

# **ADHI COLLEGE OF ENGINEERING AND TECHNOLOGY**

Approved by AICTE, New Delhi, Permanent affiliation status by Anna University, Chennai,  
Accredited by NAAC, New Delhi, Recognized U/S 12 (B) & 2 (F) of UGC Act 1956  
An ISO Certified Institution

Sankarapuram, Near Walajabad, Kanchipuram District, Pin: 631 605

## **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**(2023 - 2024)**



<b>STUDENT NAME</b>	
<b>REGISTER NO</b>	
<b>SUBJECT CODE</b>	<b>CS3381</b>
<b>SUBJECT NAME</b>	<b>Object Oriented Programming Laboratory</b>
<b>YEAR / SEMESTER</b>	<b>II / III</b>

# ADHI COLLEGE OF ENGINEERING AND TECHNOLOGY



Approved by AICTE, New Delhi, Permanent affiliation status by Anna University, Chennai,  
Accredited by NAAC, New Delhi, Recognized U/S 12 (B) & 2 (F) of UGC Act 1956  
An ISO Certified Institution

Sankarapuram, Near Walajabad, Kanchipuram District, Pin: 631 605

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

### LABORATORY RECORD NOTE BOOK

2023 - 2024

REGISTER NO:

This is to certify that this bonafide record of the work done by  
Mr./Ms. \_\_\_\_\_ of the **II Year**  
**B.E / B.Tech., Computer Science and Engineering** in the Department of Computer  
Science and Engineering in the **CS3381 - Object Oriented Programming Laboratory**  
in the **III Semester** during the year 2023- 2024.

Staff In-Charge

Head of the Department

Submitted to the University Examination held on \_\_\_\_\_

Internal Examiner

External Examiner

## INDEX

EX. NO	DATE	NAME OF THE EXPERIMENT	PAGE NO.	STAFF SIGN
1.a		Solve problems by using sequential search algorithms		
1.b		Solve problems by using binary search algorithms		
1.c		Solve problems by using selection sorting algorithms		
1.d		Solve problems by using insertion sorting algorithms		
2.a		Develop stack data structures using classes and objects.		
2.b		Develop queue data structures using classes and objects.		
3		Develop a java application with an Employee class with Emp_name, Emp_id, Address, Mail_id, Mobile_no as members. Inherit the classes, Programmer, Assistant Professor, Associate Professor and Professor from employee class. Add Basic Pay (BP) as the member of all the inherited classes with 97% of BP as DA, 10 % of BP as HRA, 12% of BP as PF, 0.1% of BP for staff club funds. Generate pay slips for the employees with their gross and net salary.		
4		Write a Java Program to create an abstract class named Shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method printArea( ) that prints the area of the given shape.		
5		Solve the above problem using an interface.		

6		Implement exception handling and creation of user defined exceptions.		
7		Write a java program that implements a multi-threaded application that has three threads. First thread generates a random integer every 1 second and if the value is even, the second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of the cube of the number.		
8		Write a program to perform file operations.		
9		Develop applications to demonstrate the features of generics classes.		
10		Develop applications using JavaFX controls, layouts and menus.		
11		Develop a mini project for any application using Java concepts.		

<b>EXP. NO: 1. a</b>	<b>Solve problems by using sequential search algorithm</b>
<b>DATE:</b>	

**Aim:**

**Algorithm:**

1. Start from the first element of the array.
2. Compare each element of the array with the given element x.
3. If the element matches, return the index.
4. If the element does not match, move to the next element in the array.
5. Repeat steps 2-4 until the end of the array is reached.
6. If the element is not found after checking all elements, return a message indicating that the element is not present.

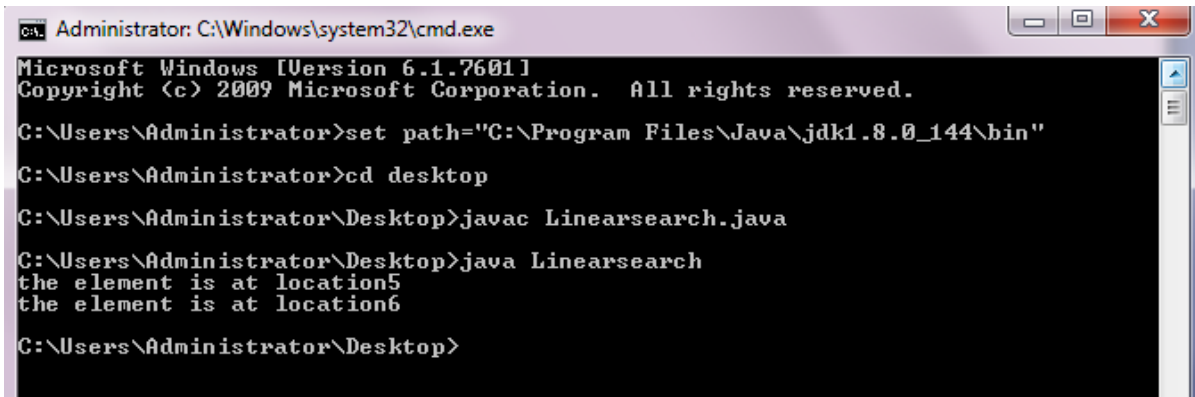
**Program:**

```

public class LinearSearch
{
    public static void main(String[] args)
    {
        int[] a = {1, 5, -2, 8, 7, 11, 40, 32};
        System.out.println("the element is at location " + Find(a, 7));
        System.out.println("the element is at location " + Find(a, 11));
    }
    public static int Find(int[] a, int key)
    {
        for (int i = 0; i < a.length; i++)
        {
            if (a[i] == key)
                return i + 1;
        }
        return -1;
    }
}

```

### Output:



```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>set path="C:\Program Files\Java\jdk1.8.0_144\bin"
C:\Users\Administrator>cd desktop
C:\Users\Administrator\Desktop>javac Linearsearch.java
C:\Users\Administrator\Desktop>java Linearsearch
the element is at location5
the element is at location6
C:\Users\Administrator\Desktop>
```

### Result:

EXP. NO: 1.b	Solve problems by using binary search algorithm
DATE:	

**Aim:**

**Algorithms:**

1. Set two pointers: left at the beginning of the array and right at the end of the array.
2. While left is less than or equal to right:
  - Calculate the middle index as  $\text{mid} = (\text{left} + \text{right}) // 2$ .
  - If the middle element is equal to the target x, return mid.
  - If the target x is less than the middle element, set  $\text{right} = \text{mid} - 1$ .
  - If the target x is greater than the middle element, set  $\text{left} = \text{mid} + 1$ .
3. If the element is not found after the search, return a message indicating that the element is not present.

**Program:**

```
public class BinSearch {
    public static void main(String[] args) {
        int[] a = {10, 20, 30, 40, 50, 60};
        int key = 40;
        Find(a, 0, 5, key);
    }

    public static void Find(int[] a, int low, int high, int key)
    {
        int mid;
        if (low > high) {
            System.out.println("error!! the element is not present in the list");
            return;
        }
        mid = (low + high) / 2;
        if (key == a[mid])
            System.out.println("the element is present at location " + (mid + 1));
        else if (key < a[mid])
            Find(a, low, mid - 1, key);
        else if (key > a[mid])
            Find(a, mid + 1, high, key);
    }
}
```

**Output:**

```
C:\Users\Administrator\Desktop>javac binsearch.java  
C:\Users\Administrator\Desktop>java binsearch  
the element is present at location4  
C:\Users\Administrator\Desktop>
```

**Result:**



<b>EXP. NO: 1.c</b>	<b>Solve problems by using selection sort algorithm</b>
<b>DATE:</b>	

**Aim:**

**Algorithms:**

Step 1: Take an input array arr[] of length n.

Step 2: Initialize variables: Set n as the length of the array (n = arr.length).

Step 3: Begin the sorting process:

Outer Loop: Initialize i from 0 to n-1. Set minIndex to i.

Inner Loop: Initialize j from i+1 to n.

If arr[j] is less than arr[minIndex], update minIndex to j.

Swap Elements: If minIndex is not equal to i, swap arr[i] with arr[minIndex].

Increment: Increment i by 1.

Step 4: Repeat the outer loop (Step 3) until i reaches n-1.

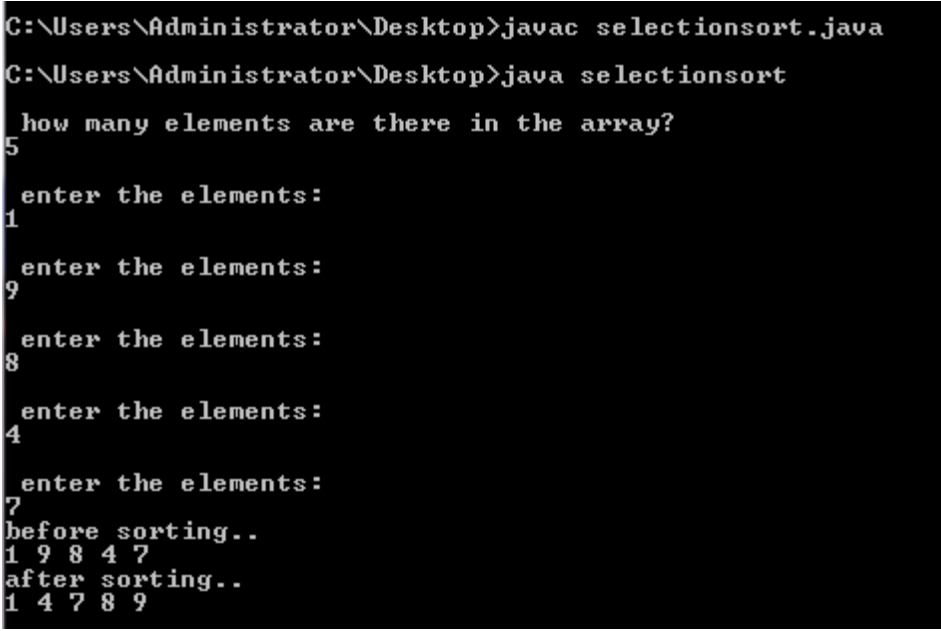
Step 5: The array arr[] is now sorted in ascending order.

**Program:**

```
import java.util.*;
class SelectionSort {
    void sort(int a[], int n) {
        for (int i = 0; i < n - 1; i++) {
            int k = i;
            for (int j = i + 1; j < n; j++)
                if (a[j] < a[k])
                    k = j;
            int temp = a[k];
            a[k] = a[i];
            a[i] = temp;
        }
    }
    void display(int a[], int n) {
        for (int i = 0; i < n; ++i)
            System.out.print(a[i] + " ");
        System.out.println();
    }
    public static void main(String[] args) {
        SelectionSort obj = new SelectionSort();
        int[] a = new int[10];
        System.out.println("\nHow many elements are there in the array?");
```

```
Scanner sc = new Scanner(System.in);
int n = sc.nextInt();
for (int i = 0; i < n; i++) {
    System.out.println("\nEnter the elements:");
    a[i] = sc.nextInt();
}
System.out.println("Before sorting...");
obj.display(a, n);
obj.sort(a, n);
System.out.println("After sorting...");
obj.display(a, n);
}
}
```

### **Output:**



```
C:\Users\Administrator\Desktop>javac selectionsort.java
C:\Users\Administrator\Desktop>java selectionsort
how many elements are there in the array?
5
enter the elements:
1
enter the elements:
9
enter the elements:
8
enter the elements:
4
enter the elements:
7
before sorting..
1 9 8 4 7
after sorting..
1 4 7 8 9
```

### **Result:**

<b>EXP. NO: 1.d</b>	<b>Solve problems by using insertion sort algorithm</b>
<b>DATE:</b>	

**Aim:**

**Algorithm:**

Step 1: Take an input array arr[] of length n.

