Ex.No:1a **Date:**	**Solve Problems by using Sequential search**

AIM:

To write a java program to solve problems by using Sequential search.

ALGORITHM:

```
Step 1: Set i to 1.
Step 2: if I > n, then go to step 7
Step 3: if Ar[i] = x then go to step 6
Step 4: Set I to I + 1
Step 5: Go to Step 2
Step 6: Print Element x found at index I and go to step 8
Step 7: print element not found
Step 8: Exit
```

PROGRAM:

```
import  java.util.Scanner;
public class SequentialSearch {
        public static void main(String[] args) {
                int i , j, k, l = 0;
                Scanner s = new Scanner(System.in);
                System.out.println("Enter the size of array: ");
                i = s.nextInt();
                int[] ar = new int[i];
                for(j = 0; j<i ; j++){
                        System.out.println("No "+ (j+1) + " = ");
                ar[j] = s.nextInt();
                }
                System.out.println("Total Inserted list is : ");
                for(j = 0; j<i ; j++){
                        System.out.println("No "+ (j+1)+"="+ ar[j]);
                }
        System.out.println("Enter the number which needs to find: ");
```

```
                k =  s.nextInt();
                for(j = 0; j<i ; j++){
                        if(k == ar[j]){
                                System.out.println("Found"+(j+1));
                                l = l+1;
                                break;
                        }
                }
                if(l==0){
                        System.out.println("Data Not Found in the list");
                }
        }
}
```

OUTPUT:

```
Enter the size of array: 5
No 1 = 34
No 2 = 36
No 3 = 12
No 4 = 45
No 5 = 26
Total Inserted list is :
No 1=34
No 2=36
No 3=12
No 4=45
No 5=26
Enter the number which needs to find:
12
Found 3
```

RESULT:

Thus the java program for solve problem by using sequential search was executed and ouput is verified successfully

Ex.No:1b **Date:**	**Solve Problems by using Binary search**

AIM:

To write a java program for solve problems by using Binary search.

ALGORITHM:

Step 1: Created one function named binarySearch having the parameters of an array variable and the target element variable.

Step 2: Initialized the start variable with value 0 and end variable with the value of arr.length -1 which means the last element of the array. Hence, here take the starting and ending value of the sorted array.

Step 3: Start a while loop with the condition that start value should always be less than or equal to the end value.

Step 4: Then find the middle element with the given formula.

Step 5: Now check the Binary Search cases which was also explained in the algorithm of it and if none of the value matches then simply return -1.

Step 6: In the Driver Code, taken one sorted array and defined the target element whose index value needs to be find out and then calls the function.

Step 7: Take four variables => start, end, middle, target.

Step 8: Find the middle element of an array => now your array is divided into two parts left and right.

Step 9: If **target element > middle element** => search in **right** part.

Step 10: If **target element < middle element** => search in **left** part.

Step 11: If **middle element == target element** => return that element

Step 12: If **start > end** => element not found.

PROGRAM:

```
class BinarySearch {
    int binarySearch(int arr[], int l, int r, int x)
    {
        if (r >= l) {
            int mid = l + (r - l) / 2;
            if (arr[mid] == x)
                return mid;
            if (arr[mid] > x)
                return binarySearch(arr, l, mid - 1, x);
            return binarySearch(arr, mid + 1, r, x);
        }
        return -1;
    }
    public static void main(String args[])
    {
        BinarySearch ob = new BinarySearch();
        int arr[] = { 2, 3, 4, 10, 40 };
        int n = arr.length;
        int x = 10;
        int result = ob.binarySearch(arr, 0, n - 1, x);
        if (result == -1)
            System.out.println("Element not present");
        else
            System.out.println("Element found at index " + result);
    }
}
```

OUTPUT:

```
Element found at index 3
```

RESULT:

Thus the java program for solve problem by using binary search was executed and output is verified successfully.

Ex.No:1c **Date:**	**Solve Problems by using Quadratic sorting**

AIM:

To write a java program for solve problems by using quadratic sorting.

ALGORITHM:

Step 1: Run a nested for loop to traverse the input array using two variables **i** and **j**, such that 0 ≤ i < n-1 and 0 ≤ j < n-i-1

Step 2: If **arr[j]** is greater than **arr[j+1]** then swap these adjacent elements, else move on

Step 3: Print the sorted array

PROGRAM:

```
class BubbleSort {
    void bubbleSort(int arr[])
    {
        int n = arr.length;
        for (int i = 0; i < n - 1; i++)
            for (int j = 0; j < n - i - 1; j++)
                if (arr[j] > arr[j + 1]) {
                    // swap arr[j+1] and arr[j]
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
    }

    /* Prints the array */
    void printArray(int arr[])
    {
        int n = arr.length;
        for (int i = 0; i < n; ++i)
            System.out.print(arr[i] + " ");
        System.out.println();
    }

    // Driver method to test above
    public static void main(String args[])
    {
        BubbleSort ob = new BubbleSort();
        int arr[] = { 64, 34, 25, 12, 22, 11, 90 };
        ob.bubbleSort(arr);
        System.out.println("Sorted array");
        ob.printArray(arr);
    }
}
```

OUTPUT:

```
Sorted array:
1 2 4 5 8
```

RESULT:

Thus the java program for solve problems by using quardratic sorting was executed and output is verified successfully.

Ex.No:2a Date:	**Develop Stack data structures using classes and object**

AIM:

To write a java program to develop stack data structures using classes and object.

ALGORITHM:

Step 1: push inserts an item at the top of the stack (i.e., above its current top element).

Step 2: pop removes the object at the top of the stack and returns that object from the function. The stack size will be decremented by one.

Step 3: isEmpty tests if the stack is empty or not.

Step 4: isFull tests if the stack is full or not.

Step 5: peek returns the object at the top of the stack without removing it from the stack or modifying the stack in any way.

Step 6: size returns the total number of elements present in the stack.

PROGRAM:

```
public class StackDemo
{
int  size;

int stack[];

int top:

// constructor having size as parameter

StackDemo(int size){

this.size = size;

this.stack = new int[size];

this.top = -1;

} // this method pushes an element on stack

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...");

}
}
// this method deletes an element from stack

public int pop(){

if (lisEmpty())

{

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;

}

}
```

```
// this method returns topmost element from stack

public int peek()

{

if(!this.isEmpty())
return stack[top]:

else

{

System.out.println("Stack is Empty"); return -1;

}

}

// this method checks stack is empty
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 Operation------------ ");

obj.pop():

obj.pop();
obj.pop();

obj.pop();

}

}
```

OUTPUT:

```
-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...
```

RESULT:

Thus the java program for develop stack data structures using classes and object was executed and output is verified successfully.

