

## PRACTICAL – 1

### 1.1

**Aim:** 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 NetBeans IDE, or BlueJ and Console Programming.

#### Program:

- Object Oriented Concept: Object-Oriented Programming or OOPs refers to languages that use objects in programming. Object-oriented programming aims to implement real-world entities like inheritance, hiding, polymorphism etc in programming.

#### Comparison of Java with other object-oriented programming languages.

Parameter for comparison	C++	JAVA
Origin	Based on C language	Based on C & C++
Translator	Compiler	Compiler + Interpreter
Platform Dependency	Platform Dependent	Platform Independent
Code Execution	Direct	Executed by JVM
Approach	Bottom up	Bottom up
File Generation	.exe files	.class files
Pre-processor directives	Supported 63 Keywords	Use packages
Keywords	Supports 63 Keywords	50 defined Keywords
Datatypes (union, structures)	Supported	Not Supported

- JVM:** JVM (Java Virtual Machine) is an abstract machine that enables your computer to run a Java program. When you run the Java program, Java compiler first compiles your Java code to bytecode. Then, the JVM translates bytecode into native machine code (set of instructions that a computer's CPU executes directly).
- JRE:** JRE (Java Runtime Environment) is a software package that provides Java class libraries, Java Virtual Machine (JVM), and other components that are required to run Java applications. JVM, Javadoc, command line argument
- JDK:** JDK (Java Development Kit) is a software development kit required to develop applications in Java. When you download JDK, JRE is also downloaded with it. In addition to JRE, JDK also contains a number of development tools (compilers, JavaDoc, Java Debugger, etc).

- **Command line argument:** Java command-line argument is an argument i.e passed at the time of running the Java program. In the command line, the arguments passed from the console can be received in the java program and they can be used as input.
- **JavaDoc:** Javadoc (originally cased JavaDoc) is a documentation generator created by Sun Microsystems for the Java language (now owned by Oracle Corporation) for generating API documentation in HTML format from Java source code.
- **Eclipse or NetBeans IDE, or BlueJ and Console Programming:** These are Java integrated development environments (IDEs). These programs offer excellent debugging capabilities, open-source coding, plugins, and extensions.

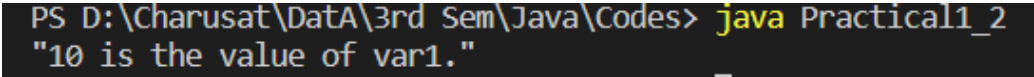
## 1.2

**Aim:** 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 :

```
public class Practical1_2
{
    public static void main(String[] args)
    {
        int var1 = 10;
        System.out.println("\n "+var1+" is the value of var1.\n");
    }
}
```

### Output:



```
PS D:\Charusat\Data\3rd Sem\Java\Codes> java Practical1_2
10 is the value of var1.
```

**Conclusion:** By this program we got to know that in Java using ‘+’ we can concatenate strings.

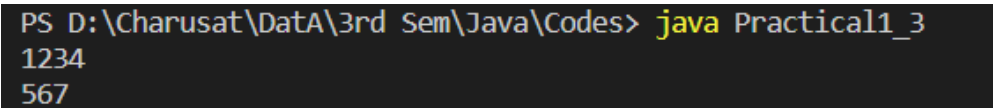
### 1.3

**Aim:** 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.

#### Program:

```
class Practical1_3
{
    public static void main(String[] args)
    {
        double a = 1234.5678;
        long b = (long)a;
        double c = (a-b)*1000;
        short d = (short)c;
        System.out.println(b);
        System.out.println(d);
    }
}
```

#### Output:



```
PS D:\Charusat\Data\3rd Sem\Java\Codes> java Practical1_3
1234
567
```

#### Conclusion:

In this program we used explicit type conversion and an error pops up regarding data loss when an implicit type conversion is done from higher data type to lower data type conversion.

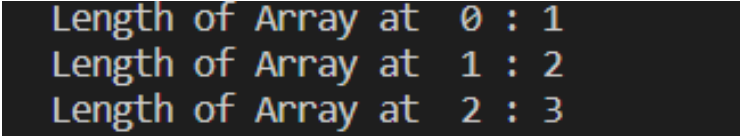
## 1.4

**Aim:** 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.

### Program:

```
public class Practical1_4
{
    public static void main(String Args[]) {
        int[][] arr={{1},{1,2},{1,2,3}};
        System.out.println("Length of Array at 0 : "+arr[0].length);
        System.out.println("Length of Array at 1 : "+arr[1].length);
        System.out.println("Length of Array at 2 : "+arr[2].length);
    }
}
```

### Output:

A screenshot of a terminal window showing the output of the Java program. The text is displayed in a monospaced font with a light blue/green color on a dark background. It shows three lines of output: 'Length of Array at 0 : 1', 'Length of Array at 1 : 2', and 'Length of Array at 2 : 3'.

```
Length of Array at 0 : 1
Length of Array at 1 : 2
Length of Array at 2 : 3
```

### Conclusion:

In this practical, we have learned how to initialize a 2D array. Also, we can find length of the array using arrayname.length funtion. This gives us the number of rows in the array and arrayname[index].length function gives us the number of columns in the array.

## 1.5

**Aim:** 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.

### Program:

```
import java.util.*;

class Practical1_5

{
    public static void main(String[] args)
    {
        System.out.println("1 for Motor");
        System.out.println("2 for Fan");
        System.out.println("3 for Tube");
        System.out.println("4 for Wires");
        System.out.println("5 for other");
        System.out.println("6 for billing");
        Scanner s=new Scanner(System.in);
        int n[]=new int[5];
        n[0]=1200;
        n[1]=230;
        n[2]=100;
        n[3]=1200;
        n[4]=320;
        int b=1;
        for(int i=0;b!=0;i++)
        {
            System.out.print("Enter your choice : ");
            b=s.nextInt();
        }
    }
}
```

```
switch(b)
{
case 1:
System.out.println("Motor Price = "+n[0]);
break;
case 2:
System.out.println("Fan Price = "+n[1]);
break;
case 3:
System.out.println("Tube Light Price = "+n[2]);
break;
case 4:
System.out.println("Wires Price = "+n[3]);
break;
case 5:
System.out.println("All other Price = "+n[4]);
break;
case 6:
double
sum=((n[0]*0.08)+n[0])+((n[1]*0.12)+n[1])+((n[2]*0.05)+n[2])+((n[3]*0.75)+n[3])+((n[4]*
0.03)+n[4]);
System.out.println("Total Bill is : "+sum);
}
}
}
}
```

**Output:**

```
1 for Motor
2 for Fan
3 for Tube
4 for Wires
5 for other
Enter your choice : 1
Motor Price = 1200
Enter your choice : 4
Wires Price = 1200
Enter your choice : 2
Fan Price = 230
Enter your choice : 6
Total Bill is : 4088.2
```

**Conclusion:**

In this program we used the switch statement to select the choice of the objects from the user and according that a bill is made by adding the corresponding tax values to the original prices of the objects.



