file.

Thank You !

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 6>_
```

**CONCLUSION:**

I learnt the usage of BufferedReader and BufferedWriter in the file handling concept in JAVA.

**PART - 7 : MULTITHREADING**

1. **Write a program to create thread which display “Hello World” message.**  
**A. by extending Thread class**  
**B. by using Runnable interface.**

**SOURCECODE:**

```
class Multithreading_A extends Thread {
 public void run() {

 try {
 // Displaying the thread which is running
 System.out.println("\nDisplaying by extending thread...");
 System.out.println("Hello JAVA !");
 }

 catch (Exception e) {
 // Throwing an exception
 System.out.println("Exception is caught..." + e.getMessage());
 }
 }
}

class Multithreading_B implements Runnable {
 public void run() {

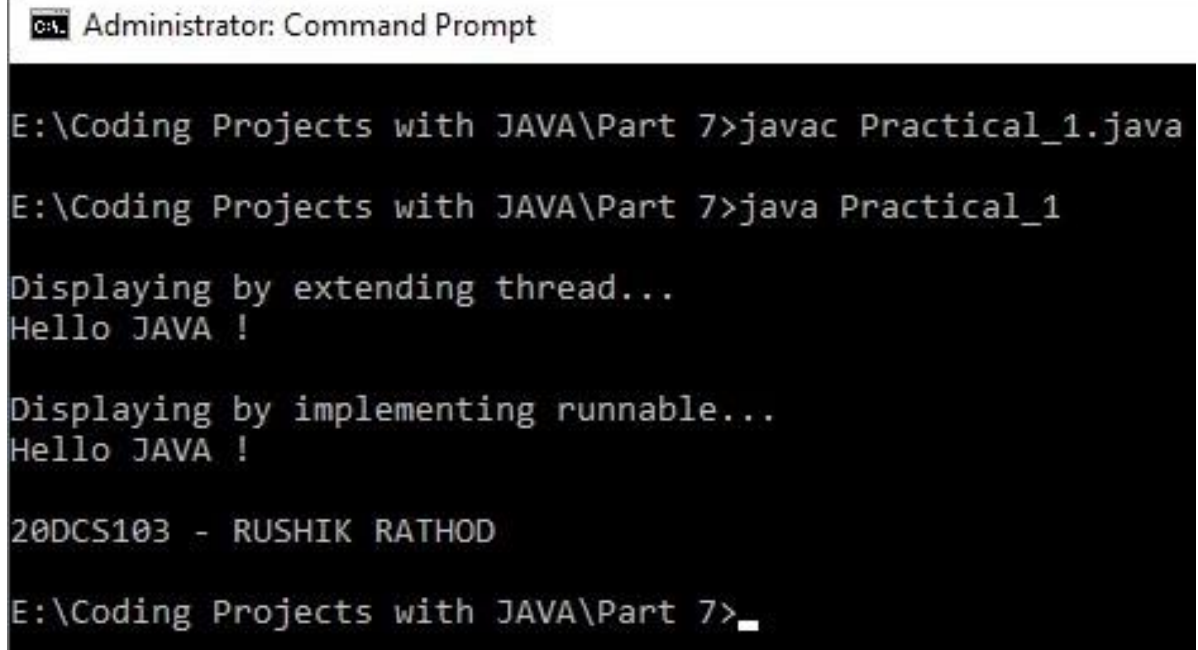
 try {
 // Displaying the thread which is running
 System.out.println("\nDisplaying by implementing runnable...");
 System.out.println("Hello JAVA !");
 }

 catch (Exception e) {
```

```
 // Throwing an exception
 System.out.println("Exception is caught.. " + e.getMessage());
 }
 System.out.println("\n20DCS103 - RUSHIK RATHOD");
}

public class Practical_1 {
 public static void main(String[] args) {

 Multithreading_A obj1 = new Multithreading_A();
 obj1.start();
 Thread obj2 = new Thread(new Multithreading_B());
 obj2.start();
 }
}
```

**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 7>javac Practical_1.java

E:\Coding Projects with JAVA\Part 7>java Practical_1

Displaying by extending thread...
Hello JAVA !

Displaying by implementing runnable...
Hello JAVA !

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 7>_
```

**CONCLUSION:**

From this practical, I learnt to print message on a console by extending the thread class and by using the runnable interface as well.

2. **Write a program which takes N and number of threads as an argument. Program should distribute the task of summation of N numbers amongst number of threads and final result to be displayed on the console.**

**SOURCECODE:**

```
import java.util.Scanner;

class sum extends Thread {

 int x, y;
 static int result = 0;

 public sum(int a, int b) {
 x = a;
 y = b;
 }

 public void run() {
 add(x, y); // addition
 result(); // final sum
 }

 public void add(int a, int b) {
 int s = 0;
 for (int i = a; i <= b; i++) {
 s += i;
 }
 System.out.println("From " + Thread.currentThread().getName() + "\tSum of " + a + " to " + b + " numbers = " + s);
 result = result + s;
 }

 public void result() {
 System.out.println("Total= " + result);
 }
}
```