Step 2: Initialize variables: Set n as the length of the array (n = arr.length).

Initialize key as an element of the array.

Step 3: Begin the sorting process:

Outer Loop: Start from the second element (i = 1) to the end of the array. Set key as arr[i].

Inner Loop: Initialize j as i-1.

While j >= 0 and arr[j] > key:

Move elements of arr[] greater than key one position ahead of their current position.

Decrement j by 1.

Place Element: Insert key at position j+1.

Step 4: Repeat the outer loop (Step 3) until all elements are in their sorted positions.

Step 5: The array arr[] is now sorted in ascending order.

**Program:**

```
import java.util.*;
class InsertionSort {
    void sort(int a[], int n) {
        for (int i = 1; i < n; ++i) {
            int key = a[i];
            int j = i - 1;
            while (j >= 0 && a[j] > key) {
                a[j + 1] = a[j];
                j = j - 1;
            }
            a[j + 1] = key;
        }
    }
    void display(int a[], int n) {
        for (int i = 0; i < n; ++i) {
            System.out.print(a[i] + " ");
        }
        System.out.println();
    }
    public static void main(String args[]) {
        InsertionSort obj = new InsertionSort();
        int[] a = new int[10];
        System.out.println("\nHow many elements are there in the array?");
```

```

Scanner sc = new Scanner(System.in);
int n = sc.nextInt();

for (int i = 0; i < n; i++) {
    System.out.println("\nEnter the element:");
    a[i] = sc.nextInt();
}
System.out.println("Before sorting...");
obj.display(a, n);
obj.sort(a, n);
System.out.println("After sorting...");
obj.display(a, n);
}
}

```

### **Output:**

```

C:\Users\Administrator>set path="C:\Program Files\Java\jdk1.8.0_144\bin"
C:\Users\Administrator>cd desktop
C:\Users\Administrator\Desktop>javac Insertionsort.java
C:\Users\Administrator\Desktop>java Insertionsort
how many elements are there in an array?
5
enter the element:
79
enter the element:
95
enter the element:
2
enter the element:
84
enter the element:
57
before sorting....
79
95
2
84
57
after sorting...
2
57
79
84
95
C:\Users\Administrator\Desktop>

```

### **Result:**

EXP. NO: 2.a	Develop stack data structures using classes and objects
DATE:	

**Aim:**

**Algorithm:**

Step 1: Initialize an empty stack.

Step 2: Push an element onto the stack:

- Increment the stack pointer.
- Place the new element at the top of the stack.

Step 3: Pop an element from the stack:

- If the stack is not empty:
- Retrieve the element at the top of the stack.
- Decrement the stack pointer.
- Else, display an error indicating the stack is empty.

Step 4: Peek at the top element of the stack:

- If the stack is not empty:
- Return the element at the top of the stack without removing it.
- Else, display an error indicating the stack is empty.

Step 5: Check if the stack is empty:

- Return true if the stack pointer is zero, indicating an empty stack.
- Return false otherwise.

**Program:**

```
class StackDemo
{
    int size;
    int stack[];
    int top;
    StackDemo(int size)
    {
        this.size = size;
        this.stack = new int[size];
        this.top = -1;
    }

    public void push(int element)
    {
        if (!isStackFull())
        {
            top++;
            stack[top] = element;
            System.out.println("Element Pushed on Stack is :" + element);
        }
        else
        {
            System.out.println("Cannot insert Stack is full...");
        }
    }
}
```

```
public int pop()
{
    if (!isEmpty())
    {
        int item;
        item = stack[top];
        top--;
        System.out.println("Element Popped from Stack is:" + item);
        return item;
    }
    else
    {
        System.out.println("Stack is empty...");
        return -1;
    }
}
```

```
public int peek()
{
    if(!this.isEmpty())
    {
        return stack[top];
    }
    else
    {
        System.out.println("Stack is Empty");
        return -1;
    }
}
```

```
public (boolean isEmpty)
{
    return (top == -1);
}
```

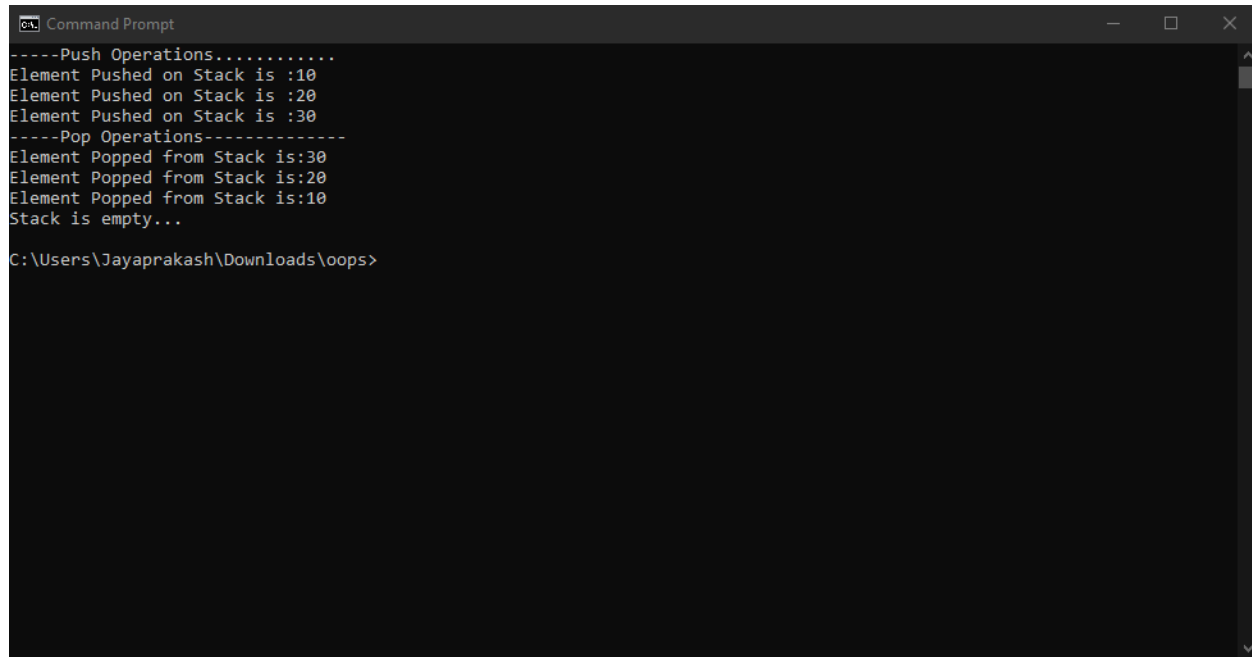
```
public boolean isStackFull()
{
    return (size-1 == top);
}
```

```
public static void main(String[] args)
{
    StackDemo obj = new StackDemo(10);
    System.out.println("-----Push Operations.....");
    obj.push(10);
    obj.push(20);
    obj.push(30);

    System.out.println("-----Pop Operations-----");
    obj.pop();
    obj.pop();
    obj.pop();
    obj.pop();

}
}
```

## Output:

A screenshot of a Windows Command Prompt window. The title bar reads "Command Prompt". The window contains the following text:

```
-----Push Operations-----  
Element Pushed on Stack is :10  
Element Pushed on Stack is :20  
Element Pushed on Stack is :30  
-----Pop Operations-----  
Element Popped from Stack is:30  
Element Popped from Stack is:20  
Element Popped from Stack is:10  
Stack is empty...  
  
C:\Users\Jayaprakash\Downloads\oops>
```

The text is displayed in a monospaced font on a black background. The prompt character is a greater-than sign (>).

## Result:

<b>EXP. NO: 2.b</b>	<b>Develop queue data structures using classes and objects</b>
<b>DATE:</b>	

**Aim:**

**Algorithm:**

Step 1: Node Class:

Define a class 'Queue Node' with attributes for data and a reference to the next node.

Step 2: Queue Class:

Define a class 'Queue' with private attributes for the front and rear nodes.

Constructor initializes front and rear to 'null'.

Step 3: Enqueue Operation:

Create a method 'enqueue(data)' in the 'Queue' class.

Create a new 'Queue Node' with the given data.

If the queue is empty, set both front and rear to the new node; otherwise, update the rear's next reference.

Step 4: Dequeue Operation:

Create a method dequeue() in the Queue class.

Check if the queue is empty; if so, indicate an error or return a special value.

Otherwise, remove the node at the front and update the front reference.

**Program:**

```
public class QueueDemo
{
    private int capacity;
    private int[] que;
    private int front;
    private int rear;
    private int size;
    public QueueDemo(int n)
    {
        this.capacity = n;
        this.que = new int[n];
        front = 0;
        rear = -1;
        size = 0;
    }

    public void insert(int item)
    {
        if(isQueueFull())
        {
            System.out.println("Queue is full!"); return;
        }
        que[++rear] = item;
        size++;
    }
}
```



```
System.out.println("Insertion to queue: " + item);
}
```

```
public int remove()
{
if(isQueueEmpty())
{
throw new RuntimeException("Queue is empty");
}
int item = que[front++];
if(front == capacity)
{
front = 0;
}
size--;
return item;
}
```

