LAB MANUAL

CS8383
OBJECT ORIENTED PROGRAMMING LABORATORY

Prepared By,

Rajasekaran S

Assistant Professor

KGISL Institute of Technology

Coimbatore.

Email:proffraja@gmail.com

Important Links

Ex.No: 1	**Electricity Bill Calculator**
Date:	

Aim:

To create a Java console application used to generate electricity bill based on connection type (Domestic and Commercial) and consumption. Both are having different tariff slots.

Algorithm:

```
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
```

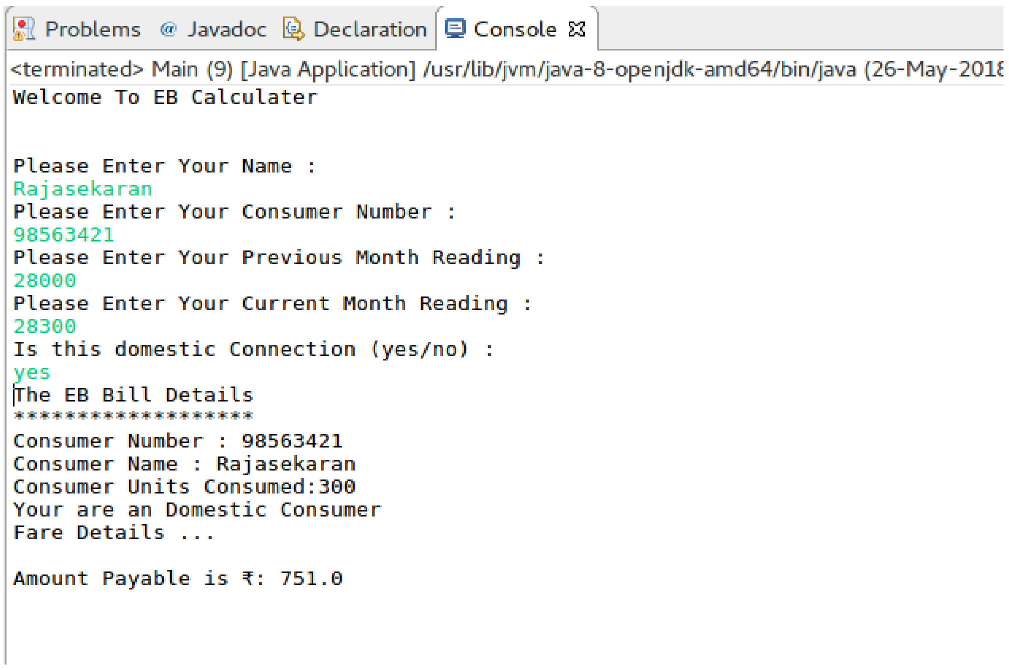
Coding

```
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:

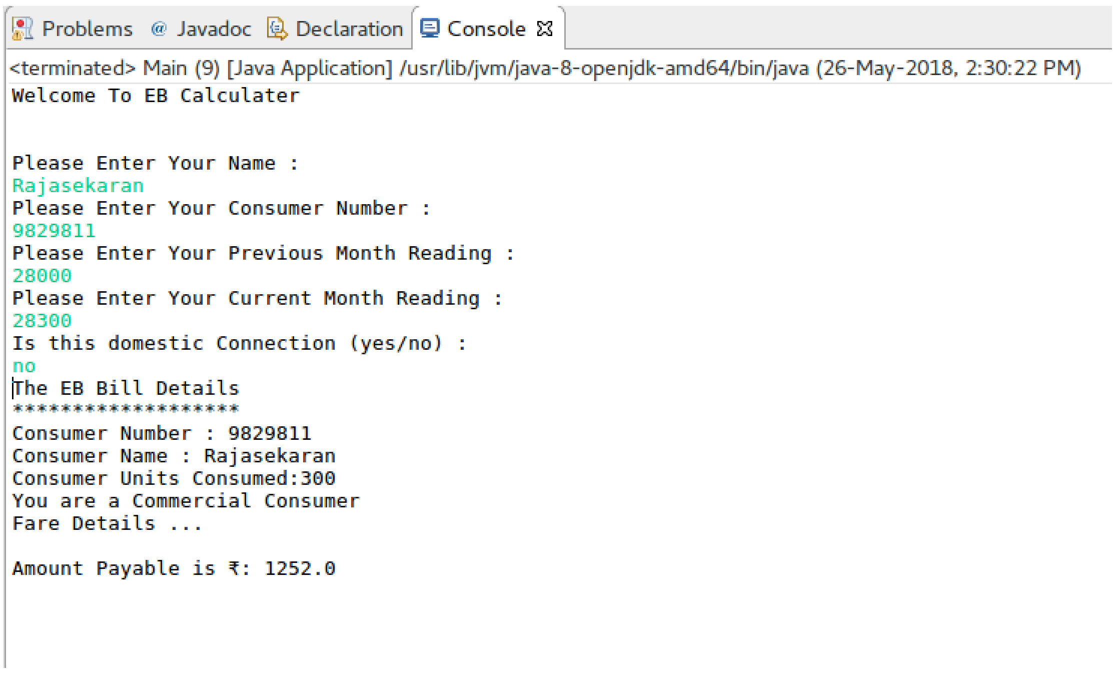
User type is domestic:



Problems @ Javadoc Declaration Console

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


Please Enter Your Name :
Rajasekaran
Please Enter Your Consumer Number :
98563421
Please Enter Your Previous Month Reading :
28000
Please Enter Your Current Month Reading :
28300
Is this domestic Connection (yes/no) :
yes
The EB Bill Details
*******************
Consumer Number : 98563421
Consumer Name : Rajasekaran
Consumer Units Consumed:300
Your are an Domestic Consumer
Fare Details ...

Amount Payable is ₹: 751.0
```

User type is commercial:



Problems @ Javadoc Declaration Console

```
<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:

The Java console application for electricity bill generator was developed and tested successfully.

Ex.No: 2	**Unit Converters Application using packages**
Date:	

Aim:

To create a Java console application that converts currency, distance, time. Converter classes must be separated and used based on the package concept in java.

Algorithm:

```
Step 1   Start the process
Step 2   Prompt the user with converter choice 1. Currency 2.Distance 3. Time 4. Exit and the
         choice.
Step 3   If user selects a Currency Converter then proceed to step 4
Step 4   Proceed with prompting Currency Converter Choices
         1. DOLLER to INR 2. EURO to INR 3. YEN to INR
         4. INR to DOLLER 5.  INR to EURO 6. INR to YEN
         7. Exit and get the user choice.
         Step 4.1  If option 1 selected get in DOLLER and display DOLLER *  66.89 as INR
         Step 4.2  If option 2 selected get in EURO and display EURO *  80 as INR
         Step 4.3  If option 3 selected get in YEN and display YEN *  0.61 as INR
         Step 4.4  If option 4 selected get in INR and display INR /  66.89 as  DOLLER
         Step 4.5  If option 5 selected get in INR and display INR /  80 as EURO
         Step 4.6  If option 6 selected get in INR and display INR /  0.61 as YEN
         Step 4.7  If option 7 selected exit from currency converter choice goto step 2
Step 5   If user selects a Distance Converter then proceed to step 6
Step 6   Proceed with prompting Distance Converter Choices
         1. METER to KILOMETER 2. MILES to KILOMETER
         3. KILOMETER to METER 4.  KILOMETER to MILES
         5. Exit and get the user choice.
         Step 6.1  If option 1 selected get in METER and display METER / 1000 as
                   KILOMETER
         Step 6.2  If option 2 selected get in MILES and display MILES * 1.60934 as
                   KILOMETER
         Step 6.3  If option 3 selected get in KILOMETER and display KILOMETER * 1000 as
                   METER
         Step 6.4  If option 4 selected get in KILOMETER and display KILOMETER / 1.60934
                   as MILES
         Step 6.5  If option 5 selected exit from distance converter choice goto step 2
Step 7   If user selects a Time Converter then proceed to step 8 currency
Step 8   Proceed with prompting Time Converter Choices
         1. HOURS to MINUTES 2. HOURS to SECONDS
         3.  MINUTES to HOURS 4. SECONDS to HOURS
         5. Exit and get the user choice.
         Step 8.1  If option 1 selected get in HOURS and display HOURS * 60 as  MINUTES
```

Step 8.2 If option 2 selected get in HOURS and display HOURS * 3600 as SECONDS

Step 8.3 If option 3 selected get in MINUTES and display MINUTES / 60 as HOURS

Step 8.2 If option 4 selected get in SECONDS and display SECONDS / 3600 as HOURS.

Step 8.5 If option 5 selected exit from tim converter choice and goto step 2

Step 9 If user selects exit then display Thank You !!! and exit from the system

Step 10 Stop the process

Coding:

Currency.java