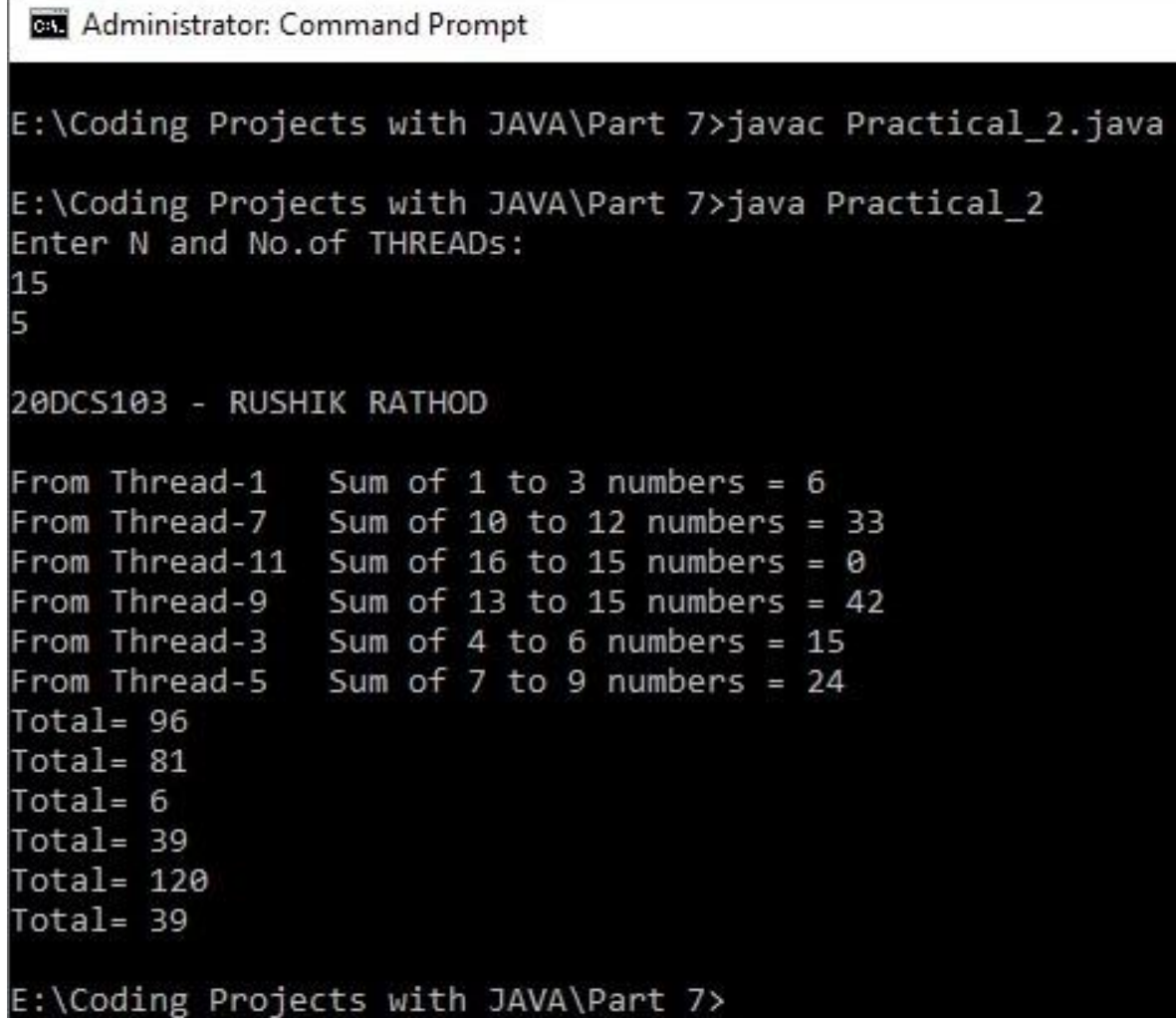
```
 }
}

public class Practical_2 {
 public static void main (String[] args)
 {
 System.out.println("Enter N and No.of THREADs: ");
 Scanner sc = new Scanner(System.in);
 int N = sc.nextInt();
 int n = sc.nextInt();
 int q= N/n;
 int r= N%n;
 int i=1; // Threads

 for (i=1; (i+q-1)<=N ;i+=q)
 {
 //new_thread(iq,(i+1)q);
 Thread t1 = new Thread(new sum(i,i+q-1));
 t1.start();
 }

 if (q<r)
 {
 r=r-q;
 Thread t2 = new Thread(new sum(i,i+r-1));
 //Calling Run method through start();
 t2.start();
 }
 System.out.println("\n20DCS103 - RUSHIK RATHOD");
 }
}
```



**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 7>javac Practical_2.java

E:\Coding Projects with JAVA\Part 7>java Practical_2
Enter N and No.of THREADs:
15
5

20DCS103 - RUSHIK RATHOD

From Thread-1 Sum of 1 to 3 numbers = 6
From Thread-7 Sum of 10 to 12 numbers = 33
From Thread-11 Sum of 16 to 15 numbers = 0
From Thread-9 Sum of 13 to 15 numbers = 42
From Thread-3 Sum of 4 to 6 numbers = 15
From Thread-5 Sum of 7 to 9 numbers = 24
Total= 96
Total= 81
Total= 6
Total= 39
Total= 120
Total= 39

E:\Coding Projects with JAVA\Part 7>
```

**CONCLUSION:**

I learnt to distribute a task among different threads in the multithreading concept.

3. **Write a java program that implements a multi-thread application that has three threads. First thread generates random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.**

**SOURCECODE:**

```
import java.util.Random;

public class Practical_3 extends Thread {

 public void run() {

 Random r = new Random();
 for (int count = 0; count < 10; count++) {

 int random_int = r.nextInt(100);
 System.out.println("\nRandom integer generated : " + random_int);
 if ((random_int % 2) == 0) {
 SqThread t2 = new SqThread(random_int);
 t2.start();
 } else {
 CubeThread t3 = new CubeThread(random_int);
 t3.start();
 }

 try {
 Thread.sleep(1000);
 } catch (InterruptedException e) {
 System.out.println("Exception occurred : " + e.getMessage());
 }
 }
 // Performed By :
 System.out.println("\n20DCS103 - RUSHIK RATHOD\n");
 }

 public static void main(String[] args) {
 Practical_3 t1 = new Practical_3();
 }
}
```

```
 t1.start();
 }
}

class SqThread extends Thread {
 long num;

 SqThread(int r_num) {
 num = r_num;
 }

 public void run() {
 System.out.println("EVEN NUMBER");
 System.out.println("Square of " + num + " is : " + num * num);
 }
}

class CubeThread extends Thread {

 long Num;