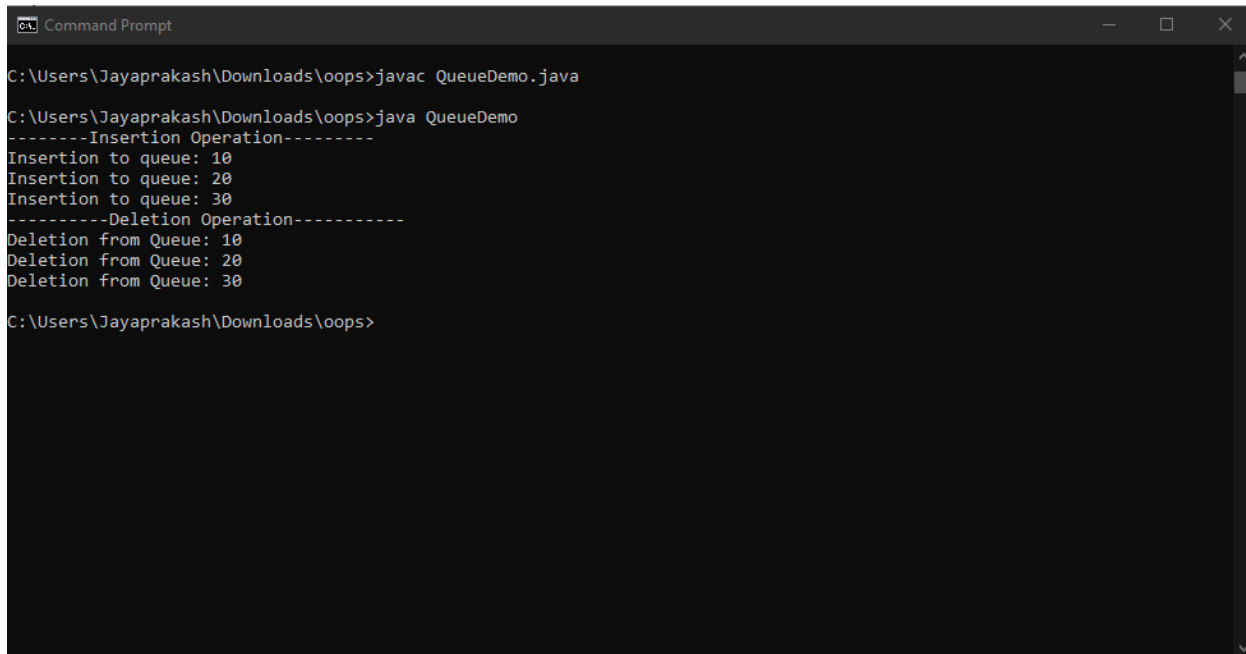
```
public int peek()
{
return que[front];
}
```

```
public boolean isQueueFull()
{
return (capacity == size);
}
```

```
public boolean isQueueEmpty()
{
return (size == 0);
}
```

```
public static void main(String[] args)
{
QueueDemo obj = new QueueDemo(10);
System.out.println("-----Insertion Operation-----");
obj.insert(10);
obj.insert(20);
obj.insert(30);
System.out.println("-----Deletion Operation----- ");
System.out.println("Deletion from Queue: " + obj.remove());
System.out.println("Deletion from Queue: " + obj.remove());
System.out.println("Deletion from Queue: " + obj.remove());
}
}
```

## **Output:**



```
Command Prompt
C:\Users\Jayaprakash\Downloads\oops>javac QueueDemo.java
C:\Users\Jayaprakash\Downloads\oops>java QueueDemo
-----Insertion Operation-----
Insertion to queue: 10
Insertion to queue: 20
Insertion to queue: 30
-----Deletion Operation-----
Deletion from Queue: 10
Deletion from Queue: 20
Deletion from Queue: 30
C:\Users\Jayaprakash\Downloads\oops>
```

## **Result:**

**EXP. NO: 3**

**DATE:**

**Generate pay slips for the employees with their gross and net salary**

**Aim:**

**Algorithm:**

- Step 1: Create the class employee with name, Empid, address, mailid, mobileneno as members.  
Step 2: Inherit the classes programmer, asstprofessor, associateprofessor and professor from employee class.  
Step 3: Add Basic Pay (BP) as the member of all the inherited classes.  
Step 4: Calculate DA as 97% of BP, HRA as 10% of BP, PF as 12% of BP, Staff club fund as 0.1% of BP.  
Step 5: Calculate gross salary and net salary.  
Step 6: Generate payslip for all categories of employees.  
Step 7: Create the objects for the inherited classes and invoke the necessary methods to display the Payslip.

**Program:**

```
import java.util.*;
class employee
{
    int empid;
    long mobile;
    String name, address, mailid;
    Scanner get = new Scanner(System.in);
    void getdata()
    {
        System.out.println("Enter Name of the Employee");
        name = get.nextLine();
        System.out.println("Enter Mail id");
        mailid = get.nextLine();
        System.out.println("Enter Address of the Employee:");
        address = get.nextLine();
        System.out.println("Enter employee id ");
        empid = get.nextInt();
        System.out.println("Enter Mobile Number");
        mobile = get.nextLong();
    }
    void display()
    {
        System.out.println("Employee Name: "+name);
        System.out.println("Employee id : "+empid);
        System.out.println("Mail id : "+mailid);
        System.out.println("Address: "+address);
        System.out.println("Mobile Number: "+mobile);
    }
}
```

```

class programmer extends employee
{
double salary,bp,da,hra,pf,club,net,gross;
void getprogrammer()
{
System.out.println("Enter basic pay");
bp = get.nextDouble();
}
void calculateprog()
{
da=(0.97*bp);
hra=(0.10*bp);
pf=(0.12*bp);
club=(0.1*bp);
gross=(bp+da+hra);
net=(gross-pf-club);
System.out.println("*****");
System.out.println("PAY SLIP FOR PROGRAMMER");
System.out.println("*****");
System.out.println("Basic Pay:Rs"+bp);
System.out.println("DA:Rs"+da);
System.out.println("PF:Rs"+pf);
System.out.println("HRA:Rs"+hra);
System.out.println("CLUB:Rs"+club);
System.out.println("GROSS PAY:Rs"+gross);
System.out.println("NET PAY:Rs"+net);
}
}
class asstprofessor extends employee
{
double salary,bp,da,hra,pf,club,net,gross;
void getasst()
{
System.out.println("Enter basic pay");
bp = get.nextDouble();
}
void calculateasst()
{
da=(0.97*bp);
hra=(0.10*bp);
pf=(0.12*bp);
club=(0.1*bp);
gross=(bp+da+hra);
net=(gross-pf-club);
System.out.println("*****");
System.out.println("PAY SLIP FOR ASSISTANT PROFESSOR");
System.out.println("*****");
System.out.println("Basic Pay:Rs"+bp);
System.out.println("DA:Rs"+da);
System.out.println("HRA:Rs"+hra);
System.out.println("PF:Rs"+pf);
System.out.println("CLUB:Rs"+club);
System.out.println("GROSS PAY:Rs"+gross);
System.out.println("NET PAY:Rs"+net);
}
}

```

```

class associateprofessor extends employee
{
double salary,bp,da,hra,pf,club,net,gross;
void getassociate()
{
System.out.println("Enter basic pay");
bp = get.nextDouble();
}
void calculateassociate()
{
da=(0.97*bp);
hra=(0.10*bp);
pf=(0.12*bp);
club=(0.1*bp);
gross=(bp+da+hra);
net=(gross-pf-club);
System.out.println("*****");
System.out.println("PAY SLIP FOR ASSOCIATE PROFESSOR");
System.out.println("*****");
System.out.println("Basic Pay:Rs"+bp);
System.out.println("DA:Rs"+da);
System.out.println("HRA:Rs"+hra);
System.out.println("PF:Rs"+pf);
System.out.println("CLUB:Rs"+club);
System.out.println("GROSS PAY:Rs"+gross);
System.out.println("NET PAY:Rs"+net);
}
}

class professor extends employee
{
double salary,bp,da,hra,pf,club,net,gross;
void getprofessor()
{
System.out.println("Enter basic pay");
bp = get.nextDouble();
}
void calculateprofessor()
{
da=(0.97*bp);
hra=(0.10*bp);
pf=(0.12*bp);
club=(0.1*bp);
gross=(bp+da+hra);
net=(gross-pf-club);
System.out.println("*****");
System.out.println("PAY SLIP FOR PROFESSOR");
System.out.println("*****");
System.out.println("Basic Pay:Rs"+bp);
System.out.println("DA:Rs"+da);
System.out.println("HRA:Rs"+hra);
System.out.println("PF:Rs"+pf);
System.out.println("CLUB:Rs"+club);
System.out.println("GROSS PAY:Rs"+gross);
System.out.println("NET PAY:Rs"+net);
}
}

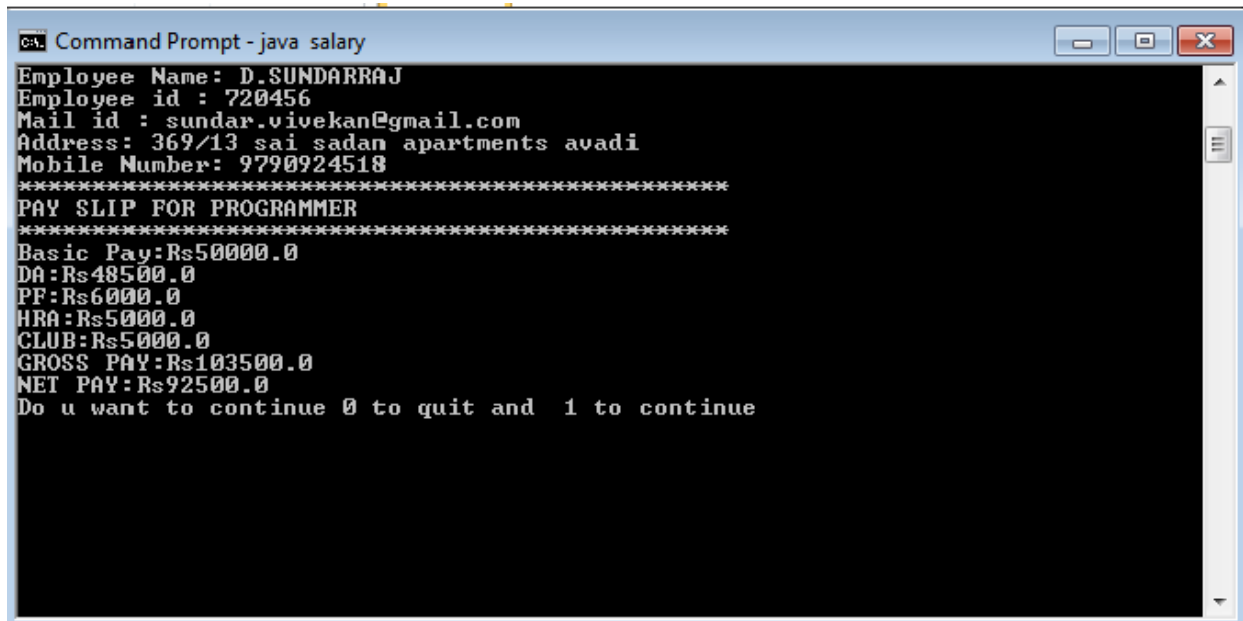
```

```

class salary
{
public static void main(String args[])
{
int choice,cont;
do
{
System.out.println("PAYROLL");
System.out.println(" 1.PROGRAMMER \t 2.ASSISTANT PROFESSOR \t 3.ASSOCIATE
PROFESSOR \t 4.PROFESSOR ");
Scanner c = new Scanner(System.in);
choice=c.nextInt();
switch(choice)
{
case 1:
{
programmer p=new programmer();
p.getdata();
p.getprogrammer();
p.display();
p.calculateprog();
break;
}
case 2:
{
asstprofessor asst=new asstprofessor();
asst.getdata();
asst.getasst();
asst.display();
asst.calculateasst();
break;
}
case 3:
{
associateprofessor asso=new associateprofessor();
asso.getdata();
asso.getassociate();
asso.display();
asso.calculateassociate();
break;
}
case 4:
{
professor prof=new professor();
prof.getdata();
prof.getprofessor();
prof.display();
prof.calculateprofessor();
break;
}
}
System.out.println("Do u want to continue 0 to quit and  1 to continue ");
cont=c.nextInt();
}while(cont==1);
}
}

```

## Output:



```
C:\> Command Prompt - java salary
Employee Name: D.SUNDARRAJ
Employee id : 720456
Mail id : sundar.vivekan@gmail.com
Address: 369/13 sai sadan apartments avadi
Mobile Number: 9790924518
*****
PAY SLIP FOR PROGRAMMER
*****
Basic Pay:Rs50000.0
DA:Rs48500.0
PF:Rs6000.0
HRA:Rs5000.0
CLUB:Rs5000.0
GROSS PAY:Rs103500.0
NET PAY:Rs92500.0
Do u want to continue 0 to quit and 1 to continue
```

## Result:

<b>EXP. NO: 4</b>	<b>Calculate area using abstract class</b>
<b>DATE:</b>	

**Aim:**

**Algorithm:**

Step 1: Create an abstract class named shape that contains two integers and an empty method named printarea().