```
package com.raja.oopslab.converters;

import java.util.Scanner;

public class Currency {

        public static double covertEUROtoINR(double EURO) {
                return EURO * 80;
        }

        public static double convertDOLLARtoINR(double DOLLAR) {
                return DOLLAR * 66.89;
        }

        public static double convertYENtoINR(double YEN) {
                return YEN * 0.61;
        }

        public static double covertINRtoEURO(double INR) {
                return INR * 0.013;
        }

        public static double convertINRtoDOLLAR(double DOLLAR) {
                return DOLLAR * 0.015;
        }

        public static double convertINRtoYEN(double YEN) {
                return YEN * 1.63;
        }

        public static void userChoice(){
        Scanner input = new Scanner(System.in);
     int currency_choice = 0;
     double money = 0;
     while(currency_choice != 7){
        System.out.println("\nCurrency Converter");
        System.out.println("------------------");
        System.out.println("1. DOLLOR to INR\n2. EURO to INR\n3. YEN to INR\n"
                                        + "4. INR to DOLLOR\n5. INR to EURO\n6. INR to YEN\n"
                                        + "7.Exit\n\nEnter Your Choice");
        currency_choice = input.nextInt();
        switch(currency_choice){
            case 1:
                        System.out.println("Enter in DOLLER");
                        money = input.nextDouble();
                        System.out.println(money+" DOLLER is equal to
"+Currency.convertDOLLARtoINR(money)+" INR");
                        break;
            case 2:
                        System.out.println("Enter in EURO");
                        money = input.nextDouble();
                        System.out.println(money+" EURO is equal to
"+Currency.covertEUROtoINR(money)+" INR");
                        break;
            case 3:
                        System.out.println("Enter in YEN");
                        money = input.nextDouble();
                        System.out.println(money+" YEN is equal to
"+Currency.convertYENtoINR(money)+" INR");
                        break;
            case 4:
                        System.out.println("Enter in INR");
                        money = input.nextDouble();
                        System.out.println(money+" INR is equal to
"+Currency.convertINRtoDOLLAR(money)+" DOLLORS");
                        break;
            case 5:
                        System.out.println("Enter in INR");
                        money = input.nextDouble();
                        System.out.println(money+" INR is equal to "+Currency.covertINRtoEURO(money)
+" EURO");
                        break;
            case 6:
                        System.out.println("Enter in INR");
                        money = input.nextDouble();
                        System.out.println(money+" INR is equal to "+Currency.convertINRtoYEN(money)
+" YEN");
                        break;
            case 7:
                        break;
            default:
                        System.out.println("Please choose valid option");
                        break;
            }
        }
    }
}
```

Distance.java

```
package com.raja.oopslab.converters;

import java.util.Scanner;

public class Distance {
        public static double convertMeterToKiloMeter(double meter) {
                return meter / 1000;
        }

        public static double convertMilesToKiloMeter(double miles) {
                return miles * 1.60934;
        }

        public static double convertKiloMetertoMeter(double kilometer) {
                return kilometer * 1000;
        }

        public static double convertKiloMeterToMiles(double kilometer) {
                return kilometer / 1.60934;
        }

        public static void userChoice(){
        Scanner input = new Scanner(System.in);
      int distance_choice = 0;
      double distance = 0;
      while(distance_choice != 5){
        System.out.println("\nDistance Converter");
        System.out.println("------------------");
        System.out.println("1. METER to KILOMETER\n2. MILES to KILOMETER\n"
                                        + "3. KILOMETER to METER\n4. KILOMETER to MILES\
n"
                                        + "5.Exit\n\nEnter Your Choice");
        distance_choice = input.nextInt();
        switch(distance_choice){
        case 1:
                System.out.println("Enter in METERS");
                distance = input.nextDouble();
                System.out.println(distance+" METERS is equal to 
"+Distance.convertMeterToKiloMeter(distance)+" KILOMETER");
                break;
        case 2:
                System.out.println("Enter in MILES");
                distance = input.nextDouble();
                System.out.println(distance+" MILES is equal to 
"+Distance.convertMilesToKiloMeter(distance)+" KILOMETER");
                break;
        case 3:
                System.out.println("Enter in KILOMETER");
                distance = input.nextDouble();
                System.out.println(distance+" KILOMETER is equal to 
"+Distance.convertKiloMetertoMeter(distance)+" METER");
                break;
        case 4:
                System.out.println("Enter in KILOMETER");
                distance = input.nextDouble();
```

```
                        System.out.println(distance+" KILOMETER is equal to
"+Distance.convertKiloMeterToMiles(distance)+" MILES");
                        break;
            case 5:
                        break;
            default:
                        System.out.println("Please choose valid option");
                        break;
            }
        }
    }
}
```

Time.java

```
package com.raja.oopslab.converters;

import java.util.Scanner;

public class Time {
            public static double convertHoursToMinutes(double hours) {
                        return hours * 60;
            }

            public static double convertHoursToSeconds(double hours) {
                        return hours * 60 * 60;
            }

            public static double convertMinutesToHours(double minutes) {
                        return minutes / 60;
            }

            public static double convertSecondsToHours(double seconds) {
                        return seconds / 60 / 60;
            }

            public static void userChoice(){
            Scanner input = new Scanner(System.in);
        int time_choice = 0;
        double time = 0;
        while(time_choice != 5){
            System.out.println("\nTime Converter");
            System.out.println("-----------------");
            System.out.println("1. HOURS to MINUTES\n2. HOURS to SECONDS\n"
                                                + "3. MINUTES to HOURS\n4. SECONDS to HOURS\n"
                                                + "5.Exit\n\nEnter Your Choice");
            time_choice = input.nextInt();
            switch(time_choice){
            case 1:
                        System.out.println("Enter in HOURS");
                        time = input.nextDouble();
                        System.out.println(time+" HOURS is equal to
"+Time.convertHoursToMinutes(time)+" MINUTES");
                        break;
            case 2:
                        System.out.println("Enter in HOURS");
                        time = input.nextDouble();
```

```
                    System.out.println(time+" HOURS is equal to
"+Time.convertHoursToSeconds(time)+" SECONDS");
                    break;
          case 3:
                    System.out.println("Enter in MINUTES");
                    time = input.nextDouble();
                    System.out.println(time+" MINUTES is equal to
"+Time.convertMinutesToHours(time)+" HOURS");
                    break;
          case 4:
                    System.out.println("Enter in SECONDS");
                    time = input.nextDouble();
                    System.out.println(time+" SECONDS is equal to
"+Time.convertSecondsToHours(time)+" HOURS");
                    break;
          case 5:
                    break;
          default:
                    System.out.println("Please choose valid option");
                    break;
          }
       }
    }
}
```

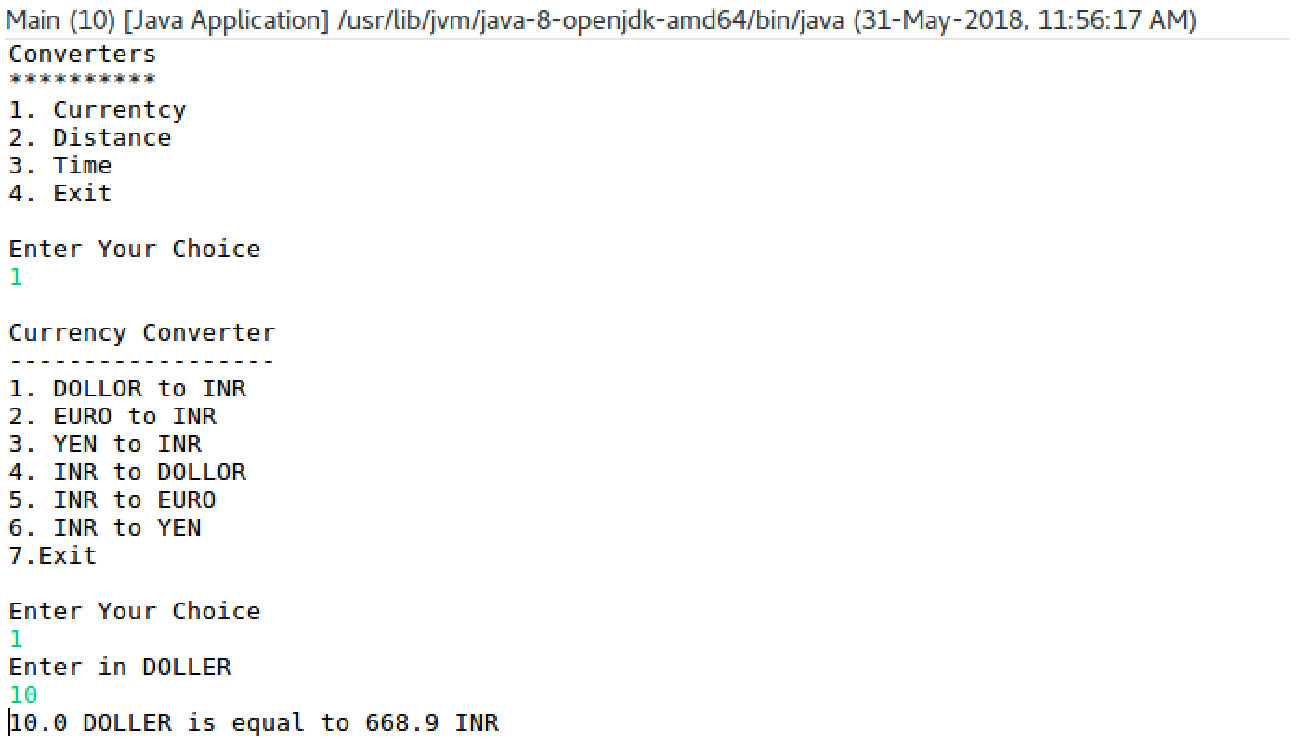
Main.java

```
import java.util.Scanner;

import com.raja.oopslab.converters.Currency;
import com.raja.oopslab.converters.Distance;
import com.raja.oopslab.converters.Time;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        int choice = 0;

        while(choice != 4){
          System.out.println("Converters");
          System.out.println("**********");
          System.out.println("1. Currentcy\n2. Distance\n3. Time\n4. Exit\n\nEnter Your Choice");
          choice = input.nextInt();
          switch(choice){
          case 1:
                    Currency.userChoice();
                    break;
          case 2:
                    Distance.userChoice();
                    break;
          case 3:
                    Time.userChoice();
                    break;
          case 4:
                    break;
          default:
                    System.out.println("Please choose valid option");
```