 CubeThread(int r_num) {
 Num = r_num;
 }

 public void run() {
 System.out.println("ODD NUMBER");
 System.out.println("Cube of " + Num + " is : " + Math.pow(Num, 3));
 }
}
```

**OUTPUT:**

```
Administrator: Command Prompt
E:\Coding Projects with JAVA\Part 7>java Practical_3

Random integer generated : 72
EVEN NUMBER
Square of 72 is : 5184

Random integer generated : 44
EVEN NUMBER
Square of 44 is : 1936

Random integer generated : 9
ODD NUMBER
Cube of 9 is : 729.0

Random integer generated : 96
EVEN NUMBER
Square of 96 is : 9216

Random integer generated : 84
EVEN NUMBER
Square of 84 is : 7056

Random integer generated : 16
EVEN NUMBER
Square of 16 is : 256

Random integer generated : 35
ODD NUMBER
Cube of 35 is : 42875.0

Random integer generated : 25
ODD NUMBER
Cube of 25 is : 15625.0

Random integer generated : 19
ODD NUMBER
Cube of 19 is : 6859.0

Random integer generated : 20
EVEN NUMBER
Square of 20 is : 400

20DCS103 - RUSHIK RATHOD
```

**CONCLUSION:**

From this practical, I learnt a different aspect of multithreading by performing the task given in the question.

4. **Write a program to increment the value of one variable by one and display it after one second using thread using sleep() method.**

**SOURCECODE:**

```
public class Practical_4 extends Thread {

 int value = 21;

 public void run() {

 // Display Before Increment
 System.out.println("\n*****Displaying before increment*****\n");
 System.out.println("Value in thread " + Thread.currentThread().getName()
+ " is: " + value);

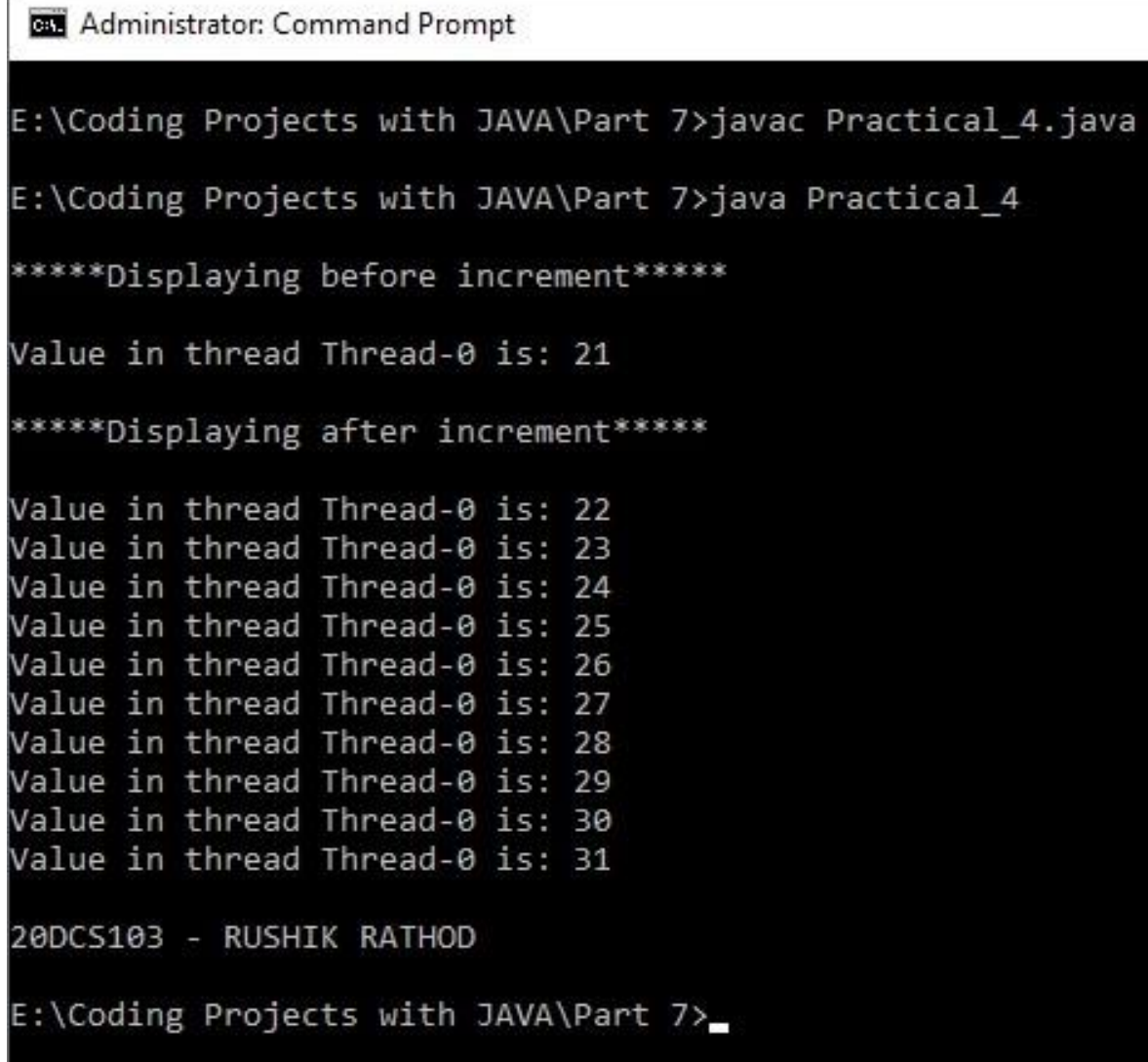
 // Increamenting value by one and then Displaying
 System.out.println("\n*****Displaying after increment*****\n");
 for (int i = 0; i < 10; i++) {
 value++;
 System.out.println("Value in thread " + Thread.currentThread().getName(
) + " is: " + value);
 try {
 Thread.sleep(1000);
 }
 catch (Exception e) {
 System.out.println(e.getMessage());
 }
 }
 System.out.println("\n20DCS103 - RUSHIK RATHOD");
 }

 public static void main(String args[]) {

 // creating one thread
```

```
Practical_4 thread = new Practical_4();