Step 2: Provide three classes named rectangle, triangle and circle such that each one of the classes extends the class Shape.

Step 3: Each of the inherited class from shape class should provide the implementation for the method printarea().

Step 4: Get the input and calculate the area of rectangle, circle and triangle .

Step 5.: In the shapeclass , create the objects for the three inherited classes and invoke the methods and display the area values of the different shapes.

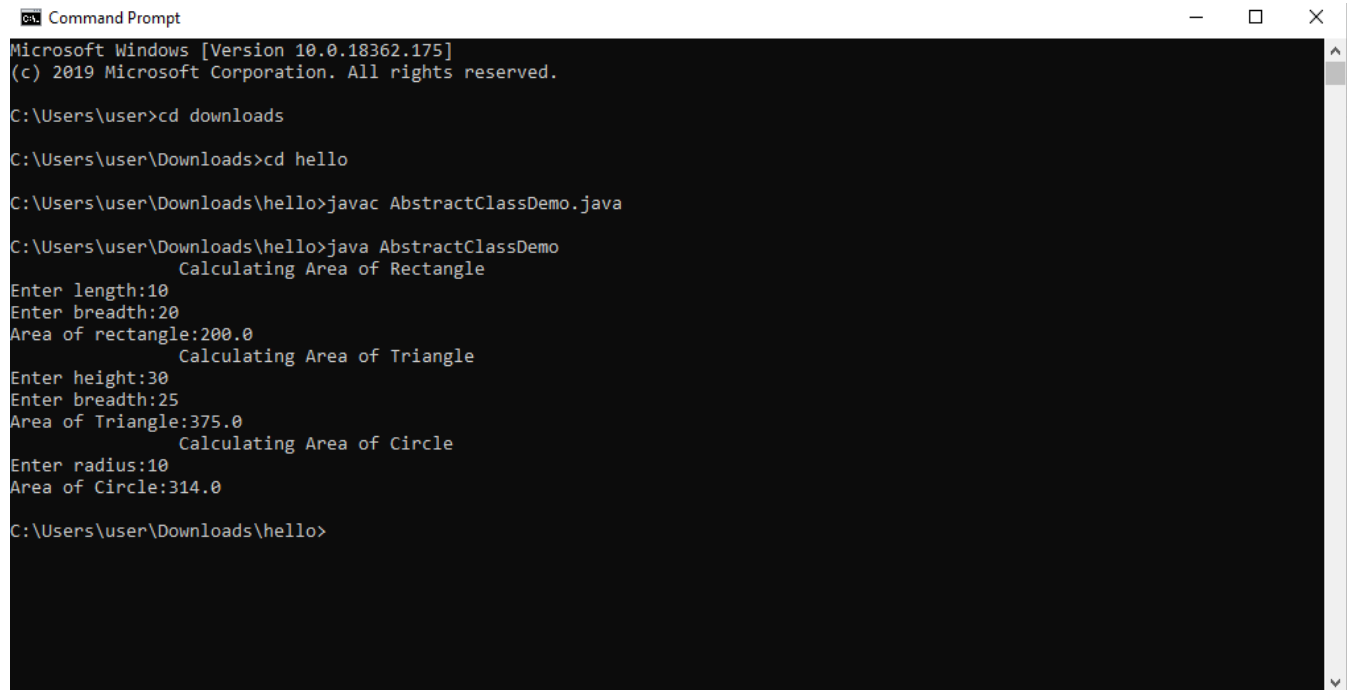
**Program:**

```
import java.util.*;
abstract class shape
{
    int a,b;
    abstract public void printarea();
}
class rectangle extends shape
{
    public int area_rect;
    public void printarea()
    {
        Scanner s=new Scanner(System.in);
        System.out.println("enter the length and breadth of rectangle");
        a=s.nextInt();
        b=s.nextInt();
        area_rect=a*b;
        System.out.println("Length of rectangle "+a +"breadth of rectangle "+b);
        System.out.println("The area ofrectangle is:"+area_rect);
    }
}
class triangle extends shape
{
    double area_tri;
    public void printarea()
    {
```



```
Scanner s=new Scanner(System.in);
System.out.println("enter the base and height of triangle");
a=s.nextInt();
b=s.nextInt();
System.out.println("Base of triangle "+a +"height of triangle "+b);
area_tri=(0.5*a*b);
System.out.println("The area of triangle is:"+area_tri);
}
}
class circle extends shape
{
double area_circle;
public void printarea()
{
Scanner s=new Scanner(System.in);
System.out.println("enter the radius of circle");
a=s.nextInt();
area_circle=(3.14*a*a);
System.out.println("Radius of circle"+a);
System.out.println("The area of circle is:"+area_circle);
}
}
public class shapeclass
{
public static void main(String[] args)
{
rectangle r=new rectangle();
r.printarea();
triangle t=new triangle();
t.printarea();
circle r1=new circle();
r1.printarea();
}
}
```

## Output:



```
Microsoft Windows [Version 10.0.18362.175]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\user>cd downloads
C:\Users\user\Downloads>cd hello
C:\Users\user\Downloads\hello>javac AbstractClassDemo.java
C:\Users\user\Downloads\hello>java AbstractClassDemo
    Calculating Area of Rectangle
Enter length:10
Enter breadth:20
Area of rectangle:200.0
    Calculating Area of Triangle
Enter height:30
Enter breadth:25
Area of Triangle:375.0
    Calculating Area of Circle
Enter radius:10
Area of Circle:314.0
C:\Users\user\Downloads\hello>
```

## Result:

EXP. NO: 5	Calculate area using an interface
DATE:	

**Aim:**

**Algorithms:**

Step 1: Define MyInterface with method printArea.

Step 2: Create abstract class Shapes implements MyInterface with double attributes a, b.

Step 3: Create class Rectangle extending Shapes:

- a. Override printArea:
  - Read length and breadth from user.
  - Print area = length \* breadth.

Step 4: Create class Circle extending Shapes:

- a. Override printArea:
  - Read radius from user.
  - Print area = 3.14 \* radius \* radius.

Step 5: In main:

- a. Create MyInterface object.
- b. Set it to a new Rectangle, call printArea
- c. Set it to a new Circle, call printArea.

**Program:**

```
import java.util.Scanner;
interface MyInterface {
    void printArea();
}
abstract class Shapes {
    double a, b;
    abstract void printArea();
}

class Rectangle extends Shapes implements MyInterface {
    public void printArea() {
        System.out.println("Calculating area of Rectangle ");
        Scanner input = new Scanner(System.in);
        System.out.println("Enter length:");
        a = input.nextDouble();
        System.out.println("Enter breadth:");
        b = input.nextDouble();
        double area = a * b;
        System.out.print("Area of Rectangle: " + area);
    }
}

class Circle extends Shapes implements MyInterface {
    public void printArea() {
        System.out.print("Calculating area of Circle ");
    }
}
```

```

Scanner input = new Scanner(System.in);
System.out.println("Enter radius:");
a = input.nextDouble();

double area = 3.14 * a * a;
System.out.print("Area of Circle: " + area);
}
}
class Triangle extends Shapes implements MyInterface {
    public void printArea() {
        System.out.print("Calculating area of Triangle ");
        Scanner input = new Scanner(System.in);
        System.out.println("Enter base:");
        a = input.nextDouble();
        System.out.println("Enter height:");
        b = input.nextDouble();
        double area = 0.5 * a * b;
        System.out.println("Area of Triangle: " + area);
    }
}
class AbstractClassDemo {
    public static void main(String[] args) {
        MyInterface obj;
        obj = new Rectangle();
        obj.printArea();
        obj = new Triangle();
        obj.printArea();
        obj = new Circle();
        obj.printArea();
    }
}

```

### **Output:**

```

Calculating area of Rectangle
Enter length: 10
Enter breadth: 20
Area of Rectangle: 200.0
Calculating area of Triangle
Enter base:10
Enter height:5
Area of Triangle: 25.0
Calculating area of Circle
Enter radius:10
Area of Circle: 314.0

```

### **Result:**

<b>EXP. NO: 6</b>	<b>Implement exception handling and creation of user defined exceptions</b>
<b>DATE:</b>	

**Aim:**

**Algorithm:**

Step 1: Create a class which extends Exception class.

Step 2: Create a constructor which receives the string as argument.

Step 3: Get the Amount as input from the user.

Step 4: If the amount is negative, the exception will be generated.

Step 5: Using the exception handling mechanism, the thrown exception is handled by the catch construct.

Step 6: After the exception is handled, the string "invalid amount "will be displayed.

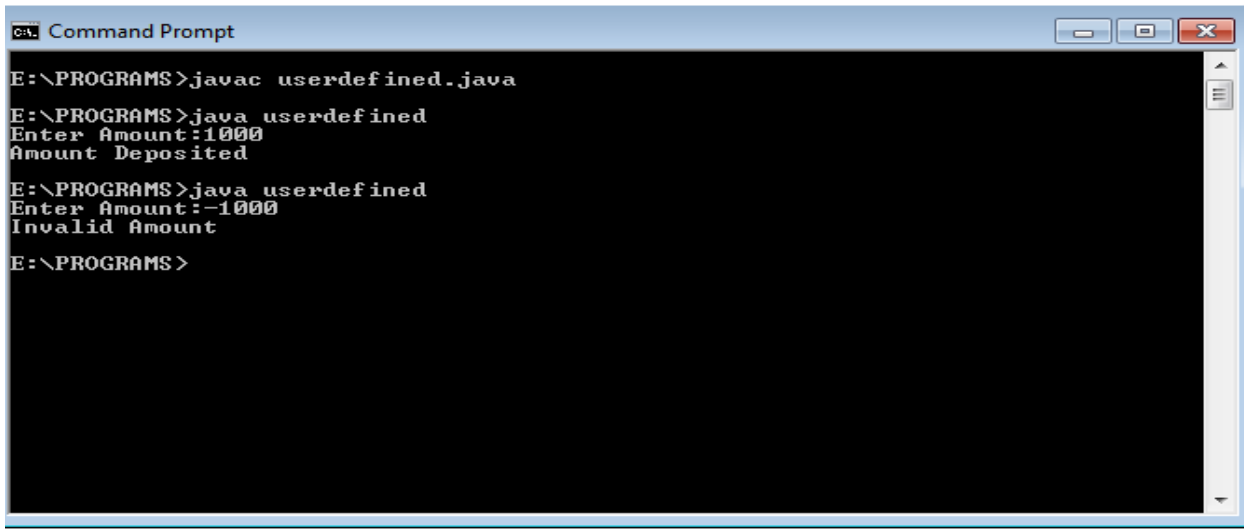
Step 7: If the amount is greater than 0, the message "Amount Deposited "will be displayed.