```
                        break;
                }
            }
            System.out.println("Thank You !!!!");
        }
}
```

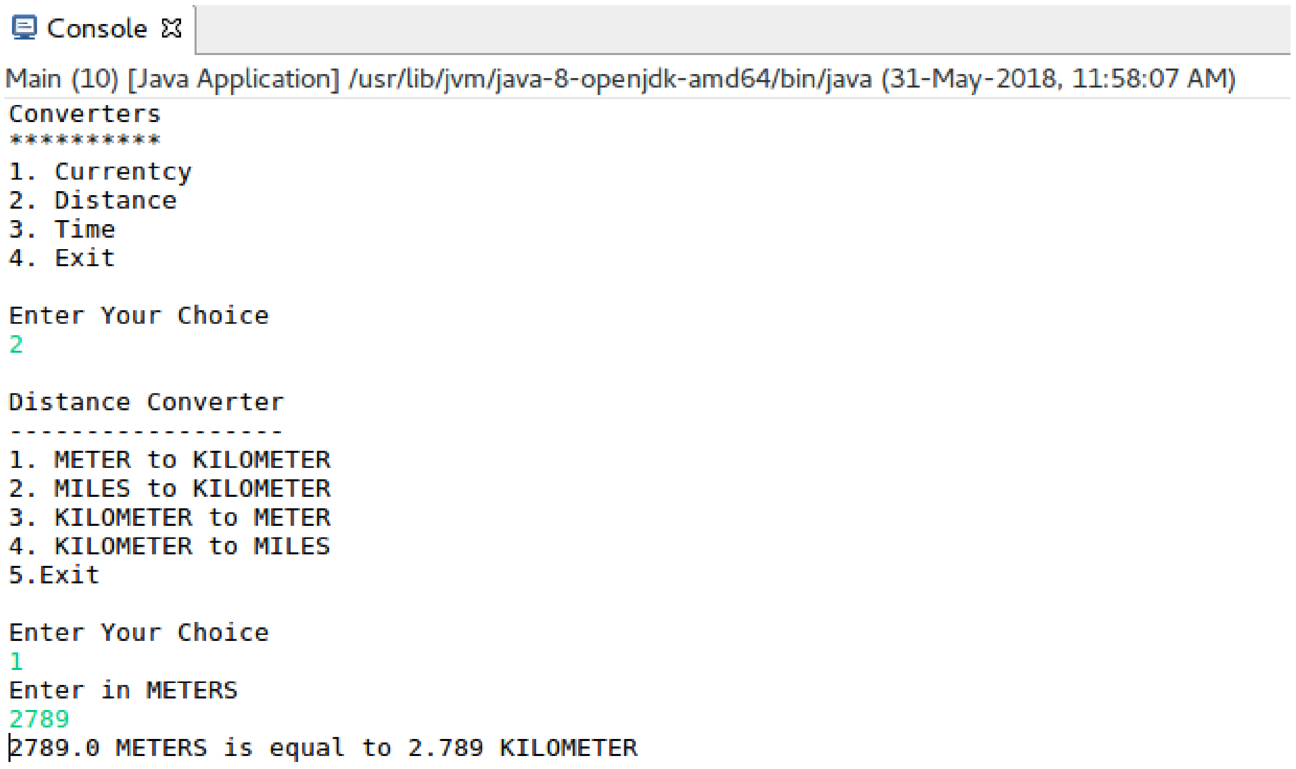
Output

Choice



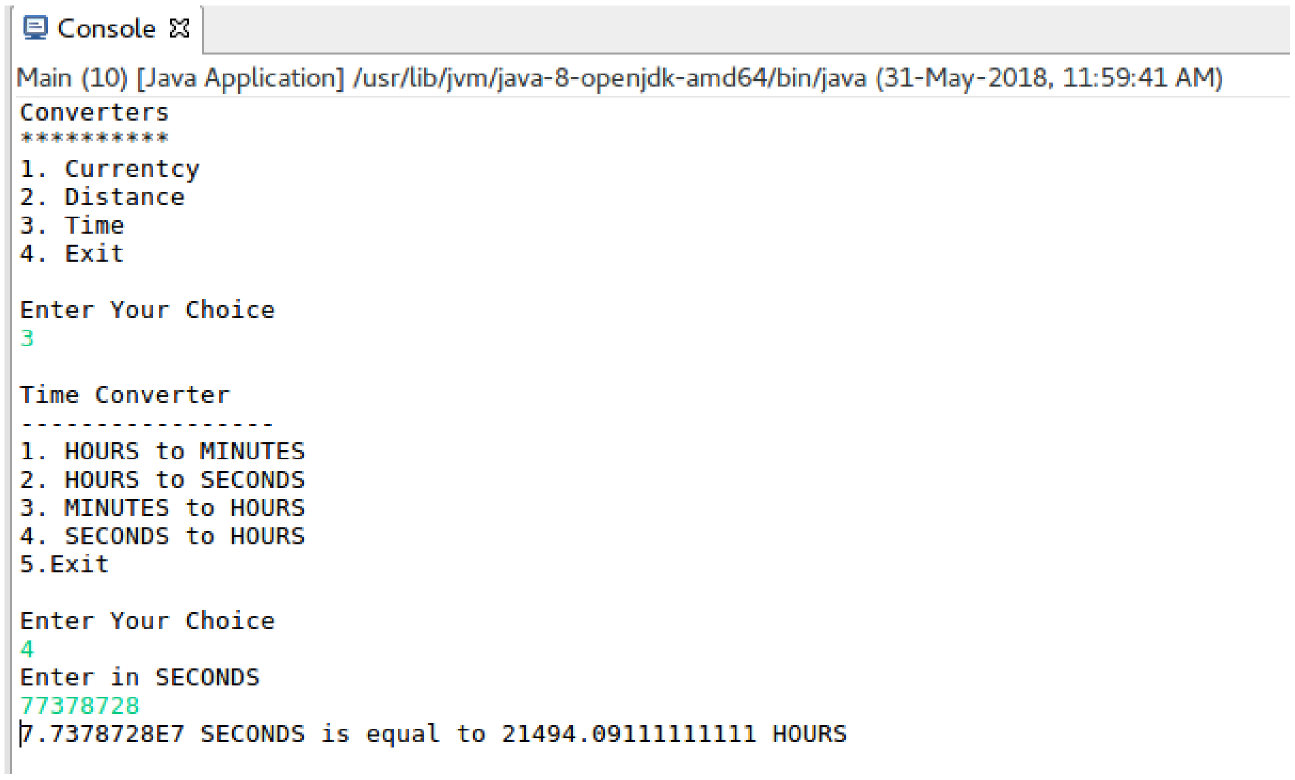
```
Main (10) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (31-May-2018, 11:56:17 AM)
Converters
**********
1. Currentcy
2. Distance
3. Time
4. Exit

Enter Your Choice
1

Currency Converter
------------------
1. DOLLOR to INR
2. EURO to INR
3. YEN to INR
4. INR to DOLLOR
5. INR to EURO
6. INR to YEN
7.Exit

Enter Your Choice
1
Enter in DOLLER
10
10.0 DOLLER is equal to 668.9 INR
```

Distance



```
Console
Main (10) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (31-May-2018, 11:58:07 AM)
Converters
**********
1. Currentcy
2. Distance
3. Time
4. Exit

Enter Your Choice
2

Distance Converter
------------------
1. METER to KILOMETER
2. MILES to KILOMETER
3. KILOMETER to METER
4. KILOMETER to MILES
5.Exit

Enter Your Choice
1
Enter in METERS
2789
2789.0 METERS is equal to 2.789 KILOMETER
```

Min to Hours



Console

Main (10) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (31-May-2018, 11:59:41 AM)

```
Converters
**********
1. Currentcy
2. Distance
3. Time
4. Exit

Enter Your Choice
3

Time Converter
-----------------
1. HOURS to MINUTES
2. HOURS to SECONDS
3. MINUTES to HOURS
4. SECONDS to HOURS
5.Exit

Enter Your Choice
4
Enter in SECONDS
77378728
7.7378728E7 SECONDS is equal to 21494.091111111 HOURS
```

Result

The java console application for converters [Currency, Distance, Time] was developed and tested successfully.

Ex.No: 3	**Employee Payroll System**
Date:	

Aim:

To create a Java console application for employee payroll process management. This project includes Employee and they further classified as Programmer, Assistant Professor, Associate Professor, Professor.

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 “Error
          Message” 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) * 97
         HRA =  (Current_Basic_Pay/100) * 12
         PF =  (Current_Basic_Pay/100) * 0.1
         GROSS_PAY =  Current_Basic_Pay + DA + HRA + PF
         NET_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
```