Ex.No:2b **Date:**	**Develop Queue data structures using classes and object**

AIM:

To write a java program to develop queue data structures using classes and object.

ALGORITHM:

Step 1: insert an element into the queue and check if the queue is full

Step 2: insert element at the rear

Step 3: remove an element from the queue and check if queue is empty

Step 4: shift elements to the right by one place uptil rear and put set queue[rear] to 0

Step 5: print queue elements

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)

{

//check if queue is full
```

```
if(isQueueFull())

{

System.out.println("Queue is full!");

return;

}


// increment rear then insert item

quel++rear] = item;

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

}

public int remove()

{

//check if queue is empty

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:

```
------------------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
```

RESULT:

Thus the java program for to develop queue data structures using classess and objects was executed and the output is verified successfully.

Ex. No:3 **Date:**	**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 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. General pay slips for the employees with their gross and net salary.**

AIM:

To write a java program to develop a java application with an employee class with Emp_name, Emp_id, Address, mail_id, Mobile_no and 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 as staff club funds. Generate pay slips for the employees with their gross and net salary.

ALGORITHM:

Step 1: Start the process

Step 2: Prompt the user with converter choice 1. Programmer 2. Assistant Professor 3. Associate Professor 4. Professor 5. Exit and get the choice.

Step 3: If user selects a Programmer then proceed to step 4

Step 4: Get the user details [Name, ID, Address, Mail ID and Mobile Number] and goto step 5

Step 5: Proceed to Programmers Payment Processing

Step 5.1: Get the basic pay of Programmer

Step 5.2: If user enters -1 assume basic pay as 30000 and goto step 15

Step 5.3: Else directly go to step 15

Step 6: If user selects Assistant Professor step 7

Step 7: Get the user details [Name, ID, Address, Mail ID and Mobile Number] and goto step 8

Step 8: Proceed to Assistant Professor Payment Processing

Step 8.1: Get the basic pay of Assistant Professor

Step 8.2: If user enters -1 assume basic pay as 25000 and goto step 15

Step 8.3: Else directly go to step 15

Step 9: If user selects Associate Professor step 10

Step 10: Get the user details [Name, ID, Address, Mail ID and Mobile Number] and goto step 11

Step 11: Proceed to Associate Professor Payment Processing

Step 11.1: Get the basic pay of Associate Professor

Step 11.2: If user enters -1 assume basic pay as 40000 and goto step 15

Step 11.3: Else directly go to step 15

Step 12: If user selects Professor step 13

Step 13: Get the user details [Name, ID, Address, Mail ID and Mobile Number] and goto step 14

Step 14: Proceed to Professor Payment Processing

Step 14.1: Get the basic pay of Professor

Step 14.2: If user enters -1 assume basic pay as 70000 and goto step 15

Step 14.3: Else directly go to step 15

Step 15: Compute Per_Day_Pay = original_basic_pay / no_of_days_in_the_current_month

Step 16: Get the number of days worked from user that include Cl, WH, FH and exclude the LOP

Step 17: Check no_days_worked <= no_of_days_in_the_current_month. Else display "ErrorMessage" and goto step 18

Step 18: Compute Current_Basic_Pay = Per_Day_Pay * no_days_worked

Step 19: Compute Following and Store DA = (Current_Basic_Pay/100) * 97HRA = (Current_Basic_Pay/100) * 12PF = (Current_Basic_Pay/100) * 0.1

GROSS_PAY = Current_Basic_Pay + DA + HRA + PFNET_PAY = GROSS_PAY - PF

Step 17: Display Payment Details [Name, ID, Address, Mail ID, Mobile Number, BASIC PAY,DA, HRA,PF,GROSS_PAY, NET_PAY].

Step 18: Stop Processing

PROGRAM:

```
import java.util.Scanner;
class Employee
{
        int Em_id;
        String Emp_name;
        String Address;
        String Mail_Id;
        String Mail_no;
        Employee(){}
        Employee(int id, String name, String addr, String mail, String mob)
        {
                this.Emp_id = id;
                this.Emp_name = name;
                this.Address = addr;
```

```
                    this.Mail_Id = mail;
                    this.Mobile_no = mob;
          }
}
class Prorammer extends Employee
{
          double BP, Gross_salary, Net_salary;
          public Programmer(int id, String name, String addr, String mail, String mob)
          {
                    super(id, name, addr, mail, mob);
          }
          void computePay()
          {
                    System.out.println("Enter Basic Pay:");
                    Scanner input = new Scanner(System.in);
                    BP = input.nextDouble();
                    double DA,HRA, PF,Fund;
                    DA = (BP*97/100);
                    HRA = (BP*10/100);
                    PF = (BP*12/100);
                    Fund = (BP*0.1/100);
                    Gross_salary = BP + DA + HRA;
                    Net_salary = BP + DA + HRA – (PF + Fund);
                    System.out.println("Emp_Id:" +Emp_id);
                    System.out.println("Emp_Name:" + Emp_name);
                    System.out.println("Address:" + Address);
                    System.out.println("Mail_ID:" + Mail_Id);
                    System.out.println("Mobile Number:" + Mobile_no);
                    System.out.println("Gross Pay:" + Gross_salary);
                    System.out.println("Net Pay:" + Net_salary);
          }
}
class Asst_Professor extends Employee
{
        double BP,Gross_salary,Net_salary;
        public Asst_Professor(int id,String name, String addr, String mail, String mob)
        {
                super(id,name,addr,mail,mob);
        }
        void computePay()
        {
                System.out.println(“Enter Basic Pay:”);
                Scanner input = new Scanner(System.in);
                BP = input.nextDouble();
                Gross_salary =  BP;
                Double DA,HRA,PF,Fund;
                DA = (BP*97/100);
                HRA = (BP*10/100);
                PF = (BP*12/100);
                Fund = (BP*0.1/100);
                Net_salary = BP + DA + HRA – (PF + Fund);
                System.out.println(“Emp_Id:” + Emp_id);
                System.out.println(“Emp_Name:” + Emp_name);
                System.out.println(“Address:” + Address);
                System.out.println(“Mail_Id:” + Mail_Id);
                System.out.println(“Mobile Number:” + Mobile_no);
                System.out.println(“Gross Pay:” + Gross_salary);
                System.out.println(“Net Pay:” + Net_salary);
        }
}
class Associate_Professor extends Employee
{
        double BP,Gross_salary,Net_salary;
        public Associate_Professor(int  id, String name, String addr, String mail, String mob)
        {
                super(id,name,addr,mail,mob);
        }
        void computePay()
        {
                System.out.println("Enter Basic Pay:");
                Scanner input = new Scanner(System.in);
                BP = input.nextDouble();
                Gross_salary =  BP;
                Double DA,HRA,PF,Fund;
                DA = (BP*97/100);
                HRA = (BP*10/100);
                PF = (BP*12/100);
                Fund = (BP*0.1/100);
                Net_salary = BP + DA + HRA – (PF + Fund);
                System.out.println("Emp_Id:" + Emp_id);
                System.out.println("Emp_Name:" + Emp_name);
                System.out.println("Address:" + Address);
                System.out.println("Mail_Id:" + Mail_Id);
                System.out.println("Gross Pay:" + Gross_salary);
                System.out.println("Net Pay:" + Net_salary);
        }
}
class Professor extends Employee
{
        double BP,Gross_salary,Net_salary;
        public Professor(int  id, String name, String addr, String mail, String mob)
        {
                super(id,name,addr,mail,mob);
        }
        void computePay()
        {
                System.out.println("Enter Basic Pay:");
                Scanner input = new Scanner(System.in);
                BP = input.nextDouble();
                Gross_salary =  BP;
                Double DA,HRA,PF,Fund;
                DA = (BP*97/100);
                HRA = (BP*10/100);
                PF = (BP*12/100);
                Fund = (BP*0.1/100);
                Net_salary = BP + DA + HRA – (PF + Fund);
                System.out.println(“Emp_Id:” + Emp_id);
                System.out.println(“Emp_Name:” + Emp_name);
                System.out.println(“Address:” + Address);
                System.out.println(“Mail_Id:” + Mail_Id);
                System.out.println(“Gross Pay:” + Gross_salary);
                System.out.println(“Net Pay:” + Net_salary);
        }
}
public class PaySlip
{
        public static void main(String[] args)
        {
         Programmer p=new Programmer(10,”AAA”,”xxx”,”aaa_xxx@gmail.com”,”1111111111”);
         System.out.println(“----------Programmer----------“);
         p.computePay();
        Asst_Professor Ap= new
        Asst_Professor(20,”BBB”,”yyy”,”bbb_yyy@gmail.com”,”2222222222”);
         System.out.println(“----------Associate Professor----------“);
         Ap.computePay();
         Associate_Professor As =  new
        Associate_Professor(30,”CCC”,”zzz”,”ccc_zzz@gmail.com”,”3333333333”);
         System.out.println(“----------Associate Professor----------“);
         As.computePay();
         Professor pf = new
         Professor(40,”DDD”,”www”,”ddd_www@gmail.com”,”4444444444”);
         System.out.println(“-----------Professor----------“);
         Pf.computePay();
        }
}
```