## 1.6

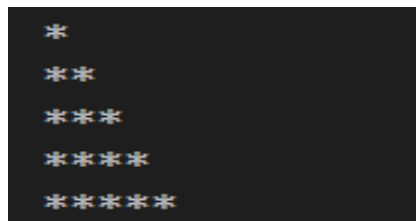
**Aim:** Write a program to show output like:

```
*  
* *  
* * *  
* * * *  
* * * * *
```

**Program:**

```
class Practical1_6  
{  
    public static void main(String[] args) {  
        for(int i=0;i<6;i++)  
        {  
            for(int j=0;j<i;j++)  
            {  
                System.out.print("*");  
            }  
            System.out.println(" ");  
        }  
    }  
}
```

**Output:**



```
*  
**  
***  
****  
*****
```

**Conclusion:**

In this program we used the concept of nested for loop and printed a pattern using it.

## PRACTICAL – 2

### 2.1

**Aim:** 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'**

#### Program:

```
import java.util.*;

class Practical2_1
{
    public static void main(String args[])
    {
        Scanner scan=new Scanner(System.in);
        System.out.print("Enter the string : ");
        String str=scan.nextLine();
        System.out.print("Enter an int : ");
        int n=scan.nextInt();
        front_times(str,n);
    }
    static void front_times(String str,int num)
    {
        for(int j=0;j<num;j++)
        {
            System.out.print(str.substring(0,3));
        }
    }
}
```

**Output:**

```
Enter the string : Rushil  
Enter an int : 4  
RusRusRusRus
```

**Conclusion:**

In this practical, we have learned how to scan a string and used a method substring of String class to get the desired output.

## 2.2

**Aim:** 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**

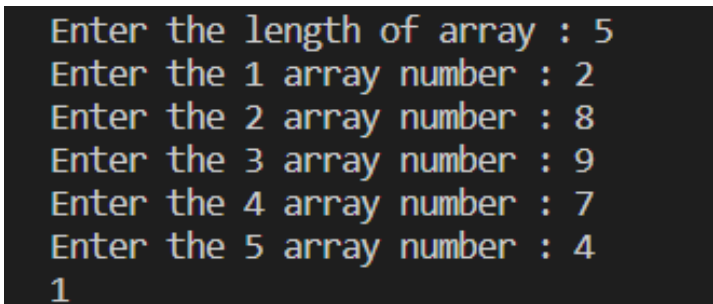
### Program:

```
import java.util.*;

class Practical2_2
{
    public static void main(String args[])
    {
        Scanner scan=new Scanner(System.in);
        System.out.print("Enter the length of array : ");
        int num=scan.nextInt();
        int arr[]=new int[num];

        for(int i=0;i<num;i++)
        {
            System.out.print("Enter the " +(i+1)+ " array number : ");
            arr[i]=scan.nextInt();
        }
        System.out.println(array_count9(arr));
    }
    static int array_count9(int arr[])
    {
        int count=0;
        for(int i=0;i<arr.length;i++)
        {
            if(arr[i]==9)
```

```
        {  
            count++;  
        }  
    }  
    return count;  
}  
}
```

**Output:**A screenshot of a terminal window with a dark background and light-colored text. It shows the output of a Java program. The text is as follows:

```
Enter the length of array : 5  
Enter the 1 array number : 2  
Enter the 2 array number : 8  
Enter the 3 array number : 9  
Enter the 4 array number : 7  
Enter the 5 array number : 4  
1
```

**Conclusion:**

In this practical, we have learned how arrays & their functions(method) work in java.

## 2.3

**Aim:** 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**

### Program:

```
import java.util.*;

class Practical2_3
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);

        System.out.print("Enter the number of elements :");

        int n=scan.nextInt();

        int arr[] = new int[n];

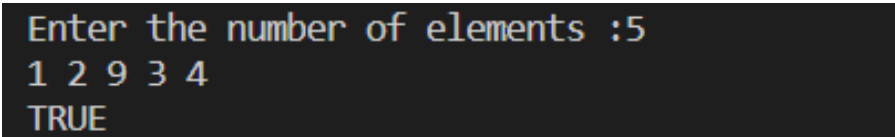
        for (int i = 0; i < n; i++)
        {
            arr[i] = scan.nextInt();
        }

        array_font9(arr);
    }

    static void array_font9(int arr[])
    {
        int count = 0;

        for (int i = 0; i < 4; i++)
        {
            if (arr[i] == 9)
            {
                count++;
            }
        }
    }
}
```

```
    }  
    }  
    if (count >= 1)  
    {  
        System.out.println("TRUE");  
    } else  
    {  
        System.out.println("Flase");  
    }  
    }  
}
```

**Output:**A screenshot of a terminal window with a dark background. It shows the output of a Java program. The first line is "Enter the number of elements :5". The second line is "1 2 9 3 4". The third line is "TRUE".

```
Enter the number of elements :5  
1 2 9 3 4  
TRUE
```

**Conclusion:**

In this practical we learned to take the input of the integer array & using the if condition we checked the array element.

## 2.4

**Aim:** 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'`

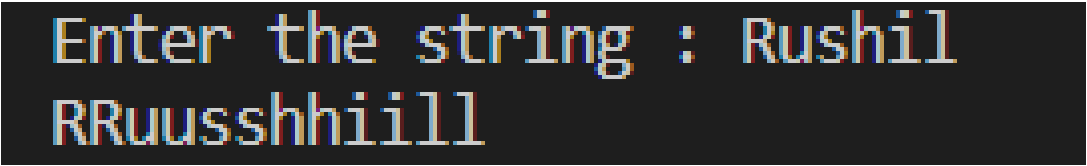
### Program :

```
import java.util.*;

class Practical2_4
{
    public static void main(String args[])
    {
        Scanner scan=new Scanner(System.in);
        System.out.print("Enter the string : ");
        String str=scan.nextLine();
        System.out.println(double_Char(str));
    }

    public static String double_Char(String stry)
    {
        String strNew = new String();
        for(int i=0; i<stry.length(); i++)
        {
            strNew = strNew+stry.charAt(i)+stry.charAt(i);
        }
        return strNew;
    }
}
```



**Output:**

```
Enter the string : Rushil
RRuusshhiill
```

**Conclusion:**

In this practical, we have learned how to scan a string and used a method charAt of String class to get the desired output.

## 2.5

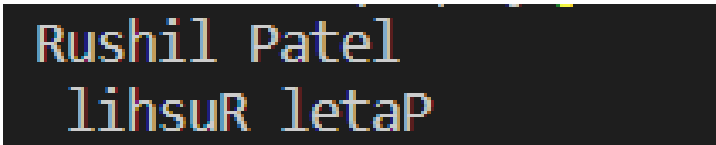
**Aim:** Write a program that will reverse the sequence of letters in each word of your chosen paragraph. For instance, “To be or not to be” would become “oT e bro ton ot eb”.

### Program:

```
import java.util.*;

class Practical2_5{
    public static void main(String arg[])
    {
        Scanner scan=new Scanner(System.in);
        String stry=scan.nextLine();
        String stry=new String();
        char ch;
        for(int i=0;i<stry.length();i++)
        {
            ch=stry.charAt(i);
            stry=ch+stry;
        }

        String str="";
        String str1=" ";
        String[] w=stry.split("\\s");
        for(int i=0;i<w.length;i++)
        {
            str=w[i]+str;
            str=str1+str;
        }
        System.out.println(str);
    }
}
```

**Output:**

```
Rushil Patel  
lihsuR letaP
```

**Conclusion:**

In this practical, we have learned how to deal with a string and split it using split method of String class.