Coding

Employee.java

```
package com.raja.oopslab.employee;
import java.util.Calendar;
import java.util.GregorianCalendar;
import java.util.Scanner;

package com.raja.oopslab.employee;

import java.util.Calendar;
import java.util.GregorianCalendar;
import java.util.Scanner;

class Employee {
	String emp_name;
	String emp_id;
	String emp_address;
	String emp_mail_id;
	String emp_mobile_no;
	int basic_pay;
	int per_day_pay;
	int current_basic_pay;
	int da, hra, pf, gross_pay;
	int net_pay;
	int no_of_days_in_current_month;
	int no_of_days_worked;
	Calendar cal;
	Scanner input;

	Employee() {
		input = new Scanner(System.in);
		cal = new GregorianCalendar();
		no_of_days_in_current_month = 
cal.getActualMaximum(Calendar.DAY_OF_MONTH);
		getUserBasicDetails();
	}

	public void generatePaySlip() {
		this.da = (this.current_basic_pay / 100) * 97;
		this.hra = (this.current_basic_pay / 100) * 12;
		this.pf = (int) ((this.current_basic_pay / 100) * 0.1);
		this.gross_pay = this.current_basic_pay + da + hra + pf;
		this.net_pay = gross_pay - pf;
	}

	public void displayPaySlip() {
		System.out.println("Name: " + this.emp_name);
		System.out.println("ID: " + this.emp_id);
		System.out.println("Address: " + this.emp_address);
		System.out.println("MailID: " + this.emp_mail_id);
		System.out.println("Mobile No: " + this.emp_mobile_no);
		System.out.println("\nEarnings");
		System.out.println("--------");
		System.out.println("BASIC Pay: " + current_basic_pay + " Rs");
		System.out.println("DA : " + da + " Rs");
		System.out.println("HRA : " + hra + " Rs");
		System.out.println("\nDeductions");
		System.out.println("----------");
		System.out.println("PF : " + pf + " Rs");
		System.out.println("GROSS Pay: " + gross_pay + " Rs");
		System.out.println("NET Pay: " + net_pay + " Rs");
	}

	public void getUserBasicDetails() {
		System.out.println("Enter Details");
		System.out.println("Name: ");
		this.emp_name = input.next();
		System.out.println("ID: ");
		this.emp_id = input.next();
		System.out.println("Address: ");
		this.emp_address = input.next();
		System.out.println("MailID: ");
		this.emp_mail_id = input.next();
		System.out.println("Mobile No:");
		this.emp_mobile_no = input.next();
	}

	public void computeCurrentBasicPay(String empType) {
		this.per_day_pay = this.basic_pay / no_of_days_in_current_month;
		System.out.println("\nBasic Pay of " + empType + " " + this.basic_pay + " for "
				+ this.no_of_days_in_current_month + " days");
		System.out.println("This month this " + empType + " gets " + this.per_day_pay + "
INR as basic pay per day");
		System.out.println("Enter no.of days worked by " + empType + " including CL,
WH, FH and excluding LWP:");
		this.no_of_days_worked = input.nextInt();
		if (no_of_days_worked <= no_of_days_in_current_month) {
			this.current_basic_pay = this.per_day_pay * no_of_days_worked;
			System.out.println("Programmer");
			System.out.println("----------");
			generatePaySlip();
		} else {
			System.out.println("Sorry Please Enter Valid Days");
		}
	}

	protected void finalize() {
		input.close();
		System.exit(0);
	}
```

```
}
```

Programmer.java

```
package com.raja.oopslab.employee;

public class Programmer extends Employee {

	public Programmer() {
		super();
		computeProgrammerPay();

	}

	public void computeProgrammerPay() {
		System.out.println("Enter Basic pay of Programmer [enter -1  for Default [BP = 
30000]]:");
		this.basic_pay = input.nextInt();
		if (this.basic_pay == -1) {
			this.basic_pay = 30000;
			System.out.println("Default Pay Taken");
		}
		computeCurrentBasicPay("Programmer");
		generatePaySlip();
		displayPaySlip();
	}

}
```

Assistant Professor.java

```
package com.raja.oopslab.employee;

public class AssistantProfessor extends Employee {
	public AssistantProfessor() {
		super();
		computeAssistantProfessorPay();
	}

	public void computeAssistantProfessorPay() {
		System.out.println("Enter Basic pay of AssistantProfessor [enter -1  for Default [BP 
= 25000]]:");
		this.basic_pay = input.nextInt();
		if (this.basic_pay == -1) {
			this.basic_pay = 25000;
			System.out.println("Default Pay Taken");
		}
		computeCurrentBasicPay("AssistantProfessor");
		generatePaySlip();
		displayPaySlip();
	}
}
```

Associate Professor

```
package com.raja.oopslab.employee;

public class AssociateProfessor extends Employee {
	public AssociateProfessor() {
		super();
		computeAssociateProfessorPay();
	}

	public void computeAssociateProfessorPay() {
		System.out.println("Enter Basic pay of AssociateProfessor [enter -1  for Default [BP = 40000]]:");
		this.basic_pay = input.nextInt();
		if (this.basic_pay == -1) {
			this.basic_pay = 40000;
			System.out.println("Default Pay Taken");
		}

		computeCurrentBasicPay("AssociateProfessor");
		generatePaySlip();
		displayPaySlip();
	}
}
```

Professor

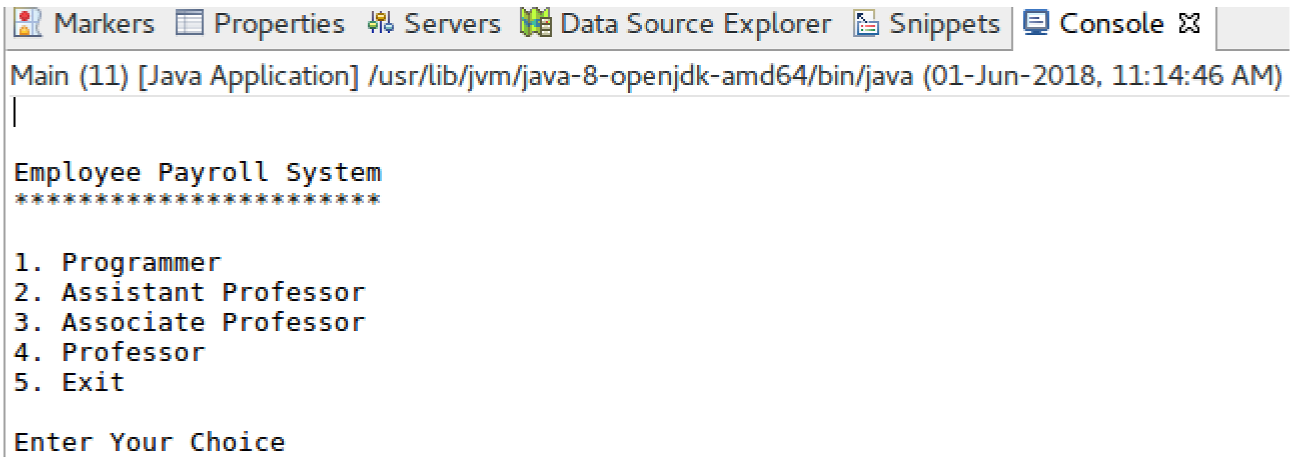
```
package com.raja.oopslab.employee;

public class Professor extends Employee {
	public Professor() {
		super();
		computeProfessorPay();
	}

	public void computeProfessorPay() {
		System.out.println("Enter Basic pay of Professor [enter -1  for Default [BP = 70000]]:");
		this.basic_pay = input.nextInt();
		if (this.basic_pay == -1) {
			this.basic_pay = 70000;
			System.out.println("Default Pay Taken");
		}
		computeCurrentBasicPay("Professor");
		generatePaySlip();
		displayPaySlip();
	}
}
```

Main.java

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.Scanner;

import com.raja.oopslab.employee.AssistantProfessor;
import com.raja.oopslab.employee.AssociateProfessor;
import com.raja.oopslab.employee.Programmer;
import com.raja.oopslab.employee.Professor;

public class Main {

        public static void main(String[] args) throws IOException {
                Programmer aProgrammer;
                AssistantProfessor aAssistantProfessor;
                AssociateProfessor aAssociateProfessor;
                Professor aProfessor;
                String choice;
                int n_choice = 0;
                Scanner userInput = new Scanner("System.in");