OUTPUT:

```
----------Programmer----------
Enter Basic Pay:5000
Emp_Id: 10
Emp_Name: AAA
Address: xxx
Mail_Id: aaa_xxx@gmail.com
Mobile Number: 1111111111
Gross Pay: 10350.0
Net Pay: 9745.0
----------Assistant Professor----------
Enter Basic Pay: 10000
Emp_Id: 20
Emp_Name: BBB
Address: yyy
Mail_Id: bbb_yyy@gmail.com
Mobile Number: 2222222222
Gross Pay: 10000.0
Net Pay: 19490.0
----------Associate Professor---------
Enter Basic Pay: 15000
Emp_Id: 30
Emp_Name: CCC
Address: zzz
Mail_Id: ccc_zzz@gmail.com
Mobile Number: 3333333333
Gross Pay: 15000.0
Net Pay: 29235.0
----------Professor----------
Enter Basic Pay: 20000
Emp_Id: 40
Emp_Name: DDD
Address: www
```

```
Mail_Id: ddd_www@gmail.com
Mobile Number: 44444444444
Gross Pay: 20000.0
Net Pay: 38980.0
```

RESULT:

Thus the java program for to develop a java application with an employee class with Emp_name, Emp_id, Address, mail_id, Mobile_no and 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 as staff club funds. Generate pay slips for the employees with their gross and net salary was executed and the output is verified successfully.

Ex. No:4 **Date:**	**Write a java program to create an abstract class named shape that contains two integers and an empty method named print Area (). 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 print Area() that prints the area of the given shape.**

AIM:

To write a java program to create an abstract class named Shape that contains tow integers and an empty method named printArea(). Provide three class 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.

ALGORITHM:

```
Step 1: Start the Process
Step 2: Prompt user with List Operation Choices
1. Rectangle 2. Triangle 3. Circle 4, Exit
Get the choice from user.
Step 3: If user selects Rectangle
Step 3.1: Get the Length and Breath from the user
Step 3.2: Compute Area = Length * Breath
Step 3.3: Display Area
Step 4: If user selects Triangle
Step 3.1: Get the Length and Height from the user
Step 3.2: Compute Area = Length * Height * 0.5
Step 3.3: Display Area
Step 4: If user selects Circle
Step 4.1: Get the Radius of the Circle
Step 4.2: Compute Area = 3.14 * Radius * Radius
Step 4.3: Display Area
Step 5: If user selects exit end the process
Step 6: Stop the Process
```

PROGRAM:

```
import jaa.util.*;
abstract class Shapes
{
        double a,b;
        abstract void printArea();
}
class Rectangle extends Shapes
{
        void printArea()
        {
        System.out.println("\t\tCalculating Area of rectangle");
        Scanner input = new Scanner(System.in);
        System.out.println("Enter length:");
        a = input.nextDouble();
        System.out.println("\nEnter breadth:");
        b = intput.nextDouble();
        double area = a*b;
        System.out.println("Area of Rectangle:" +area);
        }
}
class Triangle extends Shapes
{
        void printArea()
        {
        System.out.println("\t\tCalculating Area of Triangle");
        Scanner input = new Scanner(System.in);
        System.out.println("Enter height: ");
        a = input.nextDouble();
        System.out.println("\nEnter breadth:");
        b = input.nextDouble();
        double area = 0.5*a*a;
        System.out.println("Area of Triangle: "+area);
        }
}
class Circle extends
{
        void printArea()
        {
        System.out.println("\t\tCalculating Area of Circle");
        Scanner input = new Scanner(System.in);
        System.out.print("Enter radius:");
        a = input.nextDouble();
        double area = 3.14*a*a;
        System.out.println("Area  of Circle:" +area);
        }
}
class AbstractClassDemo
{
        public Static void main(String[] args)
        {
        shapes obj;
        obj = new Rectangle();
        obj.printArea();
        obj = new Triangle();
        obj.printArea();
        obj.PrintArea();
        }
}
```

OUTPUT:

```
D:\> javac AbstractClassDemo.java
D:\> javac 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
```

RESULT:

Thus the java program to create an abstract class named Shape that contains tow integers and an empty method named printArea(). Provide three class 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 was executed and the output is verified successfully.