## 2.6

**Aim:** Perform following functionalities of the string:

- Find Length of the String
- Lowercase of the String
- Uppercase of the String
- Reverse String
- Sort the string

**Program:**

```
import java.util.*;

class Scratch
{
    public static void main(String[] args)
    {
        Scanner scan=new Scanner(System.in);
        System.out.print("Enter the String : ");
        String str=scan.nextLine();

        int len=str.length();
        System.out.println("The length of the string : "+len);

        String lowstr=str.toLowerCase();
        System.out.println("Lower case of the String : "+lowstr);

        String upstr=str.toUpperCase();
        System.out.println("Upper case of the String : "+upstr);

        char[] arr = str.toCharArray();
        System.out.print("The reverse of the string : ");
        for(int i=arr.length-1;i>=0;i--)
        {
```

```
        System.out.print(arr[i]);
    }

    char arrnew[]=str.toCharArray();
    for(int i=0;i<arrnew.length;i++)
    {
        for(int j=i+1;j<arrnew.length;j++)
        {
            if(arrnew[i]>arrnew[j])
            {
                char temp;
                temp=arrnew[j];
                arrnew[j]=arrnew[i];
                arrnew[i]=temp;
            }
        }
    }
    System.out.println();
    System.out.print("The sorted string : ");
    for(int k=0;k<arrnew.length;k++)
    {
        System.out.print(arrnew[k]);
    }
}
```

**Output:**

```
Enter the String : rushil
The length of the string : 6
Lower case of the String : rushil
Upper case of the String : RUSHIL
The reverse of the string : lihsur
The sorted string : hilrsu
```

**Conclusion:**

In this practical, we have learned how String manipulation works in java.

## 2.7

**Aim:** 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

### Program:

```
import java.util.*;
class Practical2_7
{
    public static void main(String[] args)
    {
        String str="CHARUSAT University";

        System.out.println("The lenght of the string : "+str.length());

        StringBuilder string = new StringBuilder(str);
        string.setCharAt(2,'N');
        System.out.println(string);

        System.out.println(str.toUpperCase());

        String wordextr=str.substring(0,8);
        System.out.println("Extracted String : "+wordextr);
    }
}
```

**Output:**

```
The lenght of the string : 19  
CHNRUSAT University  
CHARUSAT UNIVERSITY  
Extracted String : CHARUSAT
```

**Conclusion:**

In this practical, we have learned how String manipulation works in java.



## PRACTICAL-3

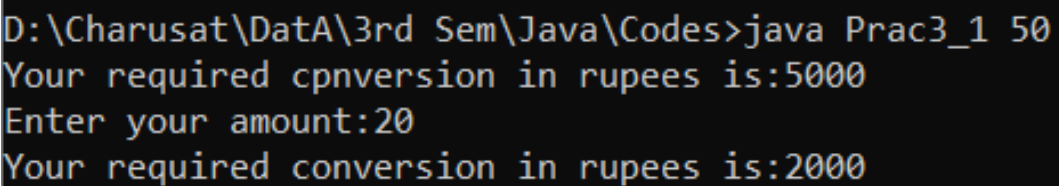
### 3.1

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

**Program:**

```
import java.util.*;
class Prac3_1
{
    public static void main(String args[])
    {
        int n;
        int x= Integer.parseInt(args[0]);
        System.out.println("Your required cpnversion in rupees is:" +(x*100));
        Scanner s= new Scanner(System.in);
        System.out.print("Enter your amount:");
        n= s.nextInt();
        System.out.println("Your required conversion in rupees is:" +(n*100));
    }
}
```

**Output:**



```
D:\Charusat\Data\3rd Sem\Java\Codes>java Prac3_1 50
Your required cpnversion in rupees is:5000
Enter your amount:20
Your required conversion in rupees is:2000
```

**Conclusion:** From this experiment we learnt to pass the argument through command line & also learnt about the Integer.parseInt() function to convert string into integer.

### 3\_2

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

**Program:**

```
import java.util.*;
class Prac3_2
{
    int b,h,a;
    Prac3_2()
    {
        b=10;
        h=4;
        System.out.println("Area using default constructor is:" + (.5*(b*h)));
    }
    Prac3_2(double a)
    {
        this.a=a;
        System.out.println("Area using one argument parameterised constructor is:" +
        (.5*(a)));
    }
    Prac3_2(double b , double h)
    {
        this.b=b;
        this.h=h;
        System.out.println("Area using two argument parameterised constructor is:" +
        (.5*(this.b * this.h)));
    }
    public static void main(String args[])
    {
        Scanner s= new Scanner(System.in);
        Set3_p2 ob1= new Set3_p2();
        System.out.println("Enter value of breath * height:");
        double n= s.nextDouble();
        Set3_p2 ob2 = new Set3_p2(n);
        System.out.println("Enter value of breath & height:");
        double x= s.nextDouble();
        double y= s.nextDouble();
        Set3_p2 ob3= new Set3_p2(x,y);
    }
}
```

**Output:**

```
D:\Charusat\Data\3rd Sem\Java\Codes>java Prac3_2
Area using default constructor is:20.0
Enter value of breath : 56
Area using one argument parameterised constructor is:28.0
Enter value of breath & height : 30 20
Area using two argument parameterised constructor is:300.0
```