// Calling Start();
thread.start();
}
}
```

**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 7>javac Practical_4.java
E:\Coding Projects with JAVA\Part 7>java Practical_4

*****Displaying before increment*****

Value in thread Thread-0 is: 21

*****Displaying after increment*****

Value in thread Thread-0 is: 22
Value in thread Thread-0 is: 23
Value in thread Thread-0 is: 24
Value in thread Thread-0 is: 25
Value in thread Thread-0 is: 26
Value in thread Thread-0 is: 27
Value in thread Thread-0 is: 28
Value in thread Thread-0 is: 29
Value in thread Thread-0 is: 30
Value in thread Thread-0 is: 31

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 7>_
```

**CONCLUSION:**

From this practical, I learnt to increase the value of a variable using sleep() method in the multithreading concept.

5. **Write a program to create three threads 'FIRST', 'SECOND', 'THIRD'. Set the priority of the 'FIRST' thread to 3, the 'SECOND' thread to 5(default) and the 'THIRD' thread to 7.**

**SOURCECODE:**

```
public class Practical_5 extends Thread {

 public void run() {
 // print the user defined priority
 System.out.println("\nPriority of thread " + Thread.currentThread().getName() + " is: " + Thread.currentThread().getPriority());
 }

 public static void main(String args[]) {

 // creating one thread
 Practical_5 FIRST = new Practical_5();
 Practical_5 SECOND = new Practical_5();
 Practical_5 THIRD = new Practical_5();

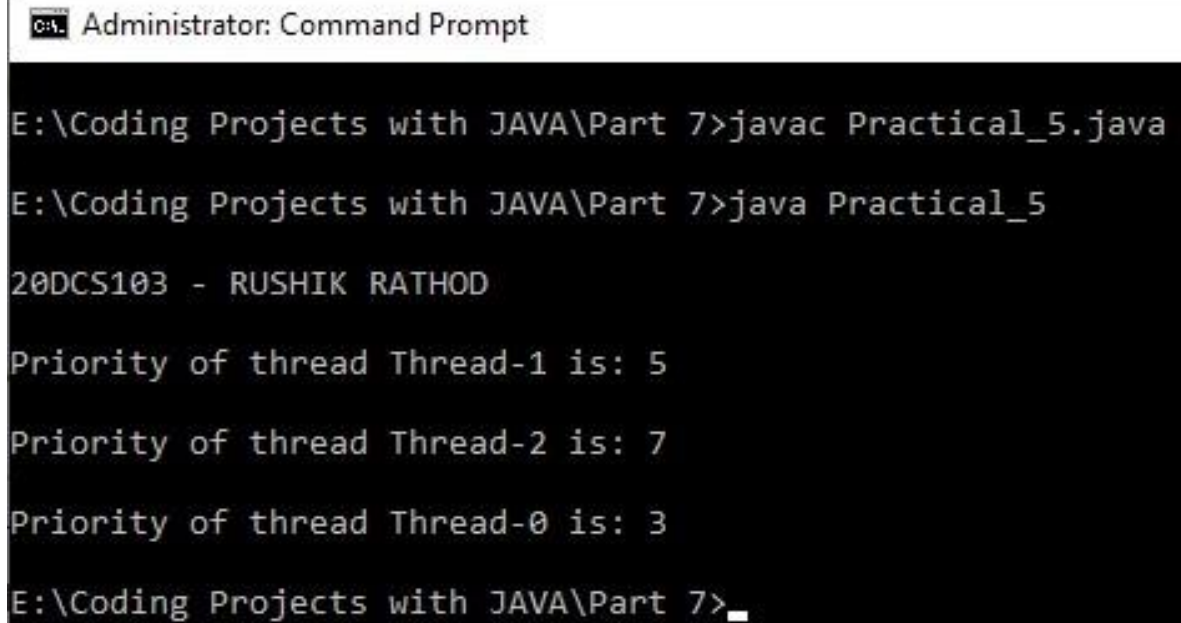
 // set the priority
 FIRST.setPriority(3);

 // set default priority to SECOND
 THIRD.setPriority(7);

 // this will call the run() method
 FIRST.start();
 SECOND.start();
 THIRD.start();

 // Performed By
 System.out.println("\n20DCS103 - RUSHIK RATHOD");
 }
}
```



**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 7>javac Practical_5.java
E:\Coding Projects with JAVA\Part 7>java Practical_5

20DCS103 - RUSHIK RATHOD

Priority of thread Thread-1 is: 5
Priority of thread Thread-2 is: 7
Priority of thread Thread-0 is: 3
E:\Coding Projects with JAVA\Part 7>_
```

**CONCLUSION:**

From this practical, I learnt to set the priority of threads.

6. **Write a program to solve producer-consumer problem using thread synchronization.**

**SOURCECODE:**

```
import java.io.*;
import java.util.*;

class Grocery
{
 int item_no;
 int n, a;

 public synchronized void Produce(int N)
 {
 n = N;
 System.out.println("\nStock Order of " + n + " Items");
 for (item_no = 1; item_no <= n; item_no++)
 {
 System.out.println("Item " + item_no + " Produced.");
 }
 }

 public synchronized void Consume(int A)
 {
 a = A;
 System.out.println("Consumer Order of " + a + " Items");
 if (a < n)
 {
 for (item_no = 1; item_no <= a; item_no++)
 {
 System.out.println("Item " + item_no + " Consumed.");
 }
 }
 else if (a > n)
```