Ex. No:5 **Date:**	**Write a java program to create an abstract class named shape that contains two integers and an empty method named print Area (). 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 print Area() that prints the area of the given shape using interface.**

AIM:

To write a java program to create an abstract class named Shape that contains tow integers and an empty method named printArea(). Provide three class 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 using interface.

ALGORITHM:

Step 1: Start the Process

Step 2: Prompt user with List Operation Choices

1. Rectangle 2. Triangle 3. Circle 4, Exit

Get the choice from user.

Step 3: If user selects Rectangle

Step 3.1: Get the Length and Breath from the user

Step 3.2: Compute Area = Length * Breath

Step 3.3: Display Area

Step 4: If user selects Triangle

Step 3.1: Get the Length and Height from the user

Step 3.2: Compute Area = Length * Height * 0.5

Step 3.3: Display Area

Step 4: If user selects Circle

Step 4.1: Get the Radius of the Circle

Step 4.2: Compute Area = 3.14 * Radius * Radius

Step 4.3: Display Area

Step 5: If user selects exit end the process

Step 6: Stop the Process.

PROGRAM:

```
interface area
{
        double pi = 3.14;
        double calc(double x, double y);
}
class rect implements area
{
        public double calc(double x, double y)
        {
                return (x*y);
        }
}
class cir implements area
{
        public double calc(double x, double y)
        {
                return(pi*x*x);
        }
}
class test7
{
        public static void main(String arg[])
        {
                rect r = new rect();
                cir c = new cir();
                area a;
                a = r;
                System.out.println("\nArea of Rectangle is: " +a.calc(10,20);
                a = c;
                System.out.println("\nArea of Circle is: "+a.calc(15,15);
        }
}
```

OUTPUT:

```
Area of Rectangle is: 200.0
Area of Circle is: 706.5
```

RESULT:

Thus the java program to create an abstract class named Shape that contains tow integers and an empty method named printArea(). Provide three class 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 using interface was executed and the output is verified successfully.

Ex.No:6 Date:	**Implement exception handling and creation of user defined exception**

AIM:

To write a java program to implement exception handling and reation of user defined exception.

ALGORITHM:

Step 1: Create a class runerrdemo

Step 2: In try block write a code for checking the statements for if its error or not

Step 3: In catch block write the error for occurring your statement

Step 4: Return the result

PROGRAM:

```
class RunErrDemo
{
        public static void main(String[] args)
        {
        int a,b,c;
        a = 10;
         b = 0;
        try
        {
        c = a/b;
        }
        catch(ArithmeticException 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
```

RESULT:

Thus the java program for implementation exception handling and creation of user defined exception was executed and output is verified successfully.

Ex. No:7 Date:	**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 values 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.**

AIM:

To 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 values 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.

ALGORITHM:

Step 1: Start the Process
Step 2: Create a thread that generates random number
Step 3: Obtain one random number and check is odd or even
Step 3.1: If number is even then create and start thread that computes square of a number
Step 3.2: Compute number * number and display the answer
Step 3.3: Notify to Random number thread and goto step 4
Step 3.4: If number is odd then create and start thread that computes cube of a number
Step 3.4: Compute number * number * number and display the answer
Step 3.5: Notify to Random number thread and goto step 4
Step 4: Wait for 1 Second and Continue to Step 3 until user wants to exits.
Step 5: Stop the Process

PROGRAM:

```
import java.util.*; class NumberGenerate

{

private int value;

private boolean flag: // producer public synchronized void put()

{

while (flag)

{
```

```
try {

wait();

} catch (InterruptedException e) {}

}

flag = true;

Random random = new Random(); this.value = random.nextInt(10);

System.out.println("The generated Number is: "+value);

notify All();

}

// consumer

public synchronized void get1()

{

while (!flag) {

try { wait();

}

catch (InterruptedException e) { }

if(value%2==0)

System.out.println("Second is executing now..."); int ans value *value;

System.out.println(value+" is Even Number and its square is: "+ans);

}
flag = false;

notifyAll();
```

```
}

public synchronized void get2()

{ while (!flag)

{

try

{

wait();

} catch (InterruptedException e) {}

}

if(value%2!=0)

{

System.out.println("Third Thread is executing now...");

int ans-value*value*value; System.out.println(value+" is Odd Number and its cube is: "+a

}

flag false; = notify All();

}

public class TestNumber

public static void main(String[] args)

final NumberGenerate obj = new NumberGenerate();

Thread producerThread = new Thread()

{ public void run()

{
```

```
for(int i=1;i<=6;i++)

{

System.out.println("Main thread started...");

obj.put();

try {

Thread.sleep(1000); }catch(InterruptedException e){}

}

};
Thread consumerThread1 = new Thread()

public void run()

{ for(int i=1;i<=3;i++) {

obj.get1();

try

{ Thread.sleep(2000);

}

catch(InterruptedException e){}

}

}

};

Thread consumerThread2 = new Thread() {

public void run() { for(int i=1;i<=3;i++)

{

obj.get2():
```

```
try

{

Thread.sleep(3000);

} catch(InterruptedException e){}

}

}

producerThread.start();
 consumerThread1.start();
 consumerThread2.start();

}

}
```

OUTPUT:

```
Main thread started...

The generated Number is: 6

Second is executing now...

6 is Even Number and its square is: 36 Main thread started...
The generated Number is: 4

Main thread started... The generated Number is: 1

Main thread started...

The generated Number is: 1 Third Thread is executing now...

1 is Odd Number and its cube is: 1

Main thread started....

The generated Number is: 0 Second is executing now...

0 is Even Number and its square is: 0 Main thread started....

The generated Number is: 1

Third Thread is executing now...

1 is Odd Number and its cube is: 1
```

RESULT:

Thus the java program for the implements a multi-threaded application that has three threads. First thread generates a random integer every 1 second and if the values 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 was executed and output is verified successfully.

Ex.No:8 **Date:**	**Write a program to perform file operation**

AIM:

To write a java program to perform file operation.