**Conclusion:** From this experiment we've tried parameterised constructor.

### 3.3

**Aim:** 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.

#### Program:

```
import java.util.*;

class Employee
{
    Scanner scan=new Scanner(System.in);
    String fname,lname;
    double sal,promsal;

    Employee()
    {
        fname="";
        lname="";
        sal=0.0;
    }
    Employee(String fn, String ln, double s)
    {
        fname=fn;
        lname=ln;
        sal=s;
    }
    void setfname()
    {
        System.out.print("Enter the first name : ");
        fname=scan.next();
    }
    void setlname()
    {
        System.out.print("Enter then last name : ");
        lname=scan.next();
    }
    void setsalary()
    {
        System.out.print("Enter the salary : ");
        sal=scan.nextFloat();
    }
}
```

```
if(sal<0)
{
sal=0.0;
}
else
{
promsal=sal+(sal*0.1);
}
}
String getfname()
{
return fname;
}
String getlname()
{
return lname;
}
double getsalary()
{
return sal=(sal*12.0);
}
double getpromsal()
{
return promsal;
}
}

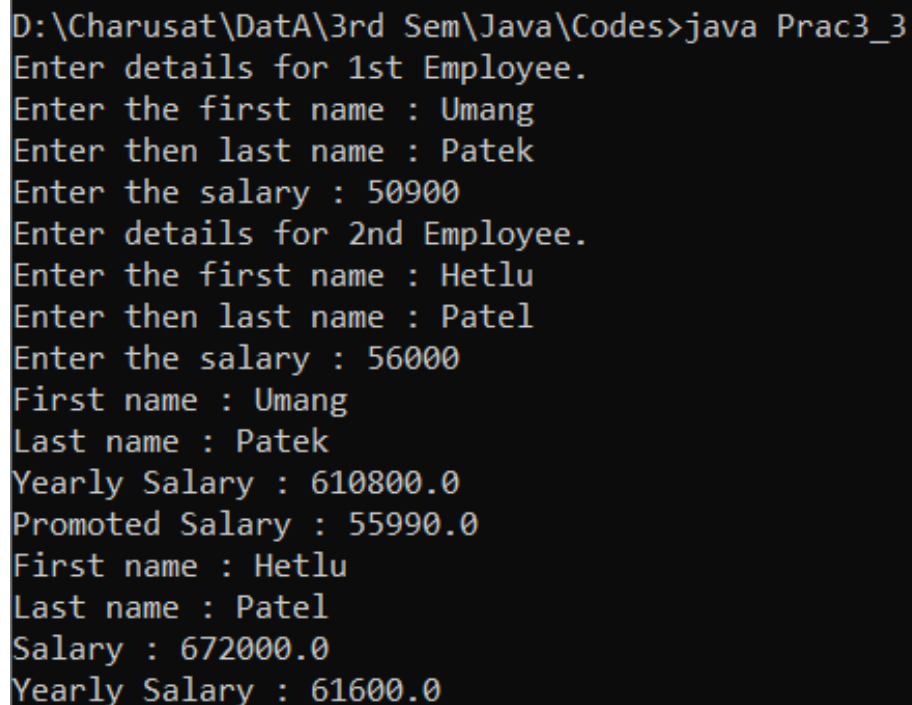
class Prac3_3
{
public static void main(String args[])
{
Employee obj=new Employee();
Employee obj1=new Employee();
System.out.println("Enter details for 1st Employee.");
obj.setfname();
obj.setlname();
obj.setsalary();

System.out.println("Enter details for 2nd Employee.");
obj1.setfname();
obj1.setlname();
obj1.setsalary();

System.out.println("First name : "+obj.getfname());
System.out.println("Last name : "+obj.getlname());
```

```
System.out.println("Yearly Salary : "+obj.getsalary());
System.out.println("Promoted Salary : "+obj.getpromsal());
System.out.println("First name : "+obj1.getfname());
System.out.println("Last name : "+obj1.getlname());
System.out.println("Salary : "+obj1.getsalary());
System.out.println("Yearly Salary : "+obj1.getpromsal());
}
}
```

### Output:

A screenshot of a terminal window showing the execution of a Java program. The prompt is 'D:\Charusat\Data\3rd Sem\Java\Codes>java Prac3\_3'. The program prompts for details for two employees. For the first employee, the inputs are 'Umang', 'Patek', and '50900'. For the second employee, the inputs are 'Hetlu', 'Patel', and '56000'. The program then displays the calculated values for each employee: 'First name : Umang', 'Last name : Patek', 'Yearly Salary : 610800.0', 'Promoted Salary : 55990.0' for the first, and 'First name : Hetlu', 'Last name : Patel', 'Salary : 672000.0', 'Yearly Salary : 61600.0' for the second.

```
D:\Charusat\Data\3rd Sem\Java\Codes>java Prac3_3
Enter details for 1st Employee.
Enter the first name : Umang
Enter then last name : Patek
Enter the salary : 50900
Enter details for 2nd Employee.
Enter the first name : Hetlu
Enter then last name : Patel
Enter the salary : 56000
First name : Umang
Last name : Patek
Yearly Salary : 610800.0
Promoted Salary : 55990.0
First name : Hetlu
Last name : Patel
Salary : 672000.0
Yearly Salary : 61600.0
```

**Conclusion:** From this experiment we can conclude that using the set & get method or using the constructor we can initialise the values of the variable.

### 3.4

**Aim:** 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.

#### Program:

```
import java.util.*;

class Date
{
    Scanner scan=new Scanner(System.in);
    int date,month,year;

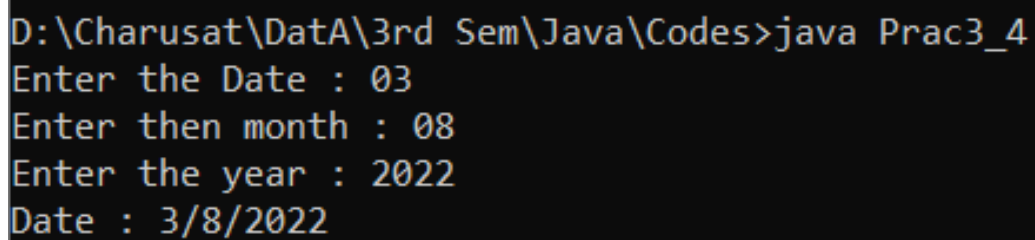
    Date()
    {
        date=0;
        month=0;
        year=0;
    }
    Date(int d, int m, int y)
    {
        date=d;
        month=m;
        year=y;
    }
    void setdate()
    {
        System.out.print("Enter the Date : ");
        date=scan.nextInt();
    }
    void setmonth()
    {
        System.out.print("Enter then month : ");
        month=scan.nextInt();
    }
    void setyear()
    {
        System.out.print("Enter the year : ");
        year=scan.nextInt();
    }
    int displaydate()
```

```
{
return date;
}
int displaymonth()
{
return month;
}
int displayyear()
{
return year;
}

}

class Prac3_4
{
public static void main(String args[])
{
Date obj=new Date();
obj.setdate();
obj.setmonth();
obj.setyear();

System.out.println("Date : "+obj.displaydate()+"/"+obj.displaymonth()+"/"+obj.displayyear());
}
}
```

**Output:**

```
D:\Charusat\Data\3rd Sem\Java\Codes>java Prac3_4
Enter the Date : 03
Enter then month : 08
Enter the year : 2022
Date : 3/8/2022
```

**Conclusion:** From this experiment we can conclude that using the set & get method or using the constructor we can initialise the values of the variable.



### 3.5

**Aim:** 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.

**Program:**

```
import java.util.*;
class Area
{
Scanner scan=new Scanner(System.in);
double area,length,breadth;

Area()
{
length=0;
breadth=0;
}

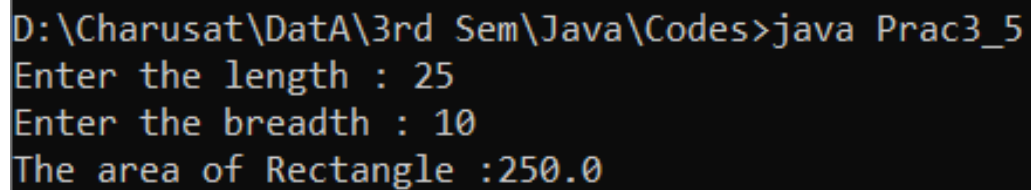
Area(double len, double bread)
{
length=len;
breadth=bread;
}
double area()
{
return (length*breadth);
}
}

class Prac3_5
{
public static void main(String args[])
{
Scanner scan=new Scanner(System.in);
System.out.print("Enter the length : ");
double length=scan.nextDouble();

System.out.print("Enter the breadth : ");
double breadth=scan.nextDouble();

Area obj=new Area(length,breadth);
```

```
System.out.print("The area of Rectangle :"+obj.area());  
}  
}
```

**Output:**A screenshot of a command prompt window with a black background and yellow text. It shows the execution of a Java program. The prompt is 'D:\Charusat\Data\3rd Sem\Java\Codes>java Prac3\_5'. The program prompts for 'Enter the length : 25' and 'Enter the breadth : 10'. The final output is 'The area of Rectangle :250.0'.

```
D:\Charusat\Data\3rd Sem\Java\Codes>java Prac3_5  
Enter the length : 25  
Enter the breadth : 10  
The area of Rectangle :250.0
```

**Conclusion:** From this experiment we can conclude that we can use the class as return type to return the values as here we have returned the double as return type.