```
{
 for (item_no = 1; item_no <= n; item_no++)
 {
 System.out.println("Item " + item_no + " Consumed.");
 }
 System.out.println("Oops! Shortage of " + (a - n) + " Items...");
}
else
 System.out.println("Items Out of Stock !!");
}
}

class Main extends Thread {
 int P, C;

 Main(int N, int A)
 {
 P = N;
 C = A;
 }

 Grocery G = new Grocery();

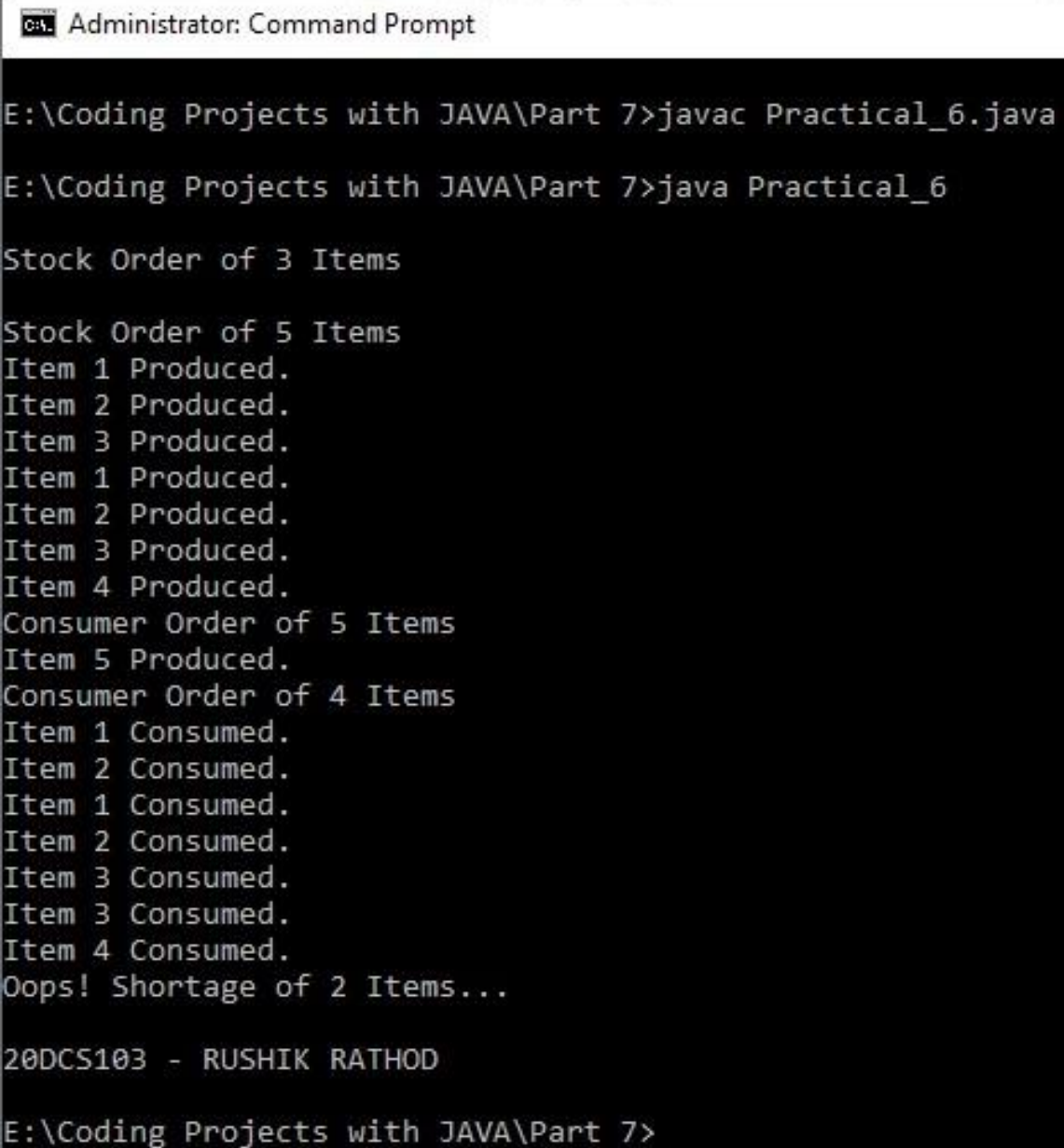
 public void run()
 {
 G.Produce(P);
 G.Consume(C);
 }
}

class Practical_6
{
 public static void main(String[] args)
 {
```

```
Main G1 = new Main(5, 4);
Main G2 = new Main(3, 5);

G1.start();
G2.start();

try
{
 G1.join();
 G2.join();
 System.out.println("\n20DCS103 - RUSHIK RATHOD");
}
catch (Exception e)
{
 System.out.println("Interrupted Exception");
}
}
```

**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 7>javac Practical_6.java
E:\Coding Projects with JAVA\Part 7>java Practical_6

Stock Order of 3 Items

Stock Order of 5 Items
Item 1 Produced.
Item 2 Produced.
Item 3 Produced.
Item 1 Produced.
Item 2 Produced.
Item 3 Produced.
Item 4 Produced.
Consumer Order of 5 Items
Item 5 Produced.
Consumer Order of 4 Items
Item 1 Consumed.
Item 2 Consumed.
Item 1 Consumed.
Item 2 Consumed.
Item 3 Consumed.
Item 3 Consumed.
Item 4 Consumed.
Oops! Shortage of 2 Items...

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 7>
```

**CONCLUSION:**

From this practical, I learnt to use 'synchro', an inbuilt keyword of java. Further, I learnt the effective use of 'notify()' and 'sleep()' methods.