ALGORITHM:

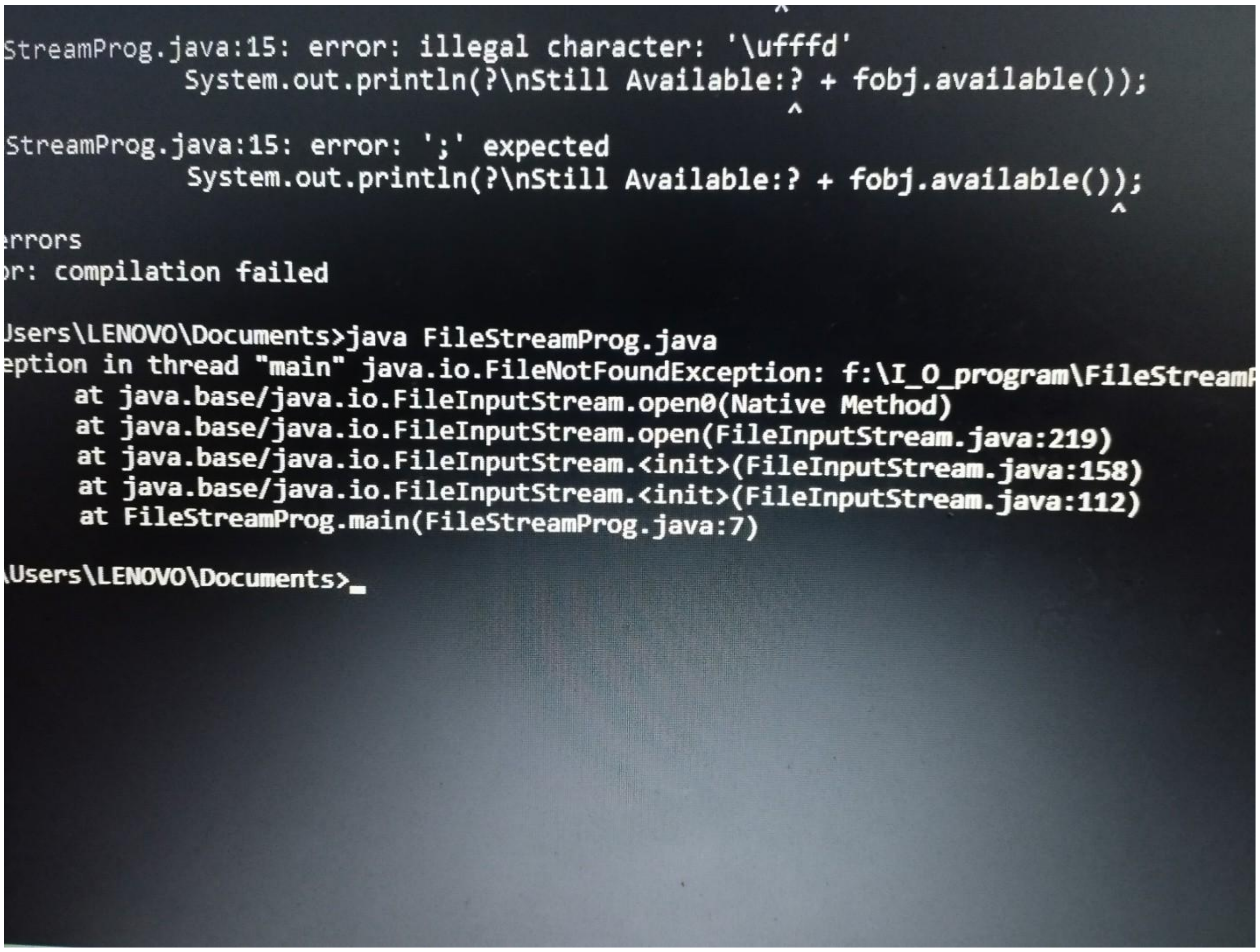
```
Step 1: Start the Process
Step 2: Prompt the user to enter the file name with path
Step 3: Get the file name
Step 3.1: Check the file is exits
Step 3.2: If file exists then proceed to step 3.3 else proceed to step 3.8
Step 3.3: Check the File is Both Readable and Writeable
Step 3.4: If yes display file is “Read and Writeable”
Step 3.5: Else check is readable if yes display “Read Only” else move to step 3.6
Step 3.6: Else check is writeable if yes display “Write Only”
Step 3.7: Compute file size and display
Step 3.8: If file not existing then display “File Not Found”
Step 4: Stop the Process
```

PROGRAM:

```
import java.io.*;
class FileStreamProg
{
	public static void main(String[] args)throws Exception
	{
		int n;
		InputStream fobj = new FileInputStream(“f:/I_O_program/FileStreamProg.java”);
		System.out.println(“Total available bytes:” + (n=fobj.available()));
		int m=n-400;
		System.out.println(“\nReading first “+m+”bytes at a time”);
		for(int I =0; i<m; i++)
		System.out.println((char)fobj.read());
		System.out.println(“\nSkipping some text”);
```

```
                fobj.skip(n/2);
                System.out.println("\nStill Available:" + fobj.available());
                fobj.close();
        }
}
```

OUTPUT:



```
StreamProg.java:15: error: illegal character: '\ufffd'
            System.out.println(?\nStill Available:? + fobj.available());
                                                  ^
StreamProg.java:15: error: ';' expected
            System.out.println(?\nStill Available:? + fobj.available());
                                                                        ^
rrors
r: compilation failed

Users\LENOVO\Documents>java FileStreamProg.java
eption in thread "main" java.io.FileNotFoundException: f:\I_O_program\FileStream
     at java.base/java.io.FileInputStream.open0(Native Method)
     at java.base/java.io.FileInputStream.open(FileInputStream.java:219)
     at java.base/java.io.FileInputStream.<init>(FileInputStream.java:158)
     at java.base/java.io.FileInputStream.<init>(FileInputStream.java:112)
     at FileStreamProg.main(FileStreamProg.java:7)

Users\LENOVO\Documents>_
```

RESULT:

Thus the java program for to perform file operation was executed and the output is verified successfully.

Ex.No:9 Date:	**Develop application to demonstrate the features of generics classes**

AIM:

To wirte a java program to develop application to demonstrate the features of generics classes

ALGORITHM:

Step 1: Start the Process

Step 2: Create a array of number and array of strings and pass it to generic function.

Step 3: If the array is Integer type

Step 3.1: Assume first element as MAX

Step 3.2: Compare [Numeric Perspetive] this element with MAX

Step 3.3: If it is greater than MAX then store current element as MAX

Step 3.4: Else do nothing

Step 3.5: Gotostep 3.1 until all the elements has been processed.

Step 4: If the array is String type

Step 4.1: Assume first element as MAX

Step 4.2: Compare [Dictionary Perspective] this element with MAX

Step 4.3: If it is greater than MAX then store current element as MAX

Step 4.4: Else do nothing

Step 4.5: Gotostep 3.1 until all the elements has been processed.

Step 5: Stop the Process

PROGRAM:

```
classGenericMax {
public<T extends Comparable<T>> void maxFinder (T[] array){
T max = array[0];
for(T element: array){
System.out.println(element);
if(element.compareTo(max) > 0)
max = element;
}
System.out.println("Max is : "+max);
}
}
public class Main {
public static void main(String[] args) {
GenericMax max = new GenericMax();
Integer[] numbers = {14,3,42,5,6,10};
String[] strings = {"R","Ra","Raj"};
max.maxFinder(numbers);
max.maxFinder(strings);
}
}
```

OUTPUT:

```
3
42
5
6
10
Max is:- 42
R
Ra
Raj
Max is:-Raj
```

RESULT:

Thus the java program for to develop application to demonstrate the features of generics classes was executed and the output is verified successfully.