### 3.6

**Aim:** 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.

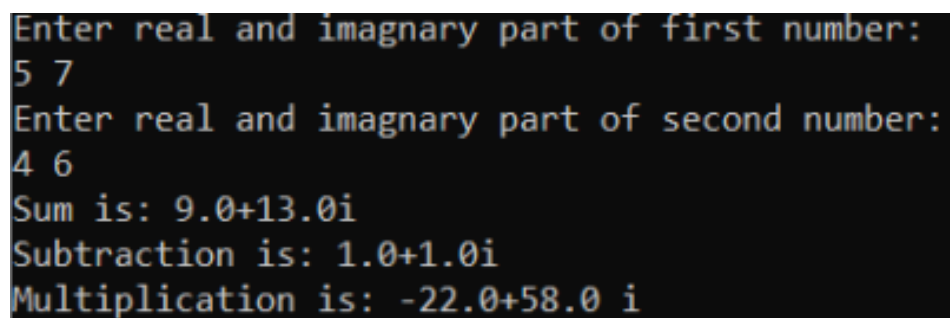
#### Program:

```
import java.util.*;
class Set3_p6
{
    float r,i;
    Set3_p6(){}
    Set3_p6(float a , float b)
    {
        r=a;
        i=b;
    }
    void Sum(Set3_p6 o1 , Set3_p6 o2)
    {
        System.out.println("Sum is: " +(o1.r + o2.r) + "+" +(o1.i + o2.i)+"i");
    }
    void sub(Set3_p6 o1 , Set3_p6 o2)
    {
        System.out.println("Subtraction is: " +(o1.r - o2.r) + "+" +(o1.i - o2.i)+"i");
    }
    void mul(Set3_p6 o1 , Set3_p6 o2)
    {
        float R,I;
        R=((o1.r * o2.r) - (o1.i * o2.i));
        I=((o1.r * o2.i) + (o1.i * o2.r));
        System.out.print("Multiplication is: " + R );
        System.out.print("+");
        System.out.print(I);
        System.out.println(" i");
    }
    class Prac3_6
    {
        public static void main(String args[])
        {
            Scanner s = new Scanner(System.in);
            System.out.print("Enter real and imaginary part of first number:");
            float a = s.nextFloat();
            float b = s.nextFloat();
        }
    }
}
```

```
Set3_p6 ob1 = new Set3_p6(a,b);  
System.out.print("Enter real and imaginary part of second number:");
```

```
float a1 = s.nextFloat();  
float b1 = s.nextFloat();  
Set3_p6 ob2 = new Set3_p6(a1,b1);  
Set3_p6 ob3 = new Set3_p6();  
ob3.Sum(ob1, ob2);  
ob3.sub(ob1, ob2);  
ob3.mul(ob1 , ob2);  
}  
}  
}
```

### Output:



```
Enter real and imaginary part of first number:  
5 7  
Enter real and imaginary part of second number:  
4 6  
Sum is: 9.0+13.0i  
Subtraction is: 1.0+1.0i  
Multiplication is: -22.0+58.0 i
```

**Conclusion:** From this experiment we've learnt to pass the variables or objects of the class as an arguments to the methods of the another class.

### 3.7

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

```
public class MethodOverloading
{
    private void methodOverloaded()
    {
        //no argument, private method
    }
    private int methodOverloaded(int i)
    {
        //code
    }
    protected int methodOverloaded(double d)
    {
        //code
    }
    public void methodOverloaded(int i, double d)
    {
        //code
    }
}
```

**Program:**

```
public class Prac3_7 {
    private void methodOverloaded() {
        System.out.println("This is private method without arguments.");
    }
    private int methodOverloaded(int i) {
        System.out.println("Single method method with integer i and its value : " + i);
        return i;
    }
    protected int methodOverloaded(double d) {
        System.out.println("single method method with double b and its value : " + d);
        int i = (int) d;
        return i;
    }
    public void methodOverloaded(int i, double d)
    {
        System.out.println("Multiple method function integer i and its value : " + i + " Another
argument is double d and value : "+d);
        methodOverloaded(10.10);
        methodOverloaded(786);
        methodOverloaded();
    }
}
```

```
public static void main(String args[]) {  
    Prac3_7 m1 = new Prac3_7();  
    m1.methodOverloaded(100,10);  
}  
}
```

**Output:**

```
D:\Charusat\DatA\3rd Sem\Java\Codes>java Prac3_7  
Multiple method function integer i and its value : 100 Another argument is double d and value : 10.0  
single method method with double b and its value : 10.1  
Single method method with integer i and its value : 786  
This is private method without arguments.
```

**Conclusion:** From this we can conclude that java does supports method overloading.

## PRACTICAL -4

### 4.1

**Aim:** 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.

#### Program:

```
import java.util.*;
class parent
{
    void prmethod()
    {
        System.out.println("This is parent class");
    }
}
class child extends parent
{
    void chmethod()
    {
        System.out.println("This is child class");
    }
}
class Inherit4_1
{
    public static void main(String args[])
    {
        parent objparent=new parent();
        child objchild=new child();
        objparent.prmethod();
        objchild.chmethod();
        objchild.prmethod();
    }
}
```

**Output:**

```
C:\Users\Administrator\Documents>javac Inherit4_1.java  
  
C:\Users\Administrator\Documents>java Inherit4_1  
This is parent class  
This is child class  
This is parent class
```

**Conclusion:**

From this practical we can conclude that using the child class we can call the methods of the child class as well as the parent class.



## 4.2

**Aim:** 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.

### Program:

```
import java.util.*;

class Member
{
    String Name,add;
    int age;
    long no;
    double salary;

    void printsalary()
    {
        System.out.println("The salary of the employee : "+salary);
    }
    void printvalues()
    {
        System.out.println("Name : "+Name);
        System.out.println("AGE : "+age);

        System.out.println("Phone Number : "+no);

        System.out.println("Address : "+add);

        System.out.println("SaLARY : "+salary);
    }
}
```

```
class Employee extends Member
{
    String spec;
}

class Manager extends Member
{
    String dep;
}

class Inherit4_1
{
    public static void main(String args[])
    {
        Employee objemp=new Employee();
        Manager objmanag=new Manager();
        objemp.Name="Umabhai Ahir";
        objemp.add="Shreedeeep";
        objemp.no=1021021021;
        objemp.age=19;
        objemp.salary=45000;

        objmanag.Name="Heltu Patel";
        objmanag.add="Surat";
        objmanag.no=1051056041;
        objmanag.age=19;
        objmanag.salary=65000;

        objemp.printvalues();
        System.out.println("-----");
        objmanag.printvalues();
    }
}
```

**Output:**

```
C:\Users\Administrator\Documents>javac Inherit4_2.java

C:\Users\Administrator\Documents>java Inherit4_2
Name : Umabhai Ahir
AGE : 19
Phone Number : 1021021021
Address : Shreedeeep
SaLARY : 45000.0
-----
Name : Heltu Patel
AGE : 19
Phone Number : 1051056041
Address : Surat
SaLARY : 65000.0
```

**Conclusion:**

From this practical we've learnt that in java also we can have hierarchical inheritance and using their objects we can use the super class methods.