                while (n_choice != 5) {
                        System.out.println("\n\nEmployee Payroll System");
                        System.out.println("***********************\n");
                        System.out.println("1. Programmer\n2. Assistant Professor\n" + "3. Associate
Professor\n4. Professor\n"
                                                + "5. Exit\n\nEnter Your Choice");
                        choice = new BufferedReader(new
InputStreamReader(System.in)).readLine();
                        n_choice = Integer.parseInt(choice);
                        switch (n_choice) {
                        case 1:
                                System.out.println("Programmer Selected");
                                aProgrammer = new Programmer();
                                break;
                        case 2:
                                System.out.println("AssistantProfessor Selected");
                                aAssistantProfessor = new AssistantProfessor();
                                break;
                        case 3:
                                System.out.println("AssociateProfessor Selected");
                                aAssociateProfessor = new AssociateProfessor();
                                break;
                        case 4:
                                System.out.println("Professor Selected");
                                aProfessor = new Professor();
                        case 5:
                                System.out.println("Thank You !!!");
                                userInput.close();
                                break;
                        default:
                                System.out.println("Enter valid choice !!!");
                                break;
                        }
                }
        }}
```

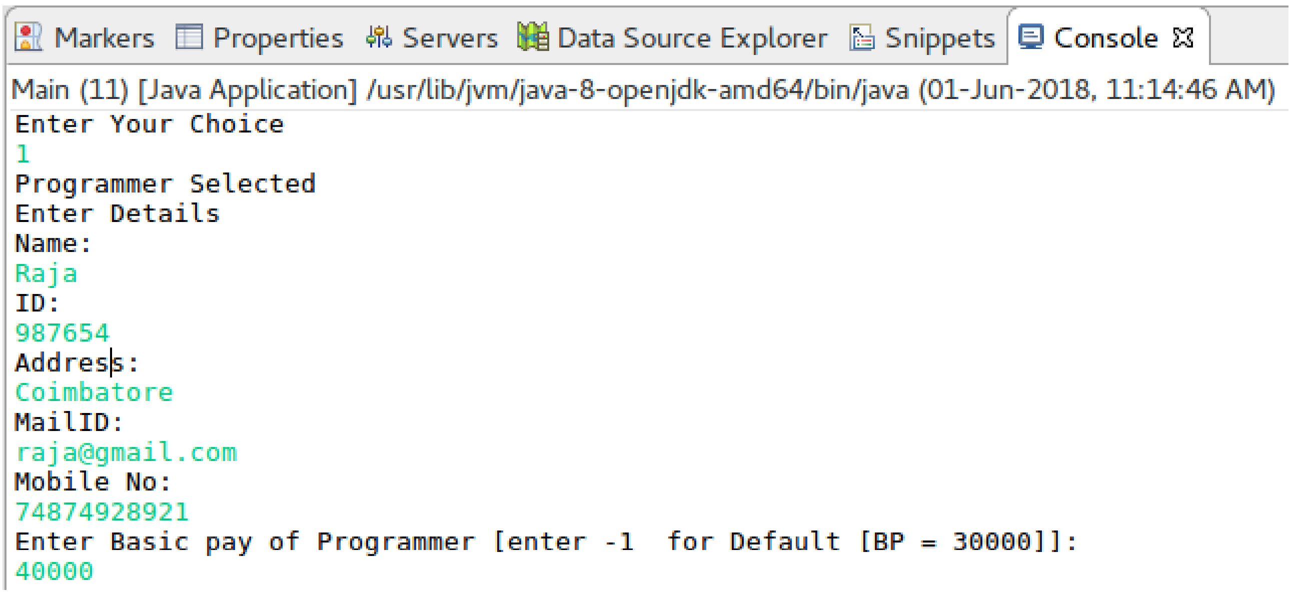
Output:

Choices



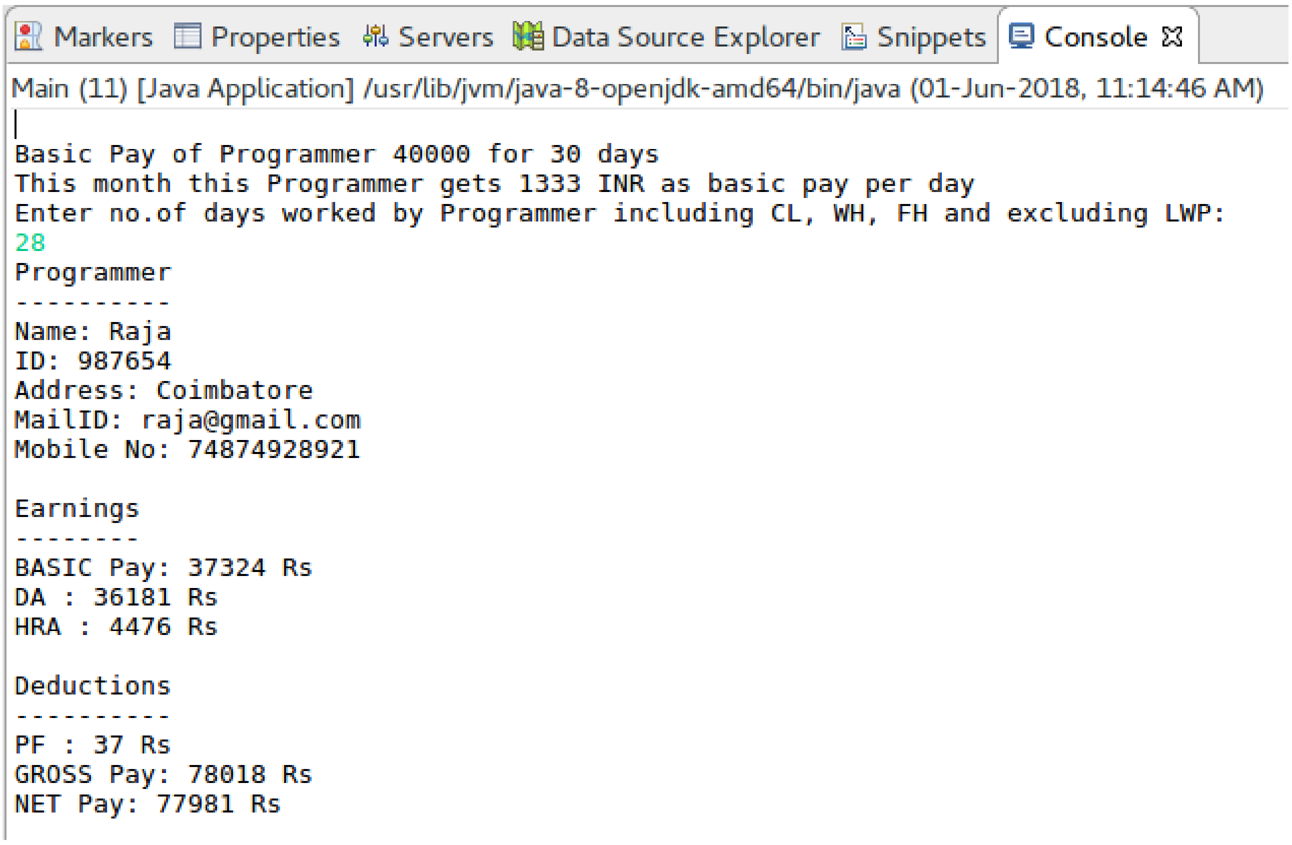
```
Main (11) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 11:14:46 AM)

Employee Payroll System
***********************

1. Programmer
2. Assistant Professor
3. Associate Professor
4. Professor
5. Exit

Enter Your Choice
```

Basic Details



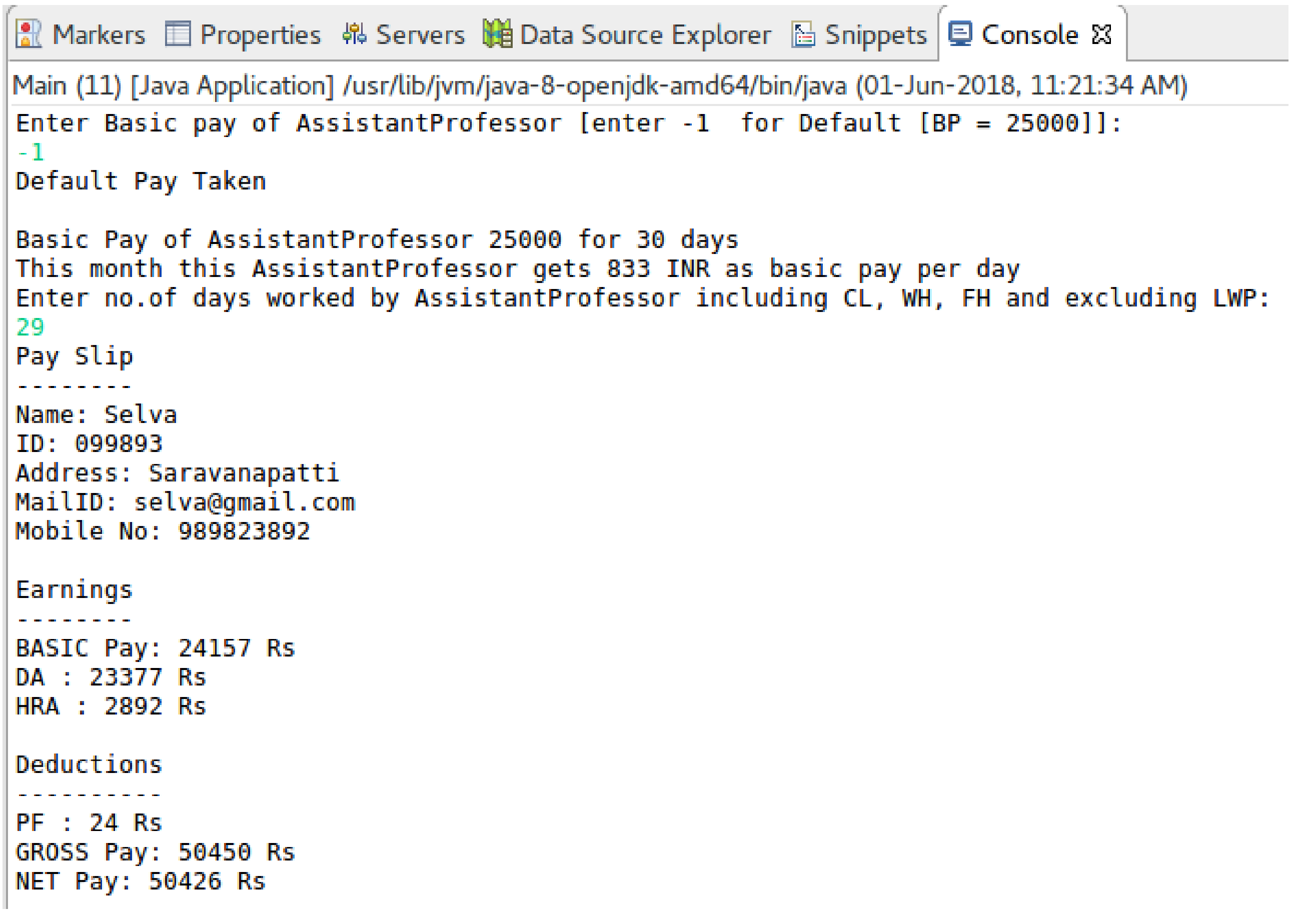
```
Main (11) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 11:14:46 AM)
Enter Your Choice
1
Programmer Selected
Enter Details
Name:
Raja
ID:
987654
Address:
Coimbatore
MailID:
raja@gmail.com
Mobile No:
74874928921
Enter Basic pay of Programmer [enter -1  for Default [BP = 30000]]:
40000
```

Programmer



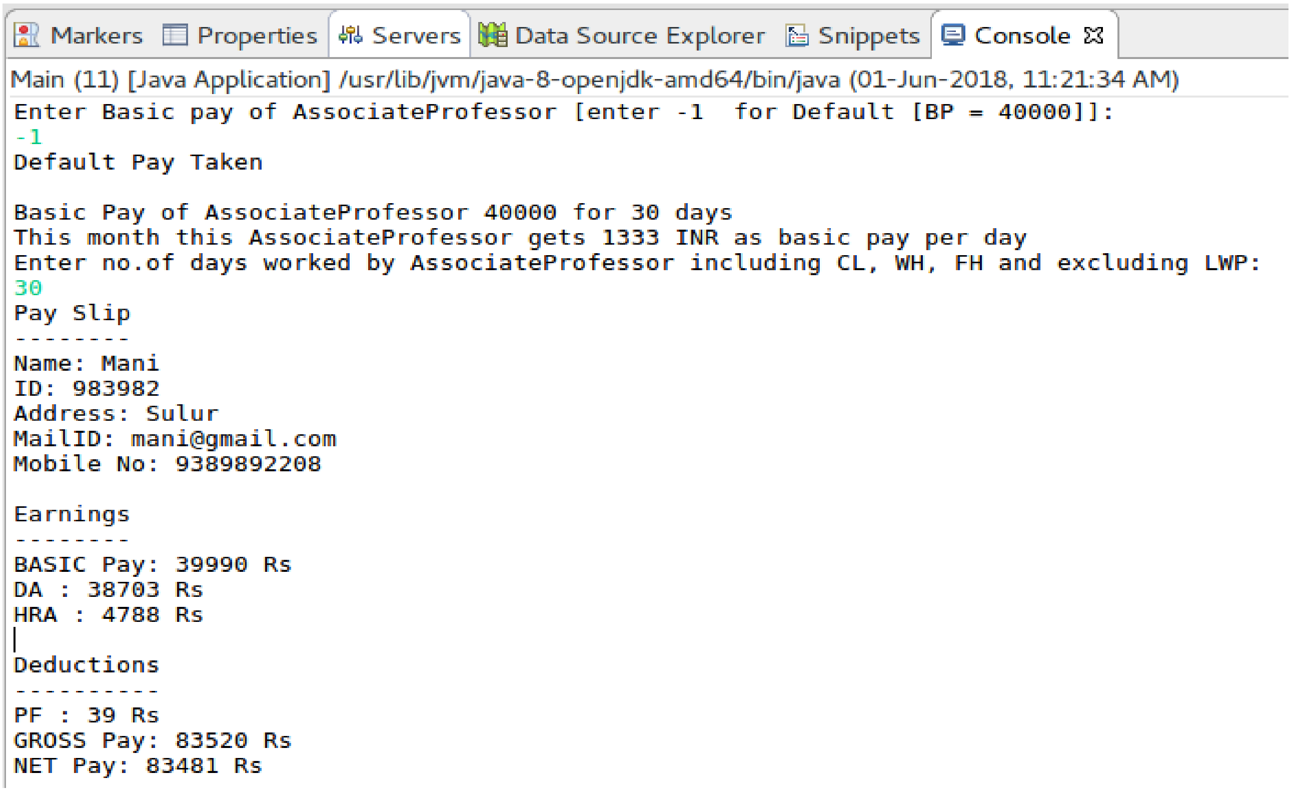
```
Main (11) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 11:14:46 AM)