Ex.No:10a **Date:**	**Develop application using JavaFx Controls**

AIM:

To write a java program to develop application using javaFx controls.

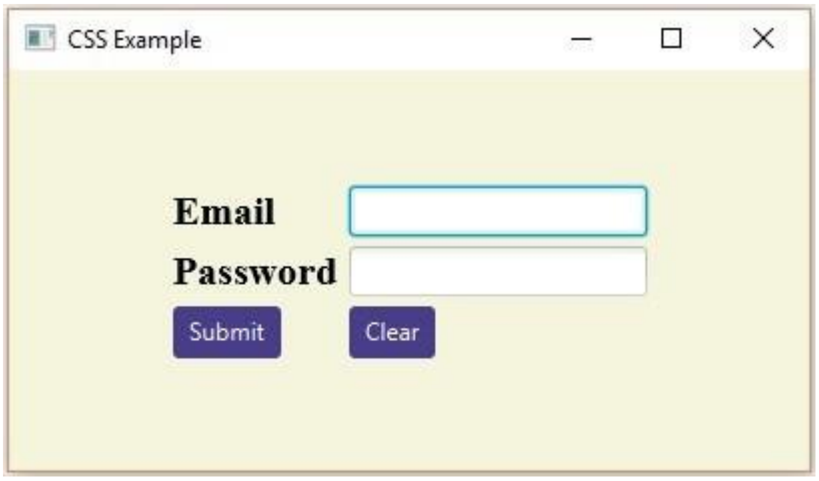
ALGORITHM:

Step 1: The first step to applying an animation to a specific node is to create that node using the respective class of the node.
Step 2: After creating the node, the next step is to instantiate the respective animation class that is to be applied to the created node
Step 3: After instantiating the respective animation class, in the step, the user should set the properties of the animation.
Step 4: In the next step, the user should set the target node on which the animation will be applied.
Step 5: After following all the above steps, the last step to apply an animation to a target node is to play the applied animation using the play() method of the Animation class.

PROGRAM:
```
importjavafx.application.Application;
importstaticjavafx.application.Application.launch;
importjavafx.geometry.Insets;
importjavafx.geometry.Pos;

importjavafx.scene.Scene;
importjavafx.scene.control.Button;
importjavafx.scene.control.PasswordField;
importjavafx.scene.layout.GridPane;
importjavafx.scene.text.Text;
importjavafx.scene.control.TextField;
importjavafx.stage.Stage;

publicclassLoginPageextendsApplication{
```

```
@Override
publicvoidstart(Stage stage){
//creating label email
Text text1 =newText("Email");

//creating label password
Text text2 =newText("Password");

//Creating Text Filed for email
TextField textField1 =newTextField();

//Creating Text Filed for password
PasswordField textField2 =newPasswordField();

//Creating Buttons
Button button1 =newButton("Submit");
Button button2 =newButton("Clear");

//Creating a Grid Pane
GridPanegridPane=newGridPane();

//Setting size for the pane
gridPane.setMinSize(400,200);

//Setting the padding
gridPane.setPadding(newInsets(10,10,10,10));

//Setting the vertical and horizontal gaps between the columns
gridPane.setVgap(5);
gridPane.setHgap(5);

//Setting the Grid alignment
gridPane.setAlignment(Pos.CENTER);
//Arranging all the nodes in the grid
gridPane.add(text1,0,0);
gridPane.add(textField1,1,0);
gridPane.add(text2,0,1);
gridPane.add(textField2,1,1);
gridPane.add(button1,0,2);
gridPane.add(button2,1,2);

//Styling nodes
    button1.setStyle("-fx-background-color: darkslateblue; -fx-text-fill: white;");
    button2.setStyle("-fx-background-color: darkslateblue; -fx-text-fill: white;");

    text1.setStyle("-fx-font: normal bold 20px 'serif' ");
    text2.setStyle("-fx-font: normal bold 20px 'serif' ");
gridPane.setStyle("-fx-background-color: BEIGE;");

//Creating a scene object
Scenescene=newScene(gridPane);

//Setting title to the Stage
stage.setTitle("CSS Example");

//Adding scene to the stage
stage.setScene(scene);

//Displaying the contents of the stage
stage.show();
}
publicstaticvoidmain(Stringargs[]){
    launch(args);
}
}
```

OUTPUT:



RESULT:

Thus the java program for to develop application using javaFx controls was executed and the output is verified successfully.

Ex.No:10b **Date:**	**Develop application using Layout**

AIM:

To write a java program to develop application using layout.

ALGORITHM:

Step 1: import the java pakage for graphical usage named awt and swing

Step 2: create a constructor BorderLayoutExample() and create a frame init

Step 3: create Button for the layout and creating an object of the BorderLayout class using

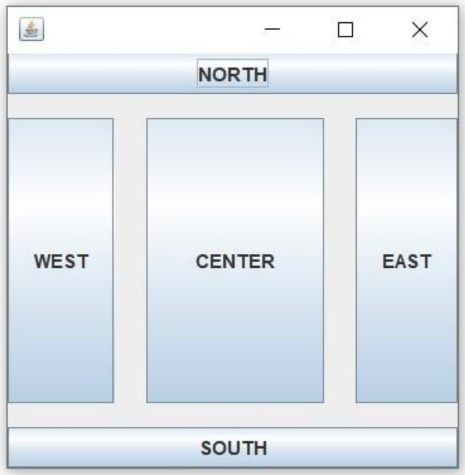
Step 4: the parameterized constructor where the horizontal gap is 20 and vertical gap is 15. The gap will be evident when buttons are placed in the frame

PROGRAM:

```
// import statement
import java.awt.*;
import javax.swing.*;
public class BorderLayoutExample
{
JFrame jframe;
// constructor
BorderLayoutExample()
{
   // creating a Frame
   jframe = new JFrame();
   // create buttons
   JButton btn1 = new JButton("NORTH");
   JButton btn2 = new JButton("SOUTH");
   JButton btn3 = new JButton("EAST");
   JButton btn4 = new JButton("WEST");
   JButton btn5 = new JButton("CENTER");
    // creating an object of the BorderLayout class using
    // the parameterized constructor where the horizontal gap is 20
    // and vertical gap is 15. The gap will be evident when buttons are placed
```

```
     // in the frame
    jframe.setLayout(new BorderLayout(20, 15));
    jframe.add(btn1, BorderLayout.NORTH);
    jframe.add(btn2, BorderLayout.SOUTH);
    jframe.add(btn3, BorderLayout.EAST);
    jframe.add(btn4, BorderLayout.WEST);
    jframe.add(btn5, BorderLayout.CENTER);
    jframe.setSize(300,300);
    jframe.setVisible(true);
}
// main method
public static void main(String argvs[])
{
    new BorderLayoutExample();
}
}
```

OUTPUT:



RESULT:

Thus the java program to develop application using Layouts was executed and the output is verified successfully.