### 4.3

**Aim:** 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.

#### Program:

```
import java.util.*;
class Rectangle
{
    double length, breadth;

    Rectangle(){ }
    Rectangle(double len, double bre)
    {
        length=len;
        breadth=bre;
    }

    void rectarea()
    {
        double area=length*breadth;
        System.out.println("Area of Rectangle = "+area);
    }

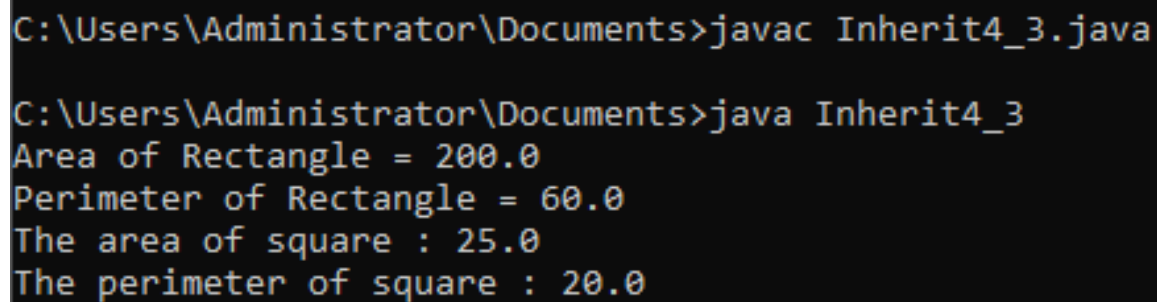
    void rectperimeter()
    {
        double peri=2*(length+breadth);
        System.out.println("Perimeter of Rectangle = "+peri);
    }
}

class Square extends Rectangle
{
    double side;
    Square(double len, double bre, double sd)
    {
        super(len,bre);
        side=sd;
    }
}
```

```
void sqarea()
{
    System.out.println("The area of square : "+(side*side));
}

void sqperi()
{
    System.out.println("The perimeter of square : "+(4*side));
}
}

class Inherit4_3
{
    public static void main(String args[])
    {
        //Rectangle object=new Rectangle(10,20);
        Square objsq=new Square(10,20,5);
        objsq.rectarea();
        objsq.rectperimeter();
        objsq.sqarea();
        objsq.sqperi();
    }
}
```

**Output:**A screenshot of a Windows command prompt window with a black background and white text. It shows the compilation and execution of a Java program. The first line shows the compilation command: 'C:\Users\Administrator\Documents>javac Inherit4\_3.java'. The second line shows the execution command: 'C:\Users\Administrator\Documents>java Inherit4\_3'. The output of the program is displayed on the following four lines: 'Area of Rectangle = 200.0', 'Perimeter of Rectangle = 60.0', 'The area of square : 25.0', and 'The perimeter of square : 20.0'.

```
C:\Users\Administrator\Documents>javac Inherit4_3.java

C:\Users\Administrator\Documents>java Inherit4_3
Area of Rectangle = 200.0
Perimeter of Rectangle = 60.0
The area of square : 25.0
The perimeter of square : 20.0
```

**Conclusion:**

From this practical we've learnt that the using super keyword we can call the constructor of the super/base class. & using this keyword we can call the constructor within the class.

## 4.4

**Aim:** 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

### Program:

```
import java.util.*;
class Shape
{
    void sh()
    {
        System.out.println("This is shape");
    }
}

class Rectangle extends Shape
{
    void rect()
    {
        System.out.println("This is Rectangular Shape");
    }
}

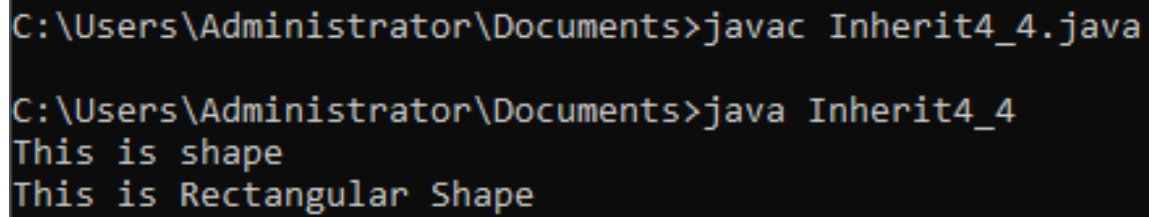
class Circle extends Shape
{
    void circle()
    {
        System.out.println("This is Circular Shape");
    }
}

class Square extends Rectangle
{
    void sq()
    {
        System.out.println("Square is Rectangle");
    }
}

class Inherit4_4
{
    public static void main(String args[])
    {

```

```
        Square objsq=new Square();  
        objsq.sh();  
        objsq.rect();  
    }  
}
```

**Output:**

```
C:\Users\Administrator\Documents>javac Inherit4_4.java  
  
C:\Users\Administrator\Documents>java Inherit4_4  
This is shape  
This is Rectangular Shape
```

**Conclusion:**

Form this practical we can conclude that using the subclass object we can access the base class or super class method or variables.

## 4.5

**Aim:** 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.

### Program:

```
import java.util.*;

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

class Undergraduate extends Degree
{
    void getdegree()
    {
        System.out.println("I am a Undergraduate");
        super.getdegree();
    }
}

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

class pract4_5
{
    public static void main(String args[])
    {
        Undergraduate objund = new Undergraduate();
        Postgraduate objpost = new Postgraduate();
        objund.getdegree();
        objpost.getdegree();
    }
}
```



**Output:**

```
C:\Users\Administrator\Documents>java pract4_5  
I am a Undergraduate  
I got a Degree  
I am a postgraduate
```

**Conclusion:**

Form this practical we can conclude that using the super keyword we can access the same name variable and method of the base class.

## 4.6

**Aim:** Write a java that implements an interface AdvancedArithmetic which contains a method signature `int divisor_sum(int n)`. You need to write a class called `MyCalculator` 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.