## PART - 8 : Collection Framework & Generic

1. **Design a Custom Stack using ArrayList class, which implements following functionalities of stack.**

| MyStack                  |                                                            |
|--------------------------|------------------------------------------------------------|
| -list: ArrayList<Object> | A list to store elements.                                  |
| +isEmpty(): boolean      | Returns true if this stack is empty.                       |
| +getSize(): int          | Returns the number of elements in this stack.              |
| +peek(): Object          | Returns the top element in this stack without removing it. |
| +pop(): Object           | Returns and removes the top element in this stack.         |
| +push(o: Object): void   | Adds a new element to the top of this stack.               |

### SOURCECODE:

```
import java.util.*;
```

```
class ArrayList extends Stack
```

```
{
 ArrayList()
 {
 Stack<Object> S = new Stack<Object>();
 }
}
```

```
public class Practical_1
```

```
{
 public static void main(String[] args)
 {
 ArrayList l = new ArrayList();
 System.out.println();
 System.out.println("List is empty : " + l.isEmpty());
 l.add(1);
 }
}
```

```
l.add(1, 2);
l.add(2, 3);
l.add(3, 5);

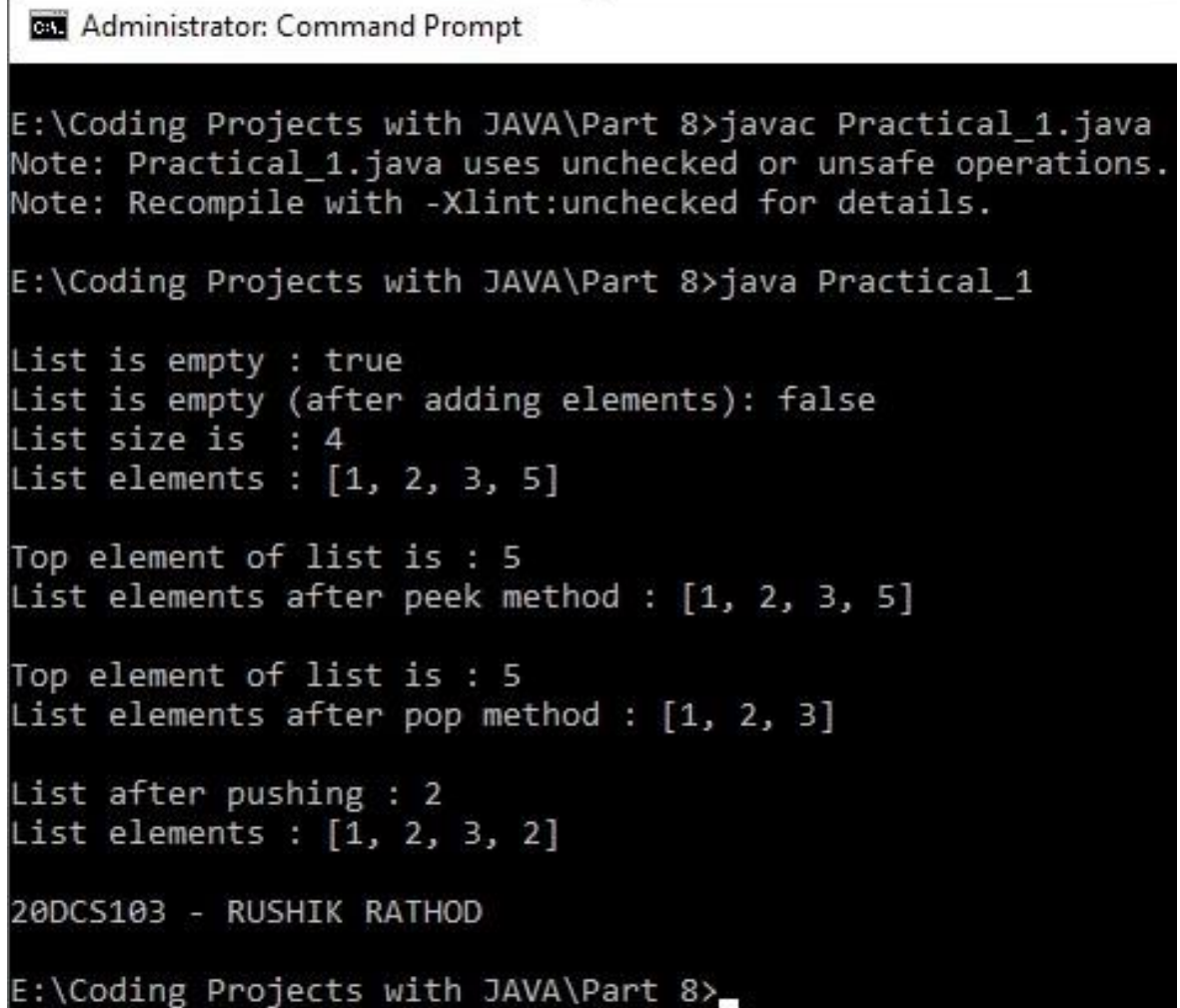
System.out.println("List is empty (after adding elements): " + l.isEmpty());
System.out.println("List size is : " + l.size());
System.out.println("List elements : " + l);

System.out.println();
System.out.println("Top element of list is : " + l.peek());
System.out.println("List elements after peek method : " + l);

System.out.println();
System.out.println("Top element of list is : " + l.pop());
System.out.println("List elements after pop method : " + l);

System.out.println();
System.out.println("List after pushing : " + l.push(2));
System.out.println("List elements : " + l);

System.out.println("\n20DCS103 - RUSHIK RATHOD");
}
```

**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 8>javac Practical_1.java
Note: Practical_1.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.