Ex.No:10c Date:	Develop application using Menus

AIM:

To write java program to develop application using Menus

ALGORITHMS:

Step 1: Create a Menu bar is the first step, MenuBar can be instantiated using new MenuBar menuBar = new MenuBar();

Step 2: Now create the menu. The name of the menu is passed as an argument to the Menu class.Menu fileMenu = new Menu("File");

Step 3: Create the menuitem. The name of the menu is passed as an argument to the Menultem class.Menuitem newitem = new MenuItem("New");

Step 4: Add menu items to the menu using add or addAll method fileMenu.getItems().addAll(newitem, openFileltem,saveltem, exitItem);

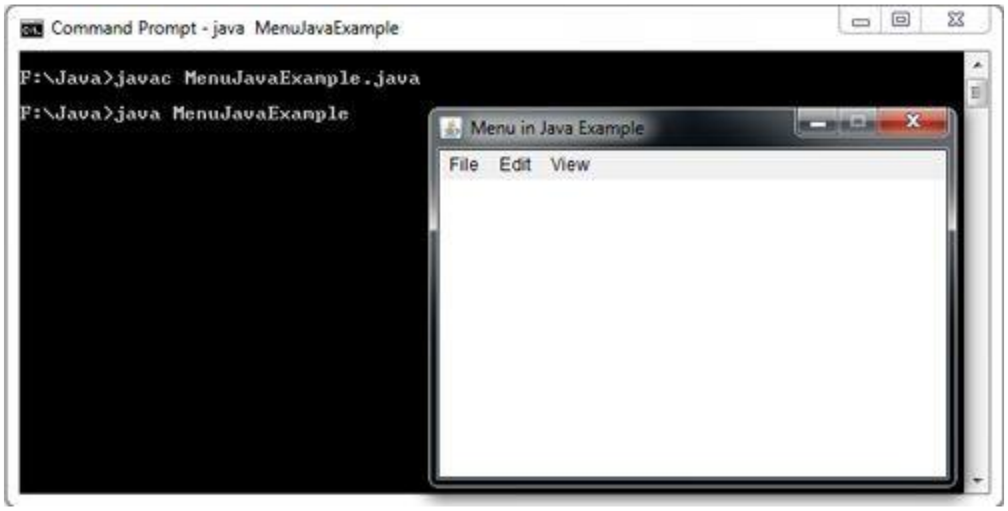
Step 5: Add Menu to Menu Bar using add or addAll method. menuBar.getMenus().addAll(file Menu, editMenu, aboutMenu);

PROGRAM:

```
importjava.awt.*;
classMenuExampleextendsFrame
{
    MenuExample()
    {
        MenuBarmenuBar=newMenuBar();
        setMenuBar(menuBar);
        MenumenuFile=newMenu("File");
        MenumenuEdit=newMenu("Edit");
        MenumenuView=newMenu("View");
        menuBar.add(menuFile);
        menuBar.add(menuEdit);
        menuBar.add(menuView);
        MenuItemitemOpen=newMenuItem("Open");
```

```
            MenuItemitemSave=newMenuItem("Save");
            MenuItemitemExit=newMenuItem("Exit");
            menuFile.add(itemOpen);
            menuFile.add(itemSave);
            menuFile.add(itemExit);
            MenuItemitemcopy=newMenuItem("Copy");
            menuEdit.add(itemcopy);
      }
}
 classMenuJavaExample
 {
          publicstaticvoid main(Stringargs[])
         {
             MenuExample frame =newMenuExample();
             frame.setTitle("Menu in Java Example");
             frame.setSize(350,250);
             frame.setResizable(false);
             frame.setVisible(true);
         }
 }
```

OUTPUT:



```
F:\Java>javac MenuJavaExample.java
F:\Java>java MenuJavaExample
```

RESULT:

Thus the java program for to Develop application using Menus was executed and the output is verified successfully.

Ex.No:11 **Date:**	**Develop a mini project for anyapplication using java concepts**

AIM:

To write a java program to develop a mini project for any application using java concepts

ALGORITHMS:

Step 1: Start the process

Step 2: Get the user informations [Name, Consumer Number, Reading’s of previous and current month, Connection Type]

Step 3: Compute units consumed by user [Current Month Reading – Previous Month Reading]

Step 4: If a connection type is domestic goto step 6

Step 5: Else goto step 7

Step 6: Initialize i with 1 and sum as 0

Step 6.1: If check i is less than or equal to 100 then compute sum = sum + 1 and goto Step 6.5

Step 6.2: Else if check i is greater than 100 and less than 200 then compute sum = sum + 2.5 and goto step 6.5

Step 6.3: Else if check i is greater than 200 and less than 500 then compute sum = sum + 4 and goto step 6.5

Step 6.4: Else compute sum = sum + 6 and goto step 6.5

Step 6.5: If i is equal to number of units consumed goto step 8 else return to same step

Step 7: Initialize i with 1 and sum as 0

Step 7.1: If check i is less than or equal to 100 then compute sum = sum + 2 and goto step 7.5

Step 7.2: Else if check i is greater than 100 and less than 200 then compute sum = sum + 4.5 and goto step 7.5

Step 7.3: Else if check i is greater than 200 and less than 500 then compute sum = sum + 6 and goto step 7.5

Step 7.4: Else compute sum = sum + 7 and goto step 7.5

Step 7.5: If i is equal to number of units consumed goto step 8 else return to same step

Step 8: Store the sum

Step 9: Display Bill Details [Name, Consumer Number, No of units consumed]

Step 10: Check the connection type

Step 10.1: If connection type is domestic display domestic tariff slot and goto step 11