### Program:

```
import java.util.*;

interface AdvancedArithmetic
{
    int divisor_sum(int n);
}

class MyCaclulator implements AdvancedArithmetic
{
    public int divisor_sum(int n)
    {
        int sum=0;
        for(int i=1;i<=n;i++)
        {
            if(n%i==0)
            {
                sum=sum+i;
            }
        }
        return sum;
    }
}

class pract4_6
{
    public static void main(String args[])
    {
        AdvancedArithmetic obj = new MyCaclulator();

        System.out.println("The sum = "+obj.divisor_sum(6));
    }
}
```

**Output:**

```
C:\Users\Administrator\Documents>javac pract4_6.java  
C:\Users\Administrator\Documents>java pract4_6  
The sum = 12
```

**Conclusion:**

Form this practical we can conclude that using the interface we can inherit multiple inheritance, which can extend classes also.

## 4.7

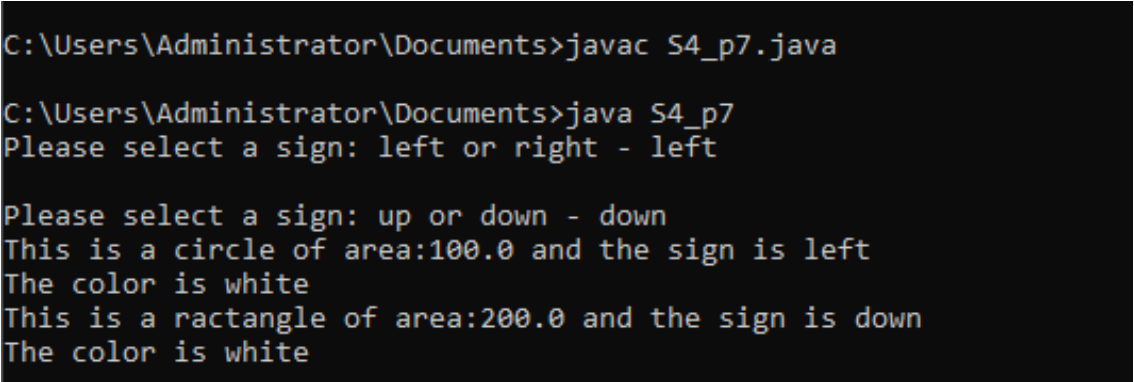
**Aim:** Assume you want to capture shapes, which can be either circles (with a radius and a color) or rectangles (with a length, width, and color). You also want to be able to create signs (to post in the campus center, for example), each of which has a shape (for the background of the sign) and the text (a String) to put on the sign. Create classes and interfaces for circles, rectangles, shapes, and signs. Write a program that illustrates the significance of interface default method.

### Program:

```
import java.util.*;
interface Shape
{
    Scanner sc = new Scanner(System.in);
    default void color()
    {
        System.out.println("The color is white");
    }
}
interface Circle extends Shape
{
    default void area(float r , String s1)
    {
        System.out.println("This is a circle of area:" + (r*r) + " and the sign is " + s1);
        Shape.super.color();
    }
}
interface Rectangle extends Shape
{
    default void area(float l , float b , String s)
    {
        System.out.println("This is a ractangle of area:" + (l*b) + " and the sign is " + s);
        Shape.super.color();
    }
}
class S4_p7 implements Circle , Rectangle
{
    static String s1, s2;
    void show()
    {
        Circle.super.area(10,s1);
        Rectangle.super.area(10,20,s2);
    }
    public static void main(String args[])
    {
```

```
System.out.print("Please select a sign: left or right - ");
s1=sc.next();
System.out.print("\nPlease select a sign: up or down - ");
s2=sc.next();
S4_p7 ob1 = new S4_p7();
ob1.show();
}
}
```

### Output:



```
C:\Users\Administrator\Documents>javac S4_p7.java

C:\Users\Administrator\Documents>java S4_p7
Please select a sign: left or right - left

Please select a sign: up or down - down
This is a circle of area:100.0 and the sign is left
The color is white
This is a ractangle of area:200.0 and the sign is down
The color is white
```

### Conclusion:

Form this practical we can conclude that the methods & variable declare in the interfaces are static by default & can be access by the interface name.

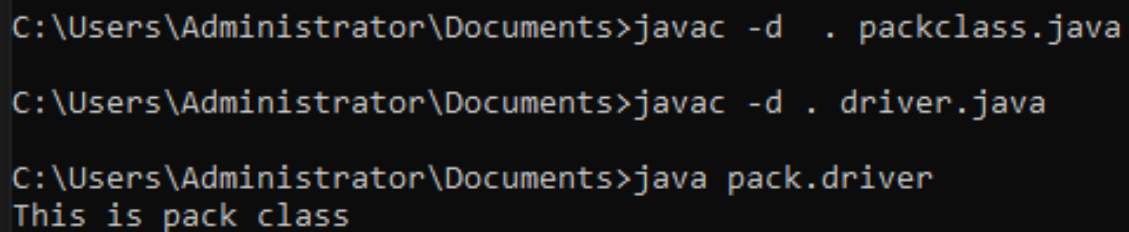
## 4.8

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

**Program:**

```
import pack.*;
class driver
{
    public static void main(String args[])
    {
        packclass obj = new packclass();
        obj.method();
    }
}
```

**Output:**



```
C:\Users\Administrator\Documents>javac -d . packclass.java
C:\Users\Administrator\Documents>javac -d . driver.java
C:\Users\Administrator\Documents>java pack.driver
This is pack class
```

**Conclusion:**

Form this practical we can conclude that using package functionalities we can access the class and methods of the one file to other using packages.

## PRACTICLA – 5

### 5.1

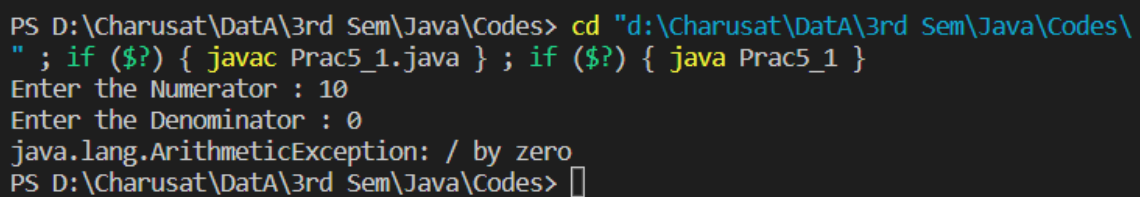
**Aim:** Write a java program which takes two integers x & y as input, you have to compute x/y. If x and y are not integers or if y is zero, exception will occur and you have to report it.

**Program:**

```
import java.util.*;
class Prac5_1
{
    public static void main(String args[])
    {
        Scanner scan=new Scanner(System.in);
        System.out.print("Enter the Numerator : ");
        int x=scan.nextInt();
        System.out.print("Enter the Denominator : ");
        int y=scan.nextInt();

        try
        {
            int ans=x/y;
        }
        catch(ArithmeticException e)
        {
            System.out.println(e);
        }
    }
}
```

**Output:**



```
PS D:\Charusat\Data\3rd Sem\Java\Codes> cd "d:\Charusat\Data\3rd Sem\Java\Codes\"
" ; if ($?) { javac Prac5_1.java } ; if ($?) { java Prac5_1 }
Enter the Numerator : 10
Enter the Denominator : 0
java.lang.ArithmeticException: / by zero
PS D:\Charusat\Data\3rd Sem\Java\Codes> 
```

**Conclusion:** Using try & catch block we can find the error & also with its class we can print the error message to the console.