Basic Pay of Programmer 40000 for 30 days
This month this Programmer gets 1333 INR as basic pay per day
Enter no.of days worked by Programmer including CL, WH, FH and excluding LWP:
28
Programmer
----------
Name: Raja
ID: 987654
Address: Coimbatore
MailID: raja@gmail.com
Mobile No: 74874928921

Earnings
--------
BASIC Pay: 37324 Rs
DA : 36181 Rs
HRA : 4476 Rs

Deductions
----------
PF : 37 Rs
GROSS Pay: 78018 Rs
NET Pay: 77981 Rs
```

Assistant Professor



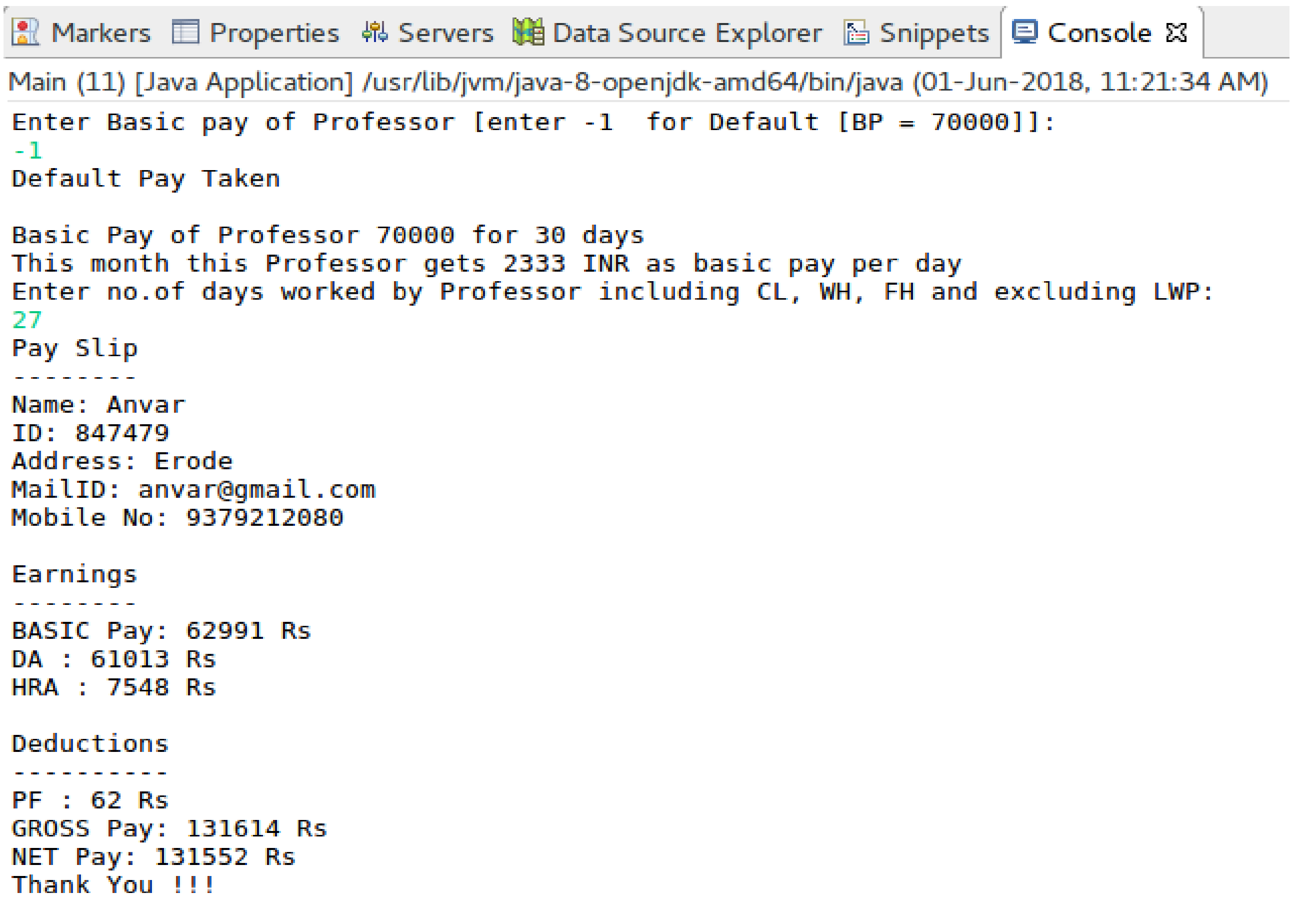
```
Main (11) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 11:21:34 AM)
Enter Basic pay of AssistantProfessor [enter -1  for Default [BP = 25000]]:
-1
Default Pay Taken

Basic Pay of AssistantProfessor 25000 for 30 days
This month this AssistantProfessor gets 833 INR as basic pay per day
Enter no.of days worked by AssistantProfessor including CL, WH, FH and excluding LWP:
29
Pay Slip
--------
Name: Selva
ID: 099893
Address: Saravanapatti
MailID: selva@gmail.com
Mobile No: 989823892

Earnings
--------
BASIC Pay: 24157 Rs
DA : 23377 Rs
HRA : 2892 Rs

Deductions
----------
PF : 24 Rs
GROSS Pay: 50450 Rs
NET Pay: 50426 Rs
```

Associate Professor



```
Main (11) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 11:21:34 AM)
Enter Basic pay of AssociateProfessor [enter -1  for Default [BP = 40000]]:
-1
Default Pay Taken

Basic Pay of AssociateProfessor 40000 for 30 days
This month this AssociateProfessor gets 1333 INR as basic pay per day
Enter no.of days worked by AssociateProfessor including CL, WH, FH and excluding LWP:
30
Pay Slip
--------
Name: Mani
ID: 983982
Address: Sulur
MailID: mani@gmail.com
Mobile No: 9389892208

Earnings
--------
BASIC Pay: 39990 Rs
DA : 38703 Rs
HRA : 4788 Rs

Deductions
----------
PF : 39 Rs
GROSS Pay: 83520 Rs
NET Pay: 83481 Rs
```

Professor



```
Main (11) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 11:21:34 AM)
Enter Basic pay of Professor [enter -1  for Default [BP = 70000]]:
-1
Default Pay Taken

Basic Pay of Professor 70000 for 30 days
This month this Professor gets 2333 INR as basic pay per day
Enter no.of days worked by Professor including CL, WH, FH and excluding LWP:
27
Pay Slip
--------
Name: Anvar
ID: 847479
Address: Erode
MailID: anvar@gmail.com
Mobile No: 9379212080

Earnings
--------
BASIC Pay: 62991 Rs
DA : 61013 Rs
HRA : 7548 Rs

Deductions
----------
PF : 62 Rs
GROSS Pay: 131614 Rs
NET Pay: 131552 Rs
Thank You !!!
```

Result:

The java console application for employee payroll system was developed and tested successfully.

Ex.No: 4	**Java Application for ADT Stack using Interfaces**
Date:	

Aim:

To create a Java console application using interface concepts of java for abstract data type stack. Stack operations must be controlled by exception handling techniques.

Algorithm:

```
Step 1    Start the process.
Step 2    Get the Stack's maximum limit from user.
Step 3    Create an array with the limit and initialize all the elements as -1 and initialize
          current_position with 0
Step 4    Prompt the user with choice for stack operations
          1. PUSH 2.POP 3.PEEK 4.DISPLAY 5.EXIT
Step 5    Get the choice from user and goto step 6
Step 6    If user selects to PUSH
          Step 6.1    Get the number from user to push.
          Step 6.2    Try to assign the value to array assuming that array index is current position.
          Step 6.3    If exception rises display message "Stack is full try to pop someting" and
                      goto step 9.
          Step 6.4    Else increment current position with the value 1 and display element pushed
                      index and goto step 9.
Step 7    If user selects to POP
          Step 7.1    Try to set previous position of current position to -1
          Step 7.2    If exception rises display message "Stack is empty try to push something"
                      and goto step 9.
          Step 7.3    Else decrement current position with the value 1 and display popped element
                      and goto step 9.
Step 8    If user selects to PEEK display element display current_position -1 element in the array
Step 9    If user selects to DISPLAY
          Step 9.1    Display array reversely and elements which is not equal to -1 and goto step 4
Step 10   Exit from the process.
Step 11   Stop the process.
```

Coding

StackOpeations.java [Interface]

```
package com.raja.oopslab.stackadt;

public interface StackOperations {

	boolean push(int number);

	boolean pop();

	void peek();

	void display();
}
```

CustomStack.java

```
package com.raja.oopslab.stackadt;

public class CustomStack implements StackOperations {
	int[] stack_array;
	int limit;
	int current_position = 0;

	public CustomStack(int limit) {
		this.limit = limit;
		stack_array = new int[limit];
		initStack();
	}

	public void initStack() {
		for (int i = 0; i < limit; i++)
			stack_array[i] = -1;
	}

	@Override
	public boolean push(int number) {

		try {

			stack_array[current_position] = number;
			current_position++;
			System.out.println("The element " + number + " pushed in the position " +
current_position);
			display();
			return true;
		}

		catch (ArrayIndexOutOfBoundsException e) {
			System.out.println("Sorry Stack Full Please do some POP's");
		}
		return false;

	}
	@Override
	public boolean pop() {
		int poped_element;
		try {
			poped_element = stack_array[current_position - 1];
			stack_array[current_position - 1] = -1;
			current_position--;
			System.out.println("Poped element is : " + poped_element);
			display();
			return true;
		} catch (ArrayIndexOutOfBoundsException e) {
			System.out.println("Sorry Stack is Empty try to do some push");
		}
		return false;
	}

	@Override
	public void display() {

		System.out.println("\nStack Display");
		System.out.println("*************\n");
		for (int i = limit - 1; i >= 0; i--)
			if (stack_array[i] != -1)
				System.out.println(stack_array[i]);
		System.out.println("\n*************");
	}

	@Override
	public void peek() {
		int peek_element = 0;
		peek_element = stack_array[current_position - 1];
		System.out.println("Peek Element of the Stack is " + peek_element);
	}

}
```

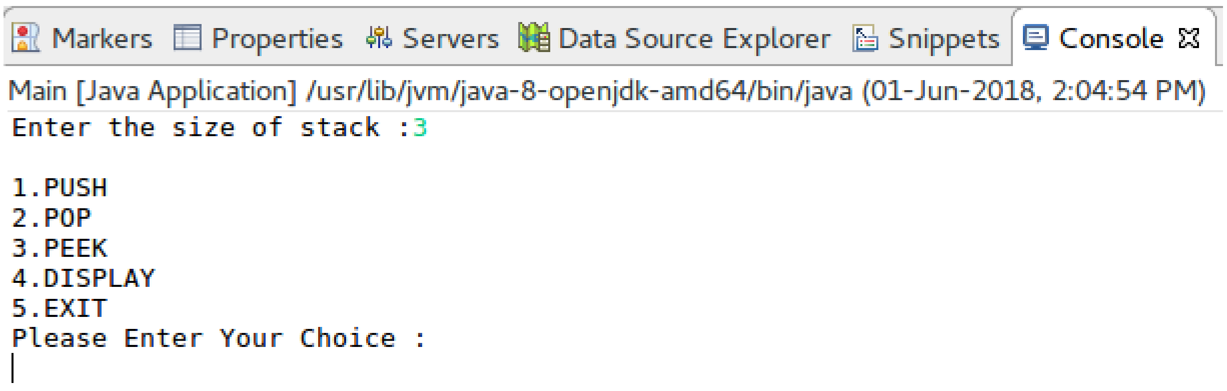
Main.java

```
import java.util.Scanner;
import com.raja.oopslab.stackadt.*;

public class Main {
	public static void main(String[] args) {
		Scanner input = new Scanner(System.in);
		System.out.print("Enter the size of stack :");
		CustomStack mystack = new CustomStack(input.nextInt());
		int choice = 0;
		while (choice != 5) {
			System.out.println("\n1.PUSH\n2.POP\n3.PEEK\n4.DISPLAY\n5.EXIT");
			System.out.println("Please Enter Your Choice : ");
			choice = input.nextInt();
			switch (choice) {
			case 1:
				System.out.println("Enter the Element to PUSH : ");
				mystack.push(input.nextInt());
				break;
			case 2:
				mystack.pop();
```

```
                        break;
                case 3:
                        mystack.peek();
                        break;
                case 4:
                        mystack.display();
                        break;
                case 5:
                        System.out.println("!!! Thank You !!!");
                        break;
                }

            }
            input.close();
            System.exit(0);
        }

}
```

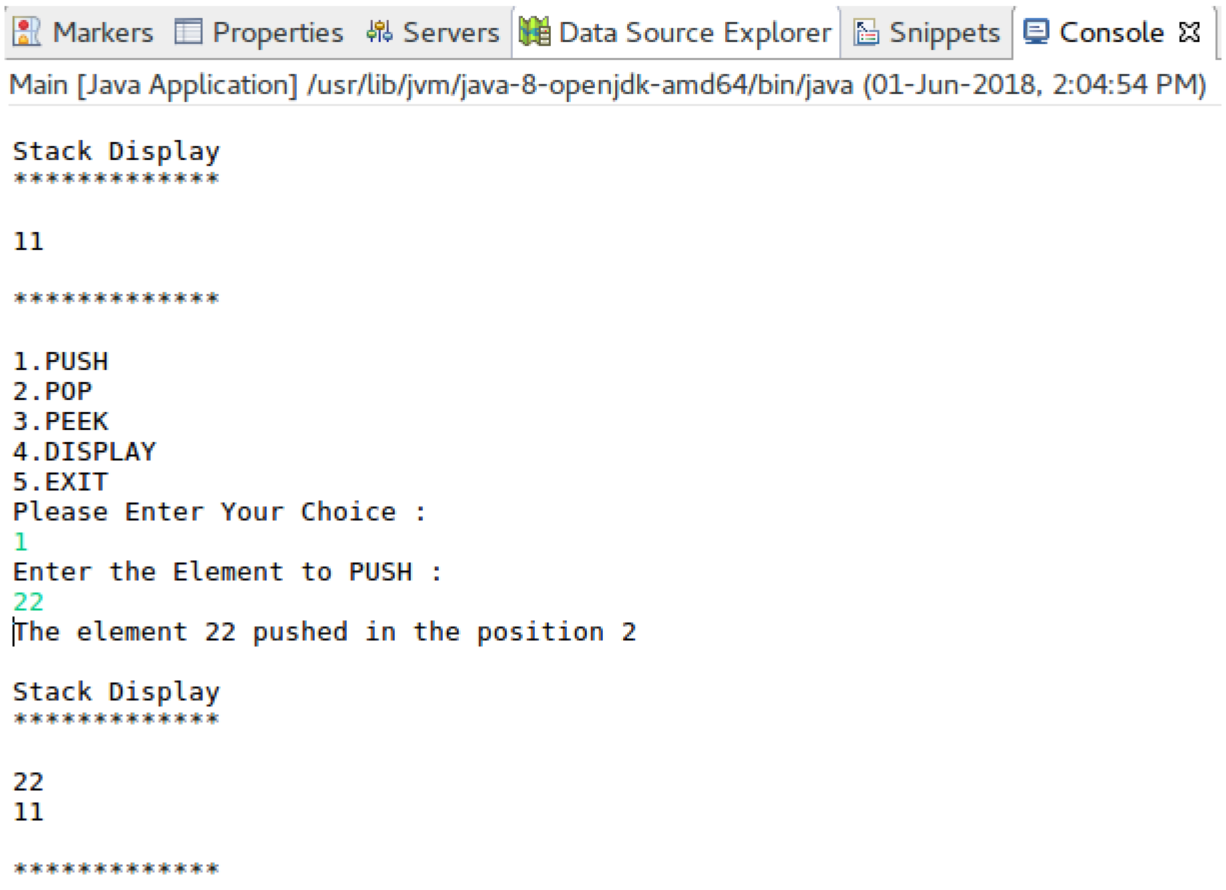
Output:

Choice:



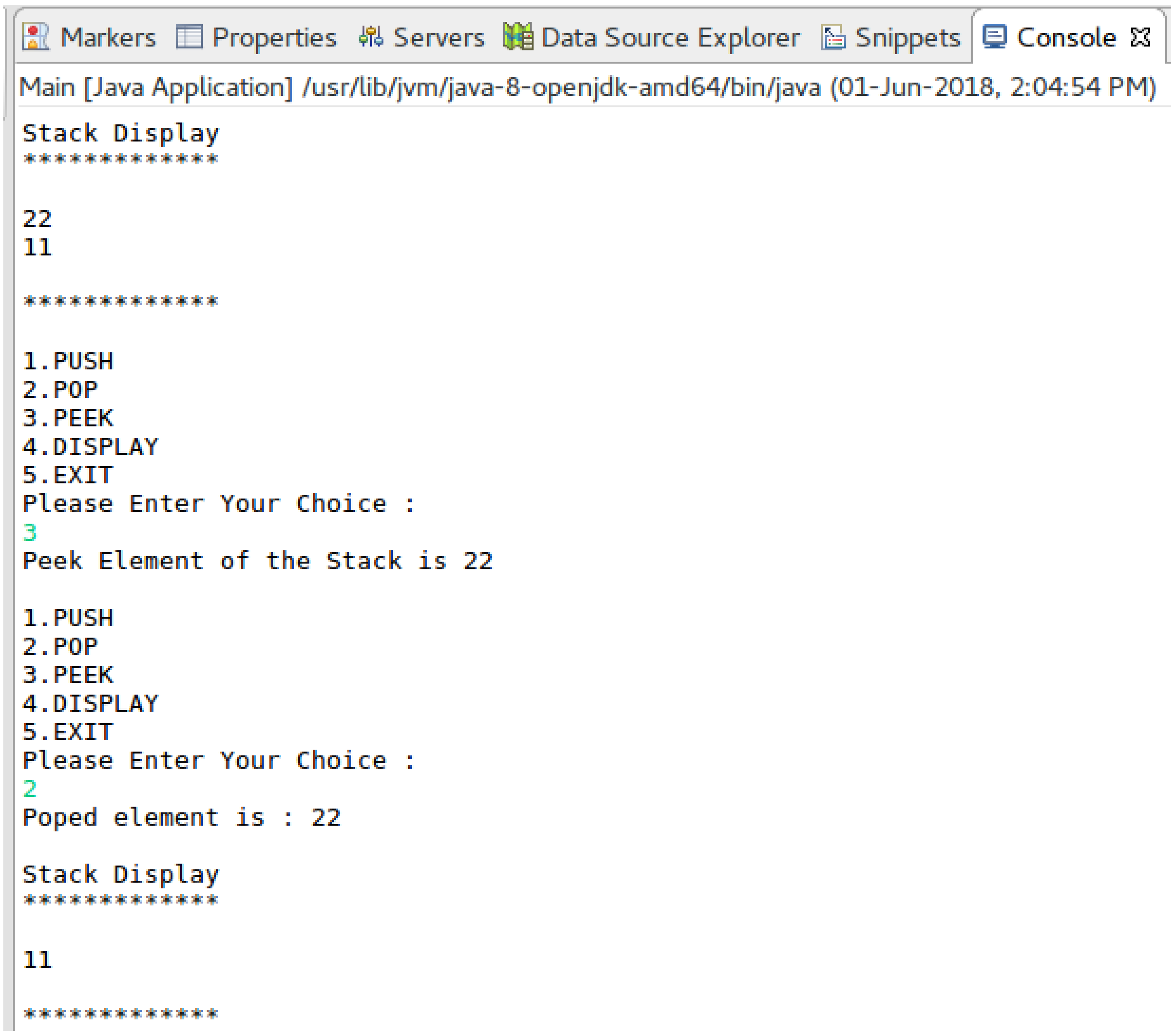
```
Main [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 2:04:54 PM)
Enter the size of stack :3