**(a) Program:**

```
import java.util.Scanner;
class NegativeAmtException extends Exception
{
    String msg;
    NegativeAmtException(String msg)
    {
        this.msg=msg;
    }
    public String toString()
    {
        return msg;
    }
}
public class userdefined
{
    public static void main(String[] args)
    {
        Scanner s=new Scanner(System.in);
        System.out.print("Enter Amount:");
        int a=s.nextInt();
        try
        {
            if(a<0)
            {
                throw new NegativeAmtException("Invalid Amount");
            }
            System.out.println("Amount Deposited");
        }
    }
}
```

```
catch(NegativeAmtException e)
{
    System.out.println(e);
}
}
}
```

### **Output:**

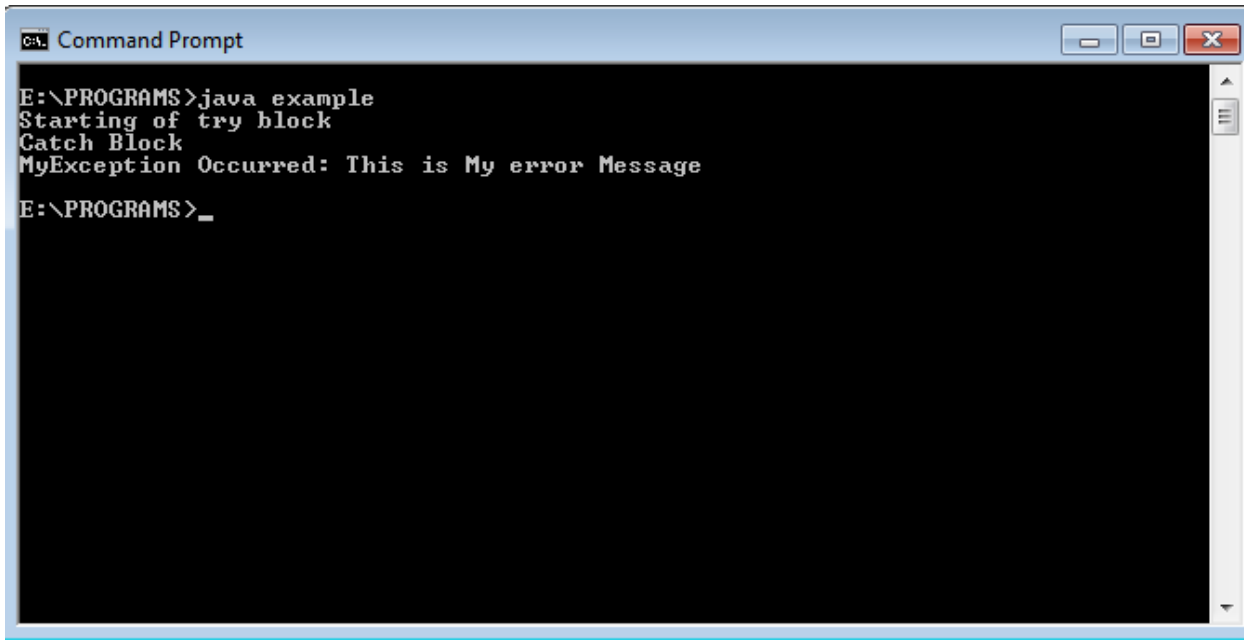


```
Command Prompt
E:\PROGRAMS>javac userdefined.java
E:\PROGRAMS>java userdefined
Enter Amount:1000
Amount Deposited
E:\PROGRAMS>java userdefined
Enter Amount:-1000
Invalid Amount
E:\PROGRAMS>
```

### **(b) PROGRAM**

```
class MyException extends Exception{
    String str1;
    MyException(String str2)
    {
        str1=str2;
    }
    public String toString()
    {
        return ("MyException Occurred: "+str1) ;
    }
}
class example
{
    public static void main(String args[])
    {
        try
        {
            System.out.println("Starting of try block");
            throw new MyException("This is My error Message");
        }
        catch(MyException exp)
        {
            System.out.println("Catch Block") ;
            System.out.println(exp) ;
        }
    }
}
```

## OUTPUT



```
c:\ Command Prompt

E:\PROGRAMS>java example
Starting of try block
Catch Block
MyException Occurred: This is My error Message
E:\PROGRAMS>_
```

### Program 2:

#### Try-catch Block - Java Program [Run ErrDemo.java]

```
Class RunErrDemo{
Public static void main(String[] args)
{
int a,b,c;
a=10;
b=0;
try{

c=a/b;
}
Catch(Arithmetic Exception e)
{
System.out.println("\n Divide by zero");
}
System.out.println("\n The value of a:"+a);
System.out.println("\n The value of b:"+b);
}}
```

### Output:

```
Divide by zero
The value of a: 10
The value of b: 0
```

**Program 3:****User defined Exception – Java Program MyExceptDemo.java**

```
import java.lang.Exception;
Class MyOwnException extends Exception
{
MyOwnException(String msg)
{
Super(msg);
}}
Class MyExceptDemo
{
public static void main(String args[])
{
int age;
age=15;
try{
if(age<21)
throw new MyOwnException("Your age is very less than the condition");
}
catch(MyOwnException e)
{
System.out.println("This is my Exception block");
System.out.println(e.getMessage());
}
finally
{
System.out.println("Finally block:End of the program");}}}
```

**Output:**

This is My Exception block  
Your age is very less than the condition  
Finally block:End of the program

**Result:**



<b>EXP. NO: 7</b>	<b>Program to implement Multithreaded application</b>
<b>DATE:</b>	

**Aim:**

**Algorithm:**

Step 1: Create a class even which implements first thread that computes the square of the number.

Step 2: run() method implements the code to be executed when thread gets executed.

Step 3: Create a class odd which implements second thread that computes the cube of the number.

Step 4: Create a third thread that generates random number. If the random number is even, it displays the square of the number. If the random number generated is odd, it displays the cube of the given number.

Step 5: The Multithreading is performed and the task switched between multiple threads.

Step 6: The sleep () method makes the thread to suspend for the specified time.

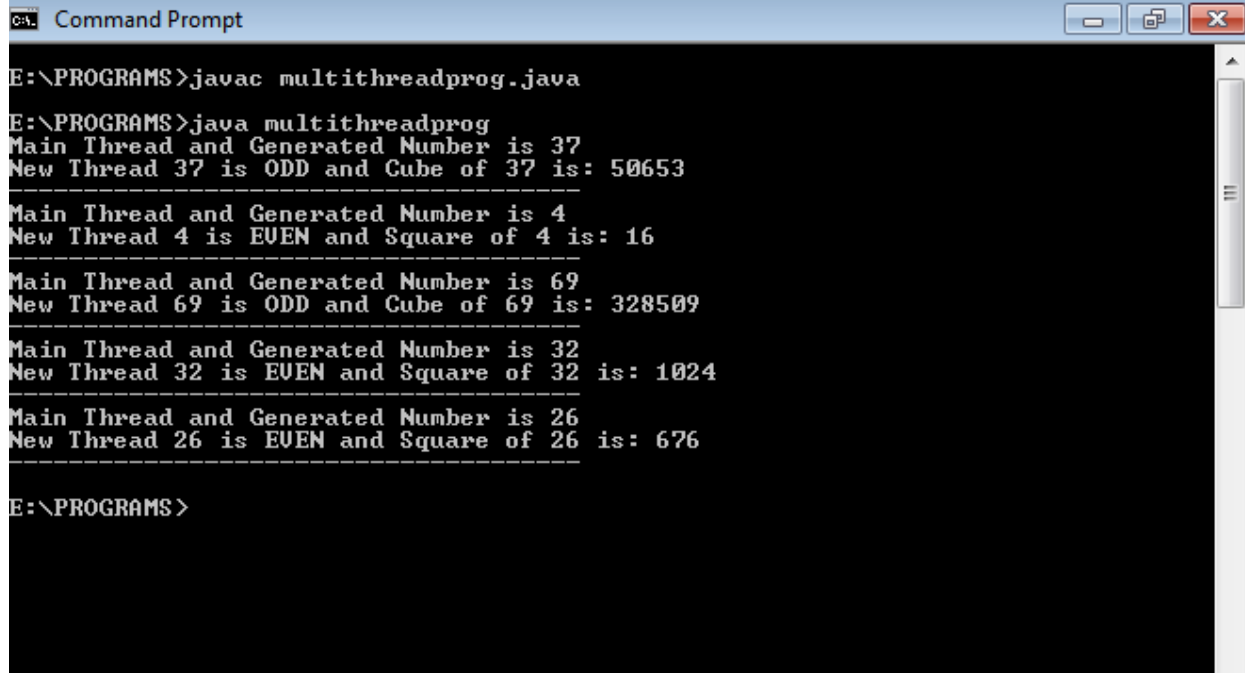
**Program:**

```
import java.util.*;
class even implements Runnable
{
    public int x;
    public even(int x)
    {
        this.x = x;
    }
    public void run()
    {
        System.out.println("New Thread "+ x +" is EVEN and Square of " + x + " is: " + x * x);
    }
}
class odd implements Runnable
{
    public int x;
    public odd(int x)
    {
        this.x = x;
    }
    public void run()
    {
        System.out.println("New Thread "+ x +" is ODD and Cube of " + x + " is: " + x * x * x);
    }
}
```

```
class A extends Thread
{
public void run()
{
int num = 0;
Random r = new Random();
try
{
for (int i = 0; i < 5; i++)
{
num = r.nextInt(100);
System.out.println("Main Thread and Generated Number is " + num);
if (num % 2 == 0)
{
Thread t1 = new Thread(new even(num));
t1.start();
}
else
{
Thread t2 = new Thread(new odd(num));
t2.start();
}
Thread.sleep(1000);
System.out.println("-----");
}
}
catch (Exception ex)
{
System.out.println(ex.getMessage());
}
}
}

public class multithreadprog
{
public static void main(String[] args)
{
A a = new A();
a.start();
}
}
```

## Output:



```
E:\PROGRAMS>javac multithreadprog.java
E:\PROGRAMS>java multithreadprog
Main Thread and Generated Number is 37
New Thread 37 is ODD and Cube of 37 is: 50653
-----
Main Thread and Generated Number is 4
New Thread 4 is EVEN and Square of 4 is: 16
-----
Main Thread and Generated Number is 69
New Thread 69 is ODD and Cube of 69 is: 328509
-----
Main Thread and Generated Number is 32
New Thread 32 is EVEN and Square of 32 is: 1024
-----
Main Thread and Generated Number is 26
New Thread 26 is EVEN and Square of 26 is: 676
-----
E:\PROGRAMS>
```

## Result:

<b>EXP. NO: 8</b>	<b>Program to perform File Operations</b>
<b>DATE:</b>	

**Aim:**

**Algorithm:**

Step 1: Create a class filedemo. Get the file name from the user.

Step 2: Use the file functions and display the information about the file.

Step 3: getName() displays the name of the file.

Step 4: getPath() displays the path name of the file.