E:\Coding Projects with JAVA\Part 8>java Practical_1

List is empty : true
List is empty (after adding elements): false
List size is : 4
List elements : [1, 2, 3, 5]

Top element of list is : 5
List elements after peek method : [1, 2, 3, 5]

Top element of list is : 5
List elements after pop method : [1, 2, 3]

List after pushing : 2
List elements : [1, 2, 3, 2]

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 8>_
```

**CONCLUSION:**

I learnt to design a custom stack which consists some functions such as isEmpty(), getSize(), push(), peek(), pop(), etc.



**2. Create a generic method for sorting an array of Comparable objects.****SOURCECODE:**

```
import java.util.Arrays;
```

```
class Student implements Comparable<Student>
```

```
{
```

```
 String name;
```

```
 int java_marks;
```

```
 Student(String name, int java_marks)
```

```
 {
```

```
 this.name = name;
```

```
 this.java_marks = java_marks;
```

```
 }
```

```
 public int compareTo(Student s)
```

```
 {
```

```
 return s.java_marks - this.java_marks;
```

```
 }
```

```
 public String toString()
```

```
 {
```

```
 return String.format("[%s, %d]", name, java_marks);
```

```
 }
```

```
}
```

```
public class Practical_2
```

```
{
```

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

```
 {
```

```
 Student[] s = new Student[5];
```

```
 s[0] = new Student("Shahil", 95);
```

```
 s[1] = new Student("Crony", 99);
```

```
s[2] = new Student("Poojan", 97);
s[3] = new Student("Ajay", 94);
s[4] = new Student("Prakash", 96);

System.out.println();
System.out.println("\t\t\t\t\t***** DATA *****");
System.out.println();
System.out.println("Before sorting in descending order : " +
Arrays.toString(s));
System.out.println();

Arrays.sort(s);
Arrays.toString(s);
System.out.println("After sorting in descending order : " +
Arrays.toString(s));

System.out.println("\n20DCS103 - RUSHIK RATHOD");
}
}
```

**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 8>javac Practical_2.java
E:\Coding Projects with JAVA\Part 8>java Practical_2

***** DATA *****

Before sorting in descending order : [[Shahil, 95], [Crony, 99], [Poojan, 97], [Ajay, 94], [Prakash, 96]]
After sorting in decending order : [[Crony, 99], [Poojan, 97], [Prakash, 96], [Shahil, 95], [Ajay, 94]]
20DCS103 - RUSHIK RATHOD
E:\Coding Projects with JAVA\Part 8>_
```

**CONCLUSION:**

From this practical, I learnt to create a generic method for sorting an array of comparable objects.

3. **Write a program that counts the occurrences of words in a text and displays the words and their occurrences in alphabetical order of the words. Using Map and Set Classes.**

**SOURCECODE:**

```
import java.util.*;

class Practical_3
{
 public static void main(String[] args)
 {

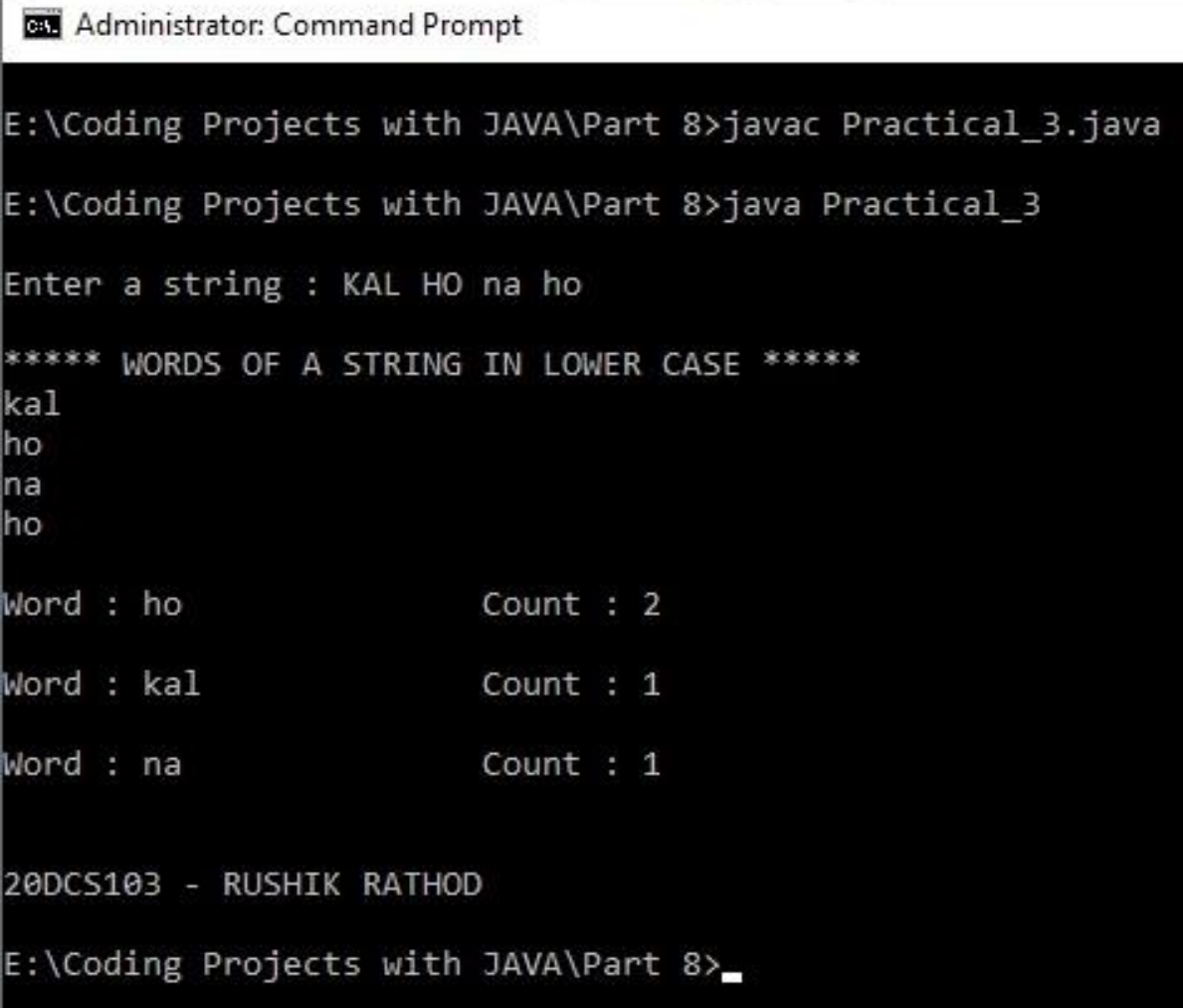
 Map<String, Integer> map = new HashMap<>();
 Scanner sc = new Scanner(System.in);
 System.out.println();
 System.out.print("Enter a string : ");
 String sentence = sc.nextLine();
 String[] tokens = sentence.split(" ");
 System.out.println();

 System.out.println("***** WORDS OF A STRING IN LOWER CASE
*****");
 for (String token : tokens)
 {
 String word = token.toLowerCase();
 System.out.println(word);
 if (map.containsKey(word))
 {
 int count = map.get(word);
 map.put(word, count + 1);
 }
 else
 {
 map.put(word, 1);
 }
 }
 }
}
```

```
 }
}

Set<String> keys = map.keySet();
TreeSet<String> sortedKeys = new TreeSet<>(keys);
System.out.println();

for (String str : sortedKeys)
{
 System.out.print("Word : " + str);
 System.out.println("\t\tCount : " + map.get(str));
 System.out.println();
}
System.out.println("\n20DCS103 - RUSHIK RATHOD");
}
}
```