1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
Please Enter Your Choice :
```

Push Operation:



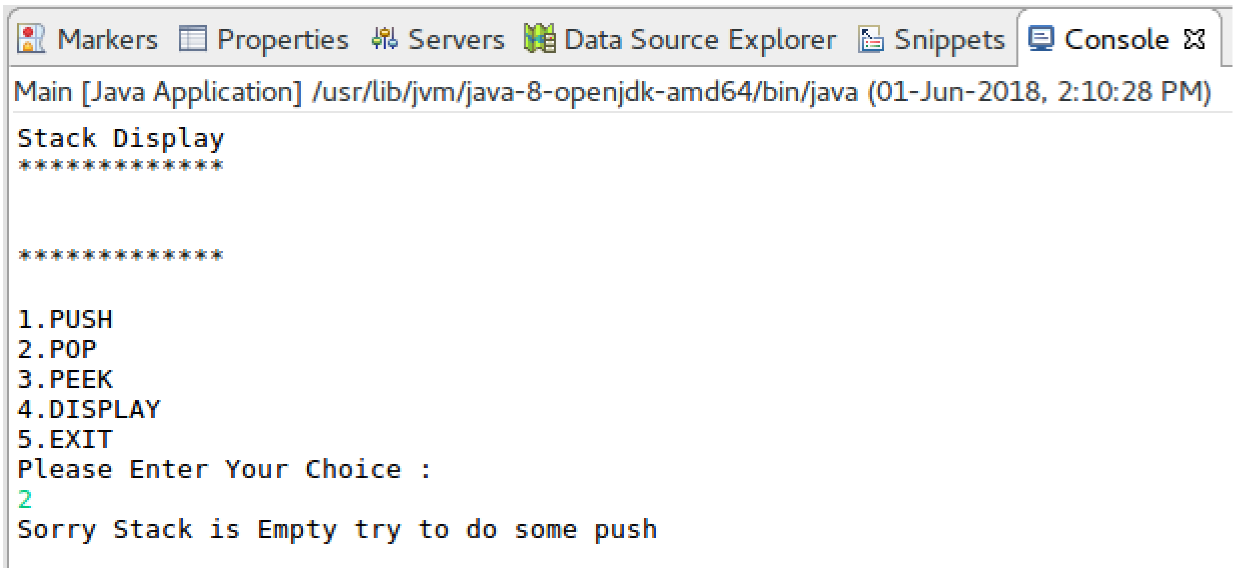
```
Main [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 2:04:54 PM)

Stack Display
*************

11

*************

1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
Please Enter Your Choice :
1
Enter the Element to PUSH :
22
The element 22 pushed in the position 2

Stack Display
*************

22
11

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

Pop and Peek Operation:



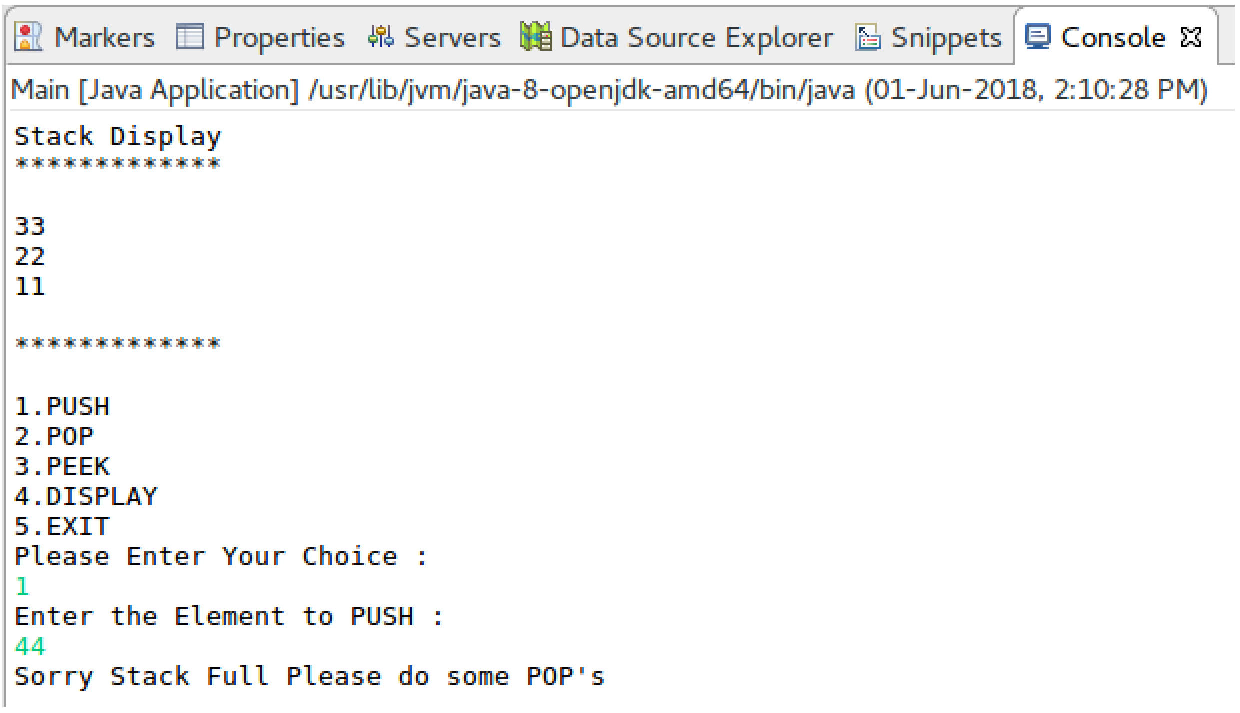
```
Main [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 2:04:54 PM)
Stack Display
*************

22
11

*************

1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
Please Enter Your Choice :
3
Peek Element of the Stack is 22

1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
Please Enter Your Choice :
2
Poped element is : 22

Stack Display
*************

11

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

Stack Empty Error:



```
Main [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 2:10:28 PM)
Stack Display
*************


*************

1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
Please Enter Your Choice :
2
Sorry Stack is Empty try to do some push
```

Stack Full Error:



```
Main [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 2:10:28 PM)
Stack Display
*************

33
22
11

*************

1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
Please Enter Your Choice :
1
Enter the Element to PUSH :
44
Sorry Stack Full Please do some POP's
```

Result:

The java console application for abstract data type stack using java interface concepts is developed and tested successfully.

Ex.No: 5	**Java Application for String Operations using ArrayList**
Date:	

Aim:

To create a Java console application for string list operations [Append, Insert at particular Index, Search a string in a list, Display all the strings that begins with a character] using ArrayList in java.

Algorithm:

```
Step 1    Start the Process
Step 2    Prompt user with List Operation Choices
          1. Append 2.  Insert at particular Index 3. Search a string in a list
          4. Display all the strings in a list 5. Display all the strings that begins with a character
          6. Exit