Step 5: getParent () -This method returns the pathname string of this abstract pathname's parent, or null if this pathname does not name a parent directory.

Step 6: exists() – Checks whether the file exists or not.

Step 7: canRead()-This method is basically a check if the file can be read.

Step 8: canWrite()-verifies whether the application can write to the file.

Step 9: isDirectory() – displays whether it is a directory or not.

Step 10: isFile() – displays whether it is a file or not.

Step 11: lastmodified() – displays the last modified information.

Step 12: length()- displays the size of the file.

Step 13: delete() – deletes the file

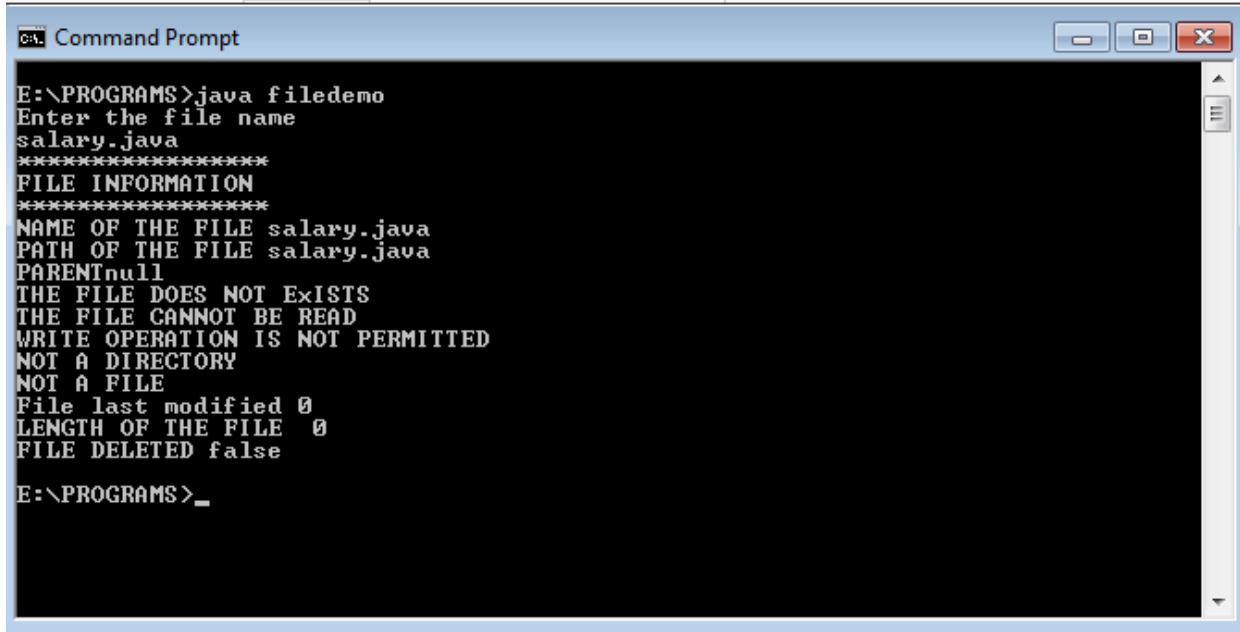
Step 14: Invoke the predefined functions and display the information about the file.

**Program:**

```
import java.io.*;
import java.util.*;
class filedemo
{
public static void main(String args[])
{
String filename;
Scanner s=new Scanner(System.in);
System.out.println("Enter the file name ");
filename=s.nextLine();
File f1=new File(filename);
```

```
System.out.println("*****");
System.out.println("FILE INFORMATION");
System.out.println("*****");
System.out.println("NAME OF THE FILE "+f1.getName());
System.out.println("PATH OF THE FILE "+f1.getPath());
System.out.println("PARENT"+f1.getParent());
if(f1.exists())
System.out.println("THE FILE EXISTS ");
else
System.out.println("THE FILE DOES NOT EXISTS ");
if(f1.canRead())
System.out.println("THE FILE CAN BE READ ");
else
System.out.println("THE FILE CANNOT BE READ ");
if(f1.canWrite())
System.out.println("WRITE OPERATION IS PERMITTED");
else
System.out.println("WRITE OPERATION IS NOT PERMITTED");
if(f1.isDirectory())
System.out.println("IT IS A DIRECTORY ");
else
System.out.println("NOT A DIRECTORY");
if(f1.isFile())
System.out.println("IT IS A FILE ");
else
System.out.println("NOT A FILE");
System.out.println("File last modified "+ f1.lastModified());
System.out.println("LENGTH OF THE FILE "+f1.length());
System.out.println("FILE DELETED "+f1.delete());
}
}
```

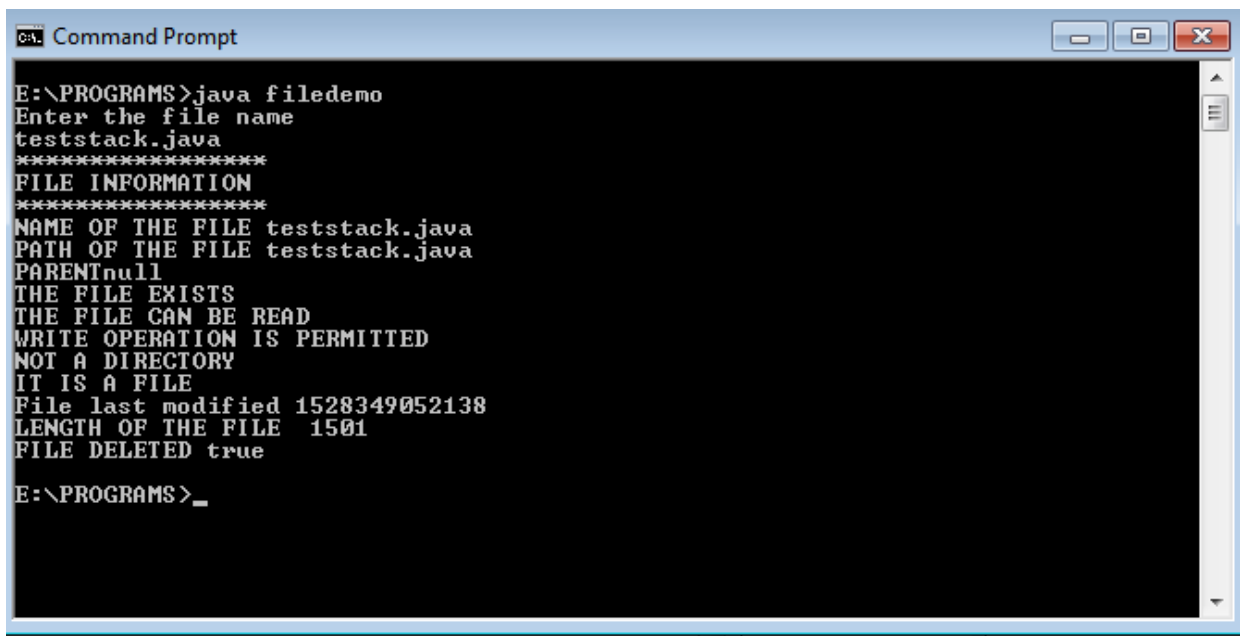
## Output:



```
CA. Command Prompt

E:\PROGRAMS>java filedemo
Enter the file name
salary.java
*****
FILE INFORMATION
*****
NAME OF THE FILE salary.java
PATH OF THE FILE salary.java
PARENTnull
THE FILE DOES NOT EXISTS
THE FILE CANNOT BE READ
WRITE OPERATION IS NOT PERMITTED
NOT A DIRECTORY
NOT A FILE
File last modified 0
LENGTH OF THE FILE 0
FILE DELETED false

E:\PROGRAMS>_
```



```
CA. Command Prompt

E:\PROGRAMS>java filedemo
Enter the file name
teststack.java
*****
FILE INFORMATION
*****
NAME OF THE FILE teststack.java
PATH OF THE FILE teststack.java
PARENTnull
THE FILE EXISTS
THE FILE CAN BE READ
WRITE OPERATION IS PERMITTED
NOT A DIRECTORY
IT IS A FILE
File last modified 1528349052138
LENGTH OF THE FILE 1501
FILE DELETED true

E:\PROGRAMS>_
```

## Result:

<b>EXP. NO: 9</b>	<b>Program to find the maximum value from the given type of elements using generics</b>
<b>DATE:</b>	

**Aim:**

**Algorithm:**

Step 1: Create a class MyClass to implement generic class and generic methods.

Step 2: Get the set of the values belonging to specific data type.

Step 3: Create the objects of the class to hold integer, character and double values.

Step 4: Create the method to compare the values and find the maximum value stored in the array.

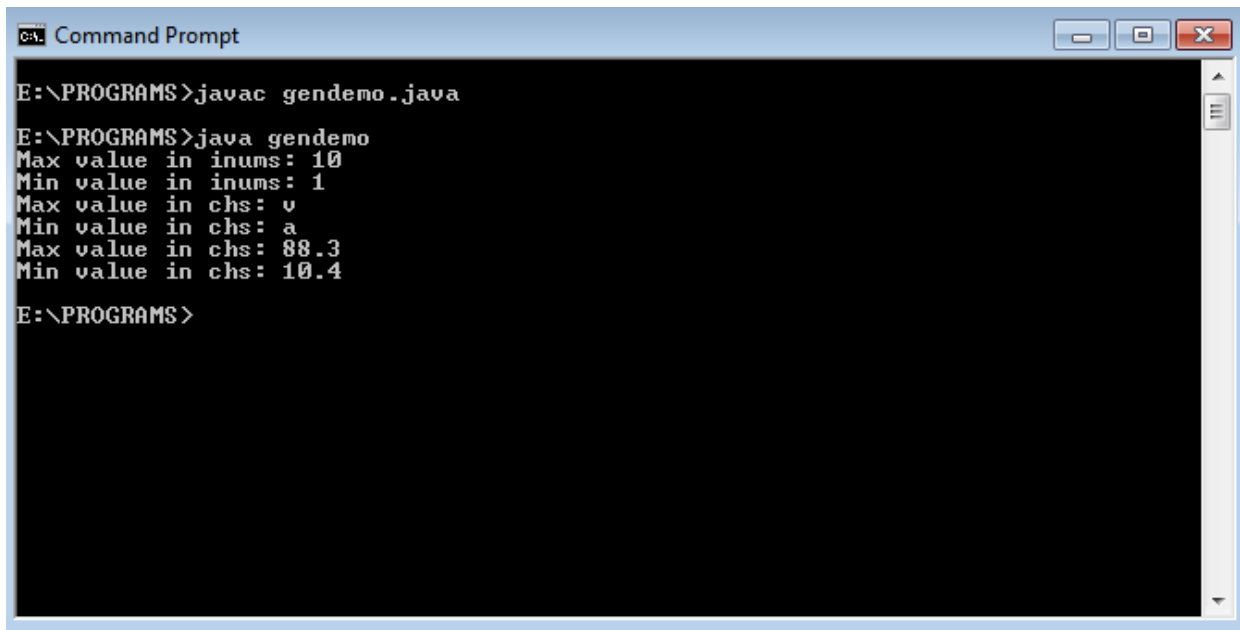
Step 5: Invoke the method with integer, character or double values. The output will be displayed based on the data type passed to the method.