**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 8>javac Practical_3.java
E:\Coding Projects with JAVA\Part 8>java Practical_3

Enter a string : KAL HO na ho

***** WORDS OF A STRING IN LOWER CASE *****
kal
ho
na
ho

Word : ho Count : 2
Word : kal Count : 1
Word : na Count : 1

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 8>_
```

**CONCLUSION:**

From this practical, I learnt to count the occurrences of words in the entered string and also learnt to print their occurrences in an alphabetical order using Map and Set classes.

4. **Write a code which counts the number of the keywords in a Java source file. Store all the keywords in a HashSet and use the contains() method to test if a word is in the keyword set.**

**SOURCECODE:**

```
import java.util.*;
import java.io.*;

public class Practical_4
{

 public static void main(String[] args) throws IOException
 {

 Scanner sc = new Scanner(System.in);
 System.out.print("\nEnter the file name that you want to scan : ");

 String f = sc.nextLine();
 File file = new File(f);
 FileReader br = new FileReader(file);
 BufferedReader fr = new BufferedReader(br);

 String keywords[] = new String[]{"abstract","assert",
 ",","boolean","break","byte","case","catch","char","class","continue","default","do",
 ",","double","else","enum ","extends","final","finally",
 "float","for","if","implements","import","instanceof","int","interface","long","nat
 ive","new","package","private","protected","public","return","short","static","stri
 ctfp","super","switch","synchronized","this","throw","throws","transient","try",
 "void","volatile","while"};

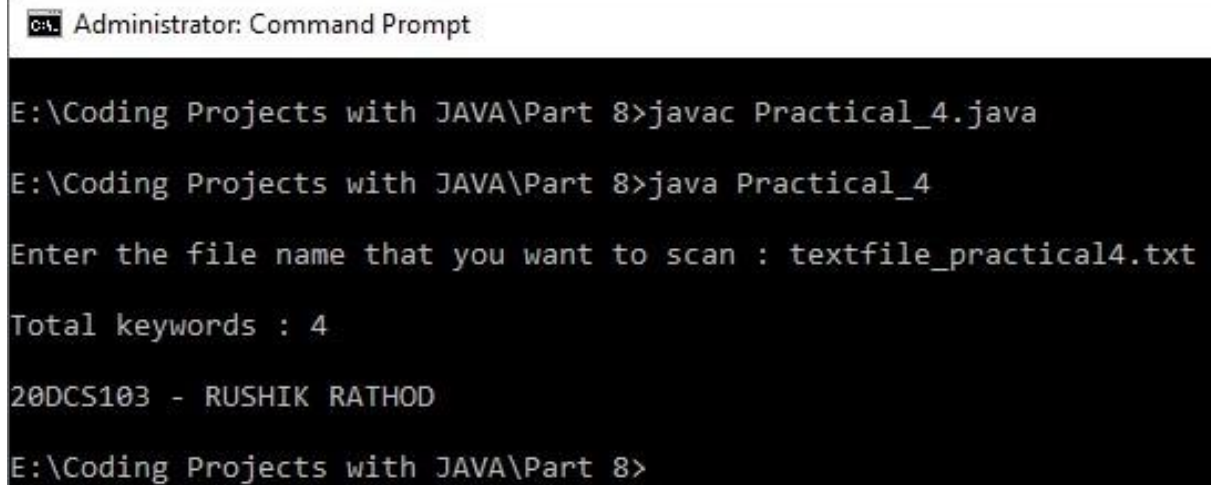
 HashSet<String> set = new HashSet<String>();

 for(int i =0;i < keywords.length; ++i)
 {
```

```
 set.add(keywords[i]);
 }

 String st;
 int count = 0 ;
 while ((st = fr.readLine()) != null)
 {
 StringTokenizer str = new StringTokenizer(st, " +/*%<>;:=&!~()");
 while(str.hasMoreTokens())
 {
 String swre = str.nextToken();
 if(set.contains(swre))
 {
 count++;
 }
 }
 }
 System.out.println();
 System.out.println("Total keywords : " + count);
 System.out.println("\n20DCS103 - RUSHIK RATHOD");
}
}
```



**OUTPUT:**

```
Administrator: Command Prompt

E:\Coding Projects with JAVA\Part 8>javac Practical_4.java

E:\Coding Projects with JAVA\Part 8>java Practical_4

Enter the file name that you want to scan : textfile_practical4.txt

Total keywords : 4

20DCS103 - RUSHIK RATHOD

E:\Coding Projects with JAVA\Part 8>
```

**CONCLUSION:**

From this practical, I learnt to check whether a java keyword is there in a provided text file or not by using 'contains()' method.

**Thank You !**