## 5.2

**Aim:** 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 &quot;Invalid division&quot;.

String parsed to a numeric variable: Print &quot;Format mismatch&quot;.

Accessing an invalid index in string: Print &quot;Index is invalid&quot;.

Accessing an invalid index in array: Print &quot;Array index is invalid&quot;.

**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 &quot;MyException[param]&quot;, here para 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 &quot;Exception encountered&quot;. Finally, after the exception is handled, print &quot;Exception Handling Completed&quot;.

**Example:** For an exception of MyException class if the parameter value is 5, the message will look like MyException[5].

### Program:

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

```
class MyException extends Exception
```

```
{
    MyException(String str)
    {
        super(str);
        System.out.println(str);
    }
}
```

```
class Prac5_2
```

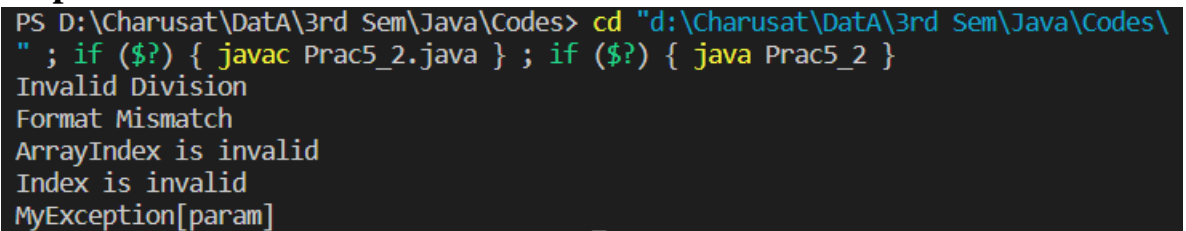
```
{
    public static void main(String args[])
    {
        Prac5_2 obj = new Prac5_2();
        obj.divisionbyzero(10,0);
        obj.stringtonum();
        obj.invalidindex();

        try
        {
            throw new MyException("MyException[param]");
        }
    }
}
```



```
    }  
    catch (MyException mye)  
    {  
  
    }  
}  
void divisionbyzero(int x, int y)  
{  
    try{  
        int z=x/y;  
    }  
    catch(ArithmeticException e)  
    {  
        System.out.println("Invalid Division");  
    }  
}  
  
void stringtonum()  
{  
    try  
    {  
        int num=Integer.parseInt("Ruhsil");  
    }  
    catch(NumberFormatException e)  
    {  
        System.out.println("Format Mismatch");  
    }  
}  
  
void invalidindex()  
{  
    String str="Rushil";  
    char []chr=str.toCharArray();  
    try  
    {  
        chr[7]='r';  
    }  
    catch(Exception e)  
    {  
        System.out.println(" ArrayIndex is invalid");  
    }  
    try  
    {  
        char c=str.charAt(7);  
    }  
    catch(Exception e)
```

```
    {  
        System.out.println("Index is invalid");  
    }  
}  
}
```

**Output:**

```
PS D:\Charusat\Data\3rd Sem\Java\Codes> cd "d:\Charusat\Data\3rd Sem\Java\Codes\  
" ; if ($?) { javac Prac5_2.java } ; if ($?) { java Prac5_2 }  
Invalid Division  
Format Mismatch  
ArrayIndex is invalid  
Index is invalid  
MyException[param]
```

**Conclusion:** From this practical we come to know about that we can make our own user defined exception by extending the Exception class.

### 5.3

**Aim:** Write a java program to generate user defined exception using “throw” and “throws” keyword. Also Write a java that differentiates checked and unchecked exceptions. (Mention at least two checked and two unchecked exception in program).

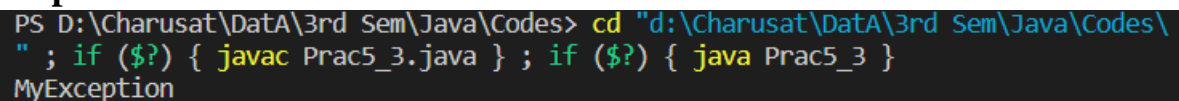
**Program:**

```
import java.util.*;

class MyException extends Exception
{
    MyException( )
    {
        super();
    }
}

class Prac5_3
{
    void method() throws MyException
    {
        throw new MyException();
    }
    public static void main(String agrs[])
    {
        Prac5_3 obj = new Prac5_3();
        try
        {
            obj.method();
        }
        catch (MyException mye)
        {
            System.out.println(mye);
        }
    }
}
```

**Output:**



```
PS D:\Charusat\Data\3rd Sem\Java\Codes> cd "d:\Charusat\Data\3rd Sem\Java\Codes\"
" ; if ($?) { javac Prac5_3.java } ; if ($?) { java Prac5_3 }
MyException
```

**Unchecked Error-1:**

```

public class JavaException
{
    public static void main(String args[])
    {
        try{
            int data=100/0;
        }
        catch(ArithmeticException e)
        {
            System.out.println(e);
        }
        System.out.println("rest of the code...");
    }
}

```

```

PS D:\Charusat\Data\3rd Sem\Java\Codes> cd "d:\Charusat\Data\3rd Sem\Java\Codes\"
; if ($?) { javac JavaException.java } ; if ($?) { java JavaException }
java.lang.ArithmeticException: / by zero
rest of the code...
PS D:\Charusat\Data\3rd Sem\Java\Codes> 

```

**Unchecked Error-2:**

```

public class JavaException
{
    public static void main(String args[]) {
        try{
            int arr[]={1,2,3,4,5};
            System.out.println(arr[7]);
        }
        catch(ArrayIndexOutOfBoundsException e){
            System.out.println("The specified index does not exist " +
                "in array. Please correct the error.");
        }
    }
}

```

```

PS D:\Charusat\Data\3rd Sem\Java\Codes> cd "d:\Charusat\Data\3rd Sem\Java\Codes\"
; if ($?) { javac JavaException.java } ; if ($?) { java JavaException }
The specified index does not exist in array. Please correct the error.

```

**Checked Error-1:**

```
public class JavaException
{
    public static void main(String[] args)
        throws ClassNotFoundException
    {

        try {
            Class temp = Class.forName("gfg");
        }
        catch (ClassNotFoundException e)
        {
            System.out.println("Class does not exist check the name of the class");
        }
    }
}
```

```
PS D:\Charusat\Data\3rd Sem\Java\Codes> cd "d:\Charusat\Data\3rd Sem\Java\Codes\"
; if ($?) { javac JavaException.java } ; if ($?) { java JavaException }
Class does not exist check the name of the class
```

**Checked Error-2:**

```
import java.io.*;
public class JavaException
{
    public static void main(String[] args)
    {
        try
        {
            throw new IOException("Device Error");
        }
        catch(IOException e)
        {
            System.out.println("Checked error 2");
        }
    }
}
```

```
PS D:\Charusat\Data\3rd Sem\Java\Codes> cd "d:\Charusat\Data\3rd Sem\Java\Codes\"
; if ($?) { javac JavaException.java } ; if ($?) { java JavaException }
Checked error 2
```