**Program:**

```
class MyClass<T extends Comparable<T>>
{
    T[] vals;
    MyClass(T[] o)
    {
        vals = o;
    }
    public T min()
    {
        T v = vals[0];
        for(int i=1; i < vals.length; i++)
            if(vals[i].compareTo(v) < 0)
                v = vals[i];
        return v;
    }
    public T max()
    {
        T v = vals[0];
        for(int i=1; i < vals.length; i++)
            if(vals[i].compareTo(v) > 0)
                v = vals[i];
        return v;
    }
}
```

```
class gendemo
{
public static void main(String args[])
{
int i;
Integer inums[]={ 10,2,5,4,6,1};
Character chs[]={ 'v','p','s','a','n','h'};
Double d[]={ 20.2,45.4,71.6,88.3,54.6,10.4};
MyClass<Integer> iob = new MyClass<Integer>(inums);
MyClass<Character> cob = new MyClass<Character>(chs);
MyClass<Double>dob = new MyClass<Double>(d);
System.out.println("Max value in inums: " + iob.max());
System.out.println("Min value in inums: " + iob.min());
System.out.println("Max value in chs: " + cob.max());
System.out.println("Min value in chs: " + cob.min());
System.out.println("Max value in chs: " + dob.max());
System.out.println("Min value in chs: " + dob.min());
}
}
```

### **Output:**



```
CA. Command Prompt

E:\PROGRAMS>javac gendemo.java

E:\PROGRAMS>java gendemo
Max value in inums: 10
Min value in inums: 1
Max value in chs: v
Min value in chs: a
Max value in chs: 88.3
Min value in chs: 10.4

E:\PROGRAMS>
```

### **Result:**



<b>EXP. NO: 10</b>	<b>Program to design a calculator using event driven programming paradigm of java</b>
<b>DATE:</b>	

**Aim:**

**Algorithm:**

Step 1: import the swing packages and awt packages.

Step 2: Create the class scientificcalculator that implements action listener.

Step 3: Create the container and add controls for digits, scientific calculations and decimal Manipulations.

Step 4: The different layouts can be used to lay the controls.

Step 5: When the user presses the control, the event is generated and handled .

Step 6: The corresponding decimal , numeric and scientific calculations are performed.

**Program:**

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import javax.swing.event.*;
public class ScientificCalculator extends JFrame implements ActionListener
{
    JTextField tfield;
    double temp, temp1, result, a;
    static double m1, m2;
    int k = 1, x = 0, y = 0, z = 0;
    char ch;
    JButton b1, b2, b3, b4, b5, b6, b7, b8, b9, zero, clr, pow2, pow3, exp,
    fac, plus, min, div, log, rec, mul, eq, addSub, dot, mr, mc, mp,
    mm, sqrt, sin, cos, tan;
    Container cont;
    JPanel textPanel, buttonpanel;
    ScientificCalculator()
    {
        cont = getContentPane();
        cont.setLayout(new BorderLayout());
        JPanel textpanel = new JPanel();
        tfield = new JTextField(25);
        tfield.setHorizontalAlignment(SwingConstants.RIGHT);
        tfield.addKeyListener(new KeyAdapter() {
```

```
public void keyTyped(KeyEvent keyevent) {
    char c = keyevent.getKeyChar();
    if (c >= '0' && c <= '9') {
    }
    else
    {
        keyevent.consume();
    }
}

});
textpanel.add(tfield);
buttonpanel = new JPanel();
buttonpanel.setLayout(new GridLayout(8, 4, 2, 2));
boolean t = true;
mr = new JButton("MR");
buttonpanel.add(mr);
mr.addActionListener(this);
mc = new JButton("MC");
buttonpanel.add(mc);
mc.addActionListener(this);
mp = new JButton("M+");
buttonpanel.add(mp);
mp.addActionListener(this);
mm = new JButton("M-");
buttonpanel.add(mm);
mm.addActionListener(this);
b1 = new JButton("1");
buttonpanel.add(b1);
b1.addActionListener(this);
b2 = new JButton("2");
buttonpanel.add(b2);
b2.addActionListener(this);
b3 = new JButton("3");
buttonpanel.add(b3);
b3.addActionListener(this);
b4 = new JButton("4");
buttonpanel.add(b4);
b4.addActionListener(this);
b5 = new JButton("5");
buttonpanel.add(b5);
b5.addActionListener(this);
b6 = new JButton("6");
buttonpanel.add(b6);
b6.addActionListener(this);
b7 = new JButton("7");
buttonpanel.add(b7);
b7.addActionListener(this);
b8 = new JButton("8");
buttonpanel.add(b8);
b8.addActionListener(this);
```

```
b9 = new JButton("9");
buttonpanel.add(b9);
b9.addActionListener(this);
zero = new JButton("0");
buttonpanel.add(zero);
zero.addActionListener(this);
plus = new JButton("+");
buttonpanel.add(plus);
plus.addActionListener(this);
min = new JButton("-");
buttonpanel.add(min);
min.addActionListener(this);
mul = new JButton("*");
buttonpanel.add(mul);
mul.addActionListener(this);
div = new JButton("/");
div.addActionListener(this);
buttonpanel.add(div);
addSub = new JButton("+/-");
buttonpanel.add(addSub);
addSub.addActionListener(this);
dot = new JButton(".");
buttonpanel.add(dot);
dot.addActionListener(this);
eq = new JButton("=");
buttonpanel.add(eq);
eq.addActionListener(this);
rec = new JButton("1/x");
buttonpanel.add(rec);
rec.addActionListener(this);
sqrt = new JButton("Sqrt");
buttonpanel.add(sqrt);
sqrt.addActionListener(this);
log = new JButton("log");
buttonpanel.add(log);
log.addActionListener(this);
sin = new JButton("SIN");
buttonpanel.add(sin);
sin.addActionListener(this);
cos = new JButton("COS");
buttonpanel.add(cos);
cos.addActionListener(this);
tan = new JButton("TAN");
buttonpanel.add(tan);
tan.addActionListener(this);
pow2 = new JButton("x^2");
buttonpanel.add(pow2);
pow2.addActionListener(this);
pow3 = new JButton("x^3");
buttonpanel.add(pow3);
```

```
pow3.addActionListener(this);
exp = new JButton("Exp");
exp.addActionListener(this);
buttonpanel.add(exp);
fac = new JButton("n!");
fac.addActionListener(this);
buttonpanel.add(fac);
clr = new JButton("AC");
buttonpanel.add(clr);
clr.addActionListener(this);
cont.add("Center", buttonpanel);
cont.add("North", textpanel);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
public void actionPerformed(ActionEvent e)
{
String s = e.getActionCommand();
if (s.equals("1"))
{
if (z == 0)
{
tfield.setText(tfield.getText() + "1");
}
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "1");
z = 0;
}
}
if (s.equals("2")) {
if (z == 0) {
tfield.setText(tfield.getText() + "2");
}
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "2");
z = 0;
}
}
if (s.equals("3")) {
if (z == 0) {
tfield.setText(tfield.getText() + "3");
}
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "3");
z = 0;
}
```

```
}  
}  
if (s.equals("4")) {  
    if (z == 0) {  
        tfield.setText(tfield.getText() + "4");  
    }  
    else  
    {  
        tfield.setText("");  
        tfield.setText(tfield.getText() + "4");  
        z = 0;  
    }  
}  
if (s.equals("5")) {  
    if (z == 0) {  
        tfield.setText(tfield.getText() + "5");  
    }  
    else  
    {  
        tfield.setText("");  
        tfield.setText(tfield.getText() + "5");  
        z = 0;  
    }  
}  
if (s.equals("6")) {  
    if (z == 0) {  
        tfield.setText(tfield.getText() + "6");  
    }  
    else  
    {  
        tfield.setText("");  
        tfield.setText(tfield.getText() + "6");  
        z = 0;  
    }  
}  
if (s.equals("7")) {  
    if (z == 0) {  
        tfield.setText(tfield.getText() + "7");  
    }  
    else  
    {  
        tfield.setText("");  
        tfield.setText(tfield.getText() + "7");  
        z = 0;  
    }  
}  
if (s.equals("8")) {  
    if (z == 0) {  
        tfield.setText(tfield.getText() + "8");  
    }  
}
```

```
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "8");
z = 0;
}
}
if (s.equals("9")) {
if (z == 0) {
tfield.setText(tfield.getText() + "9");
}
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "9");
z = 0;
}
}
if (s.equals("0"))
{
if (z == 0) {
tfield.setText(tfield.getText() + "0");
}
else
{
tfield.setText("");
tfield.setText(tfield.getText() + "0");
z = 0;
}
}
if (s.equals("AC")) {
tfield.setText("");
x = 0;
y = 0;
z = 0;
}
if (s.equals("log"))
{
if (tfield.getText().equals("")) {
tfield.setText("");
}
else
{
a = Math.log(Double.parseDouble(tfield.getText()));
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("1/x")) {
if (tfield.getText().equals("")) {
```

```

tfield.setText("");
}
else
{
a = 1 / Double.parseDouble(tfield.getText());
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("Exp")) {
if (tfield.getText().equals("")) {
tfield.setText("");
}
else
{
a = Math.exp(Double.parseDouble(tfield.getText()));
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("x^2")) {
if (tfield.getText().equals("")) {
tfield.setText("");
}
else
{
a = Math.pow(Double.parseDouble(tfield.getText()), 2);
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("x^3")) {
if (tfield.getText().equals("")) {
tfield.setText("");
}
else
{
a = Math.pow(Double.parseDouble(tfield.getText()), 3);
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("+/-")) {
if (x == 0) {
tfield.setText("-" + tfield.getText());
x = 1;
}
else
{
tfield.setText(tfield.getText());
}
}

```

```
}  
}  
if (s.equals(".")) {  
    if (y == 0) {  
        tfield.setText(tfield.getText() + ".");  
        y = 1;  
    }  
    else  
    {  
        tfield.setText(tfield.getText());  
    }  
}  
if (s.equals("+"))  
{  
    if (tfield.getText().equals(""))  
    {  
        tfield.setText("");  
        temp = 0;  
        ch = '+';  
    }  
    else  
    {  
        temp = Double.parseDouble(tfield.getText());  
        tfield.setText("");  
        ch = '+';  
        y = 0;  
        x = 0;  
    }  
    tfield.requestFocus();  
}  
if (s.equals("-"))  
{  
    if (tfield.getText().equals(""))  
    {  
        tfield.setText("");  
        temp = 0;  
        ch = '-';  
    }  
    else  
    {  
        x = 0;  
        y = 0;  
        temp = Double.parseDouble(tfield.getText());  
        tfield.setText("");  
        ch = '-';  
    }  
    tfield.requestFocus();  
}  
if (s.equals("/")) {
```