Step 10.1: Else display commercial tariff slot and goto step 11

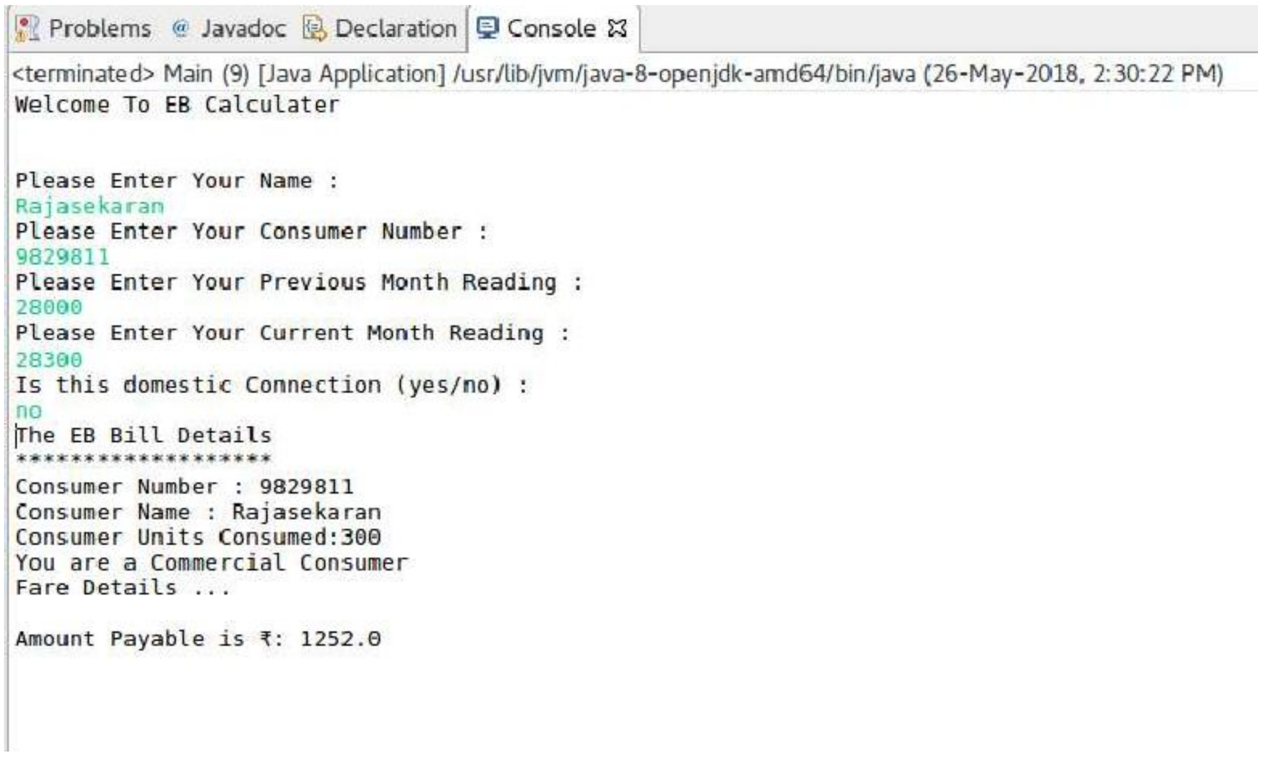
Step 11: Display amount payable using stored sum

Step 12: Stop the Process

PROGRAM:

```
import java.util.Scanner;
class EBConsumer {
int consumer_number;
String consumer_name;
int previous_month_reading;
int current_month_reading;
int units_consumed;
boolean isDomestic = false;
double bill_ammount;
public void displayDomesticFares(){
System.out.println("Domestic Fare Details");
System.out.println("*********************");
System.out.println("First 100 units - Rs. 1 per unit");
System.out.println("101-200 units - Rs. 2.50 per unit");
System.out.println("201 -500 units - Rs. 4 per unit");
System.out.println("> 501 units - Rs. 6 per unit");
}
public void displayCommercialFare() {
System.out.println("Commercial Fare Details");
System.out.println("***********************");
System.out.println("First 100 units - Rs. 2 per unit");
System.out.println("101-200 units - Rs. 4.50 per unit");
System.out.println("201 -500 units - Rs. 6 per unit");
System.out.println("> 501 units - Rs. 7 per unit");
}
public void getDetails() {
Scanner inputs = new Scanner(System.in);
System.out.println("Welcome To EB Calculater\n\n");
System.out.println("Please Enter Your Name : ");
this.consumer_name = inputs.next();
System.out.println("Please Enter Your Consumer Number : ");
this.consumer_number = inputs.nextInt();
System.out.println("Please Enter Your Previous Month Reading : ");
this.previous_month_reading = inputs.nextInt();
System.out.println("Please Enter Your Current Month Reading : ");
this.current_month_reading = inputs.nextInt();
System.out.println("Is this domestic Connection (yes/no) : ");
if(inputs.next().equals("yes"))
this.isDomestic = true;
}
public void generateBill(){
int number_of_units_consumed = this.current_month_reading - this.previous_month_reading;
this.units_consumed = number_of_units_consumed;
double sum = 0;
if(isDomestic == true) {
for (int i = 0; i <= number_of_units_consumed; i++) {
if (i <= 100)
sum = sum + 1;
else if (i > 100 && i <= 200)
sum = sum + 2.5;
else if (i > 200 && i <= 500)
sum = sum + 4;
else
sum = sum + 6;
}
}
else {
for (int i = 0; i <= number_of_units_consumed; i++) {
if (i <= 100)
sum = sum + 2;
else if (i > 100 && i <= 200)
sum = sum + 4.5;
else if (i > 200 && i <= 500)
sum = sum + 6;
else
sum = sum + 7;
}
}
this.bill_ammount = sum;
}
public void displayBill() {
generateBill();
System.out.println("The EB Bill Details");
System.out.println("*******************");
System.out.println("Consumer Number : "+this.consumer_number);
System.out.println("Consumer Name : "+this.consumer_name);
System.out.println("Consumer Units Consumed:"+this.units_consumed);
if(this.isDomestic == true)
System.out.println("Your are an Domestic Consumer\nFare Details ...");
else
System.out.println("You are a Commercial Consumer\nFare Details ...");
System.out.println("\nAmount Payable is \u20B9: "+this.bill_ammount);
}}
public class Main {
public static void main(String[] args) {
EBConsumer consumer = new EBConsumer();
consumer.getDetails();
consumer.displayBill();
}
}
```

OUTPUT:



```
<terminated> Main (9) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (26-May-2018, 2:30:22 PM)
Welcome To EB Calculater


Please Enter Your Name :
Rajasekaran
Please Enter Your Consumer Number :
9829811
Please Enter Your Previous Month Reading :
28000
Please Enter Your Current Month Reading :
28300
Is this domestic Connection (yes/no) :
no
The EB Bill Details
*******************
Consumer Number : 9829811
Consumer Name : Rajasekaran
Consumer Units Consumed:300
You are a Commercial Consumer
Fare Details ...

Amount Payable is ₹: 1252.0
```

RESULT:

Thus the java program for to develop a mini project for any application using java concepts was executed and the output is verified successfully.