```
if (tfield.getText().equals(""))
{
tfield.setText("");
temp = 1;
ch = '/';
}
else
{
x = 0;
y = 0;
temp = Double.parseDouble(tfield.getText());
ch = '/';
tfield.setText("");
}
tfield.requestFocus();
}
if (s.equals("*")) {
if (tfield.getText().equals(""))
{
tfield.setText("");
temp = 1;
ch = '*';
}
else
{
x = 0;
y = 0;
temp = Double.parseDouble(tfield.getText());
ch = '*';
tfield.setText("");
}
tfield.requestFocus();
}
if (s.equals("MC"))
{
m1 = 0;
tfield.setText("");
}
if (s.equals("MR"))
{
tfield.setText("");
tfield.setText(tfield.getText() + m1);
}
if (s.equals("M+"))
{
if (k == 1) {
m1 = Double.parseDouble(tfield.getText());
k++;
}
}
```

```
else
{
m1 += Double.parseDouble(tfield.getText());
tfield.setText("" + m1);
}
}
if (s.equals("M-"))
{
if (k == 1) {
m1 = Double.parseDouble(tfield.getText());
k++;
}
else
{
m1 -= Double.parseDouble(tfield.getText());
tfield.setText("" + m1);
}
}
if (s.equals("Sqrt"))
{
if (tfield.getText().equals(""))
{
tfield.setText("");
}
else
{
a = Math.sqrt(Double.parseDouble(tfield.getText()));
tfield.setText("");
field.setText(tfield.getText() + a);
}
}
if (s.equals("SIN"))
{
if (tfield.getText().equals(""))
{
tfield.setText("");
}
else
{
a = Math.sin(Double.parseDouble(tfield.getText()));
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("COS"))
{
if (tfield.getText().equals(""))
{
tfield.setText("");
}
}
```

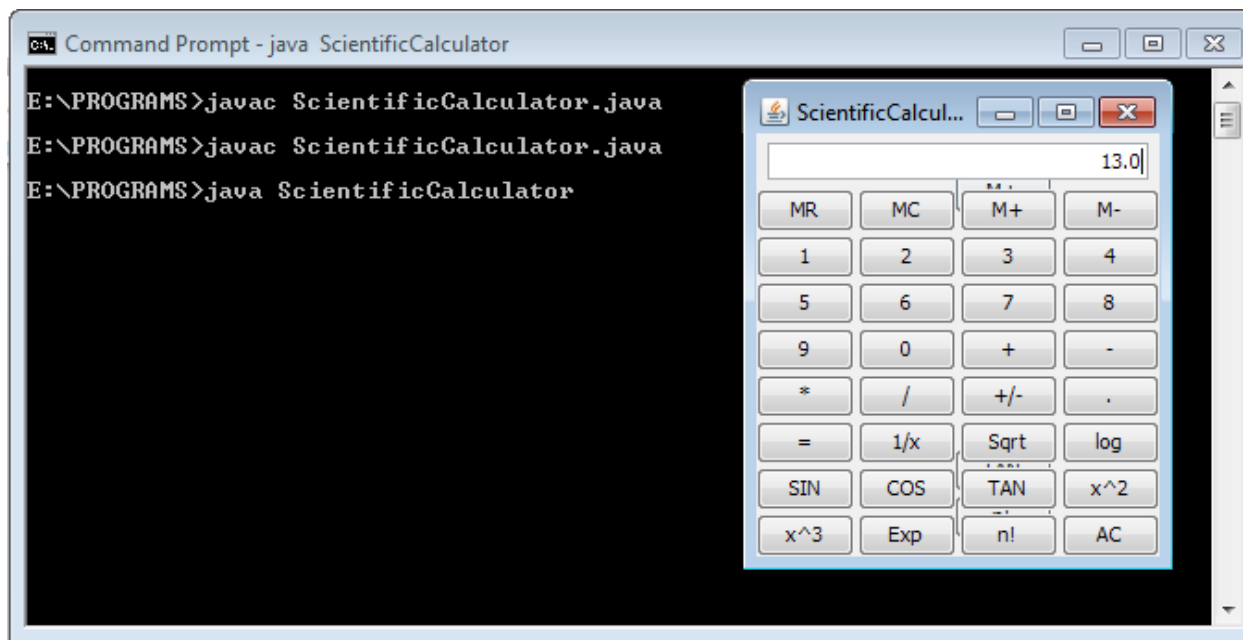
```

else
{
a = Math.cos(Double.parseDouble(tfield.getText()));
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("TAN")) {
if (tfield.getText().equals("")) {
tfield.setText("");
}
else
{
a = Math.tan(Double.parseDouble(tfield.getText()));
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
if (s.equals("="))
{
if (tfield.getText().equals(""))
{
tfield.setText("");
}
else
{
temp1 = Double.parseDouble(tfield.getText());
switch (ch)
{
case '+':
result = temp + temp1;
break;
case '-':
result = temp - temp1;
break;
case '/':
result = temp / temp1;
break;
case '*':
result = temp * temp1;
break;
}
tfield.setText("");
tfield.setText(tfield.getText() + result);
z = 1;
}
}
if (s.equals("n!"))
{

```

```
if (tfield.getText().equals(""))
{
tfield.setText("");
}
else
{
a = fact(Double.parseDouble(tfield.getText()));
tfield.setText("");
tfield.setText(tfield.getText() + a);
}
}
tfield.requestFocus();
}
double fact(double x)
{
int er = 0;
if (x < 0)
{
er = 20;
return 0;
}
double i, s = 1;
for (i = 2; i <= x; i += 1.0)
s *= i;
return s;
}
public static void main(String args[])
{
try
{
UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
}
catch (Exception e)
{
}
ScientificCalculator f = new ScientificCalculator();
f.setTitle("ScientificCalculator");
f.pack();
f.setVisible(true);
}
}
```

## Output:



## Result:

<b>EXP. NO: 11</b>	<b>Program to develop a Java application to generate Electricity bill</b>
<b>DATE:</b>	

**Aim:**

**Algorithm:**

Step 1: Create a class with the following members

Consumer no., consumer name, previous month reading, current month reading, type of EB connection (i.e domestic or commercial)

Step 2: Compute the bill amount using the following tariff.

If the type of the EB connection is domestic, calculate the amount to be paid as follows:

- First 100 units - Rs. 1 per unit
- 101-200 units - Rs. 2.50 per unit
- 201 -500 units - Rs. 4 per unit
- > 501 units - Rs. 6 per unit

If the type of the EB connection is commercial, calculate the amount to be paid as follows:

- First 100 units - Rs. 2 per unit
- 101-200 units - Rs. 4.50 per unit
- 201 -500 units - Rs. 6 per unit
- > 501 units - Rs. 7 per unit

Step 3: Create the object for the created class.

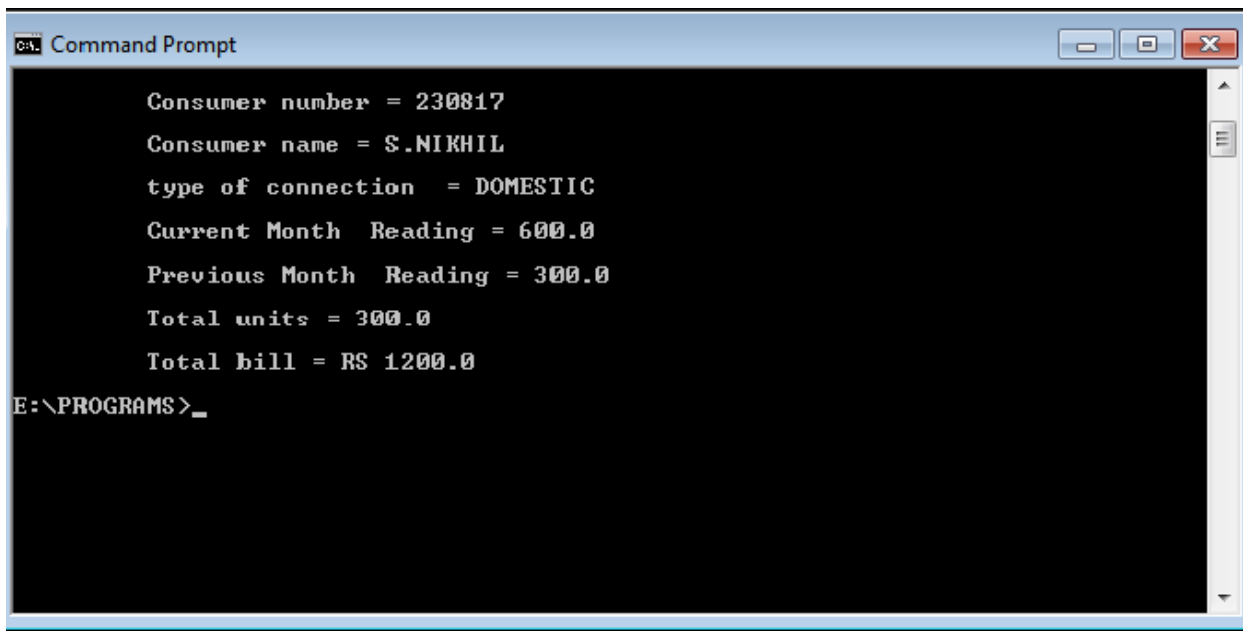
Invoke the methods to get the input from the consumer and display the consumer information with the generated electricity bill.

**Program:**

```
import java.util.*;
class ebill
{
public static void main (String args[])
{
customerdata ob = new customerdata();
ob.getdata();
ob.calc();
ob.display();
}
}
class customerdata
{
Scanner in = new Scanner(System.in);
Scanner ins = new Scanner(System.in);
String cname,type;
int bn;
double current,previous,tbill,units;
void getdata()
{
System.out.print ("\n\t Enter consumer number ");
bn = in.nextInt();
System.out.print ("\n\t Enter Type of connection (D for Domestic or C for Commercial) ");
type = ins.nextLine();
System.out.print ("\n\t Enter consumer name ");
cname = ins.nextLine();
System.out.print ("\n\t Enter previous month reading ");
previous= in.nextDouble();
System.out.print ("\n\t Enter current month reading ");
current= in.nextDouble();
}
void calc()
{
units=current-previous;
if(type.equals("D"))
{
if (units<=100)
tbill=1 * units;
else if (units>100 && units<=200)
tbill=2.50*units;
else if(units>200 && units<=500)
tbill= 4*units;
else
tbill= 6*units;
}
else
{
if (units<=100)
tbill= 2 * units;
```

```
else if(units>100 && units<=200)
tbill=4.50*units;
else if(units>200 && units<=500)
tbill= 6*units;
else
tbill= 7*units;
}}
void display()
{
System.out.println("\n\t Consumer number = "+bn);
System.out.println ("\n\t Consumer name = "+cname);
if(type.equals("D"))
System.out.println ("\n\t type of connection = DOMESTIC ");
else
System.out.println ("\n\t type of connection = COMMERCIAL ");
System.out.println ("\n\t Current Month  Reading = "+current);
System.out.println ("\n\t Previous Month  Reading = "+previous);
System.out.println ("\n\t Total units = "+units);
System.out.println ("\n\t Total bill = RS "+tbill);
}}
```

### **Output:**



```
Command Prompt

Consumer number = 230817
Consumer name = S.NIKHIL
type of connection = DOMESTIC
Current Month  Reading = 600.0
Previous Month  Reading = 300.0
Total units = 300.0
Total bill = RS 1200.0

E:\PROGRAMS>_
```

### **Result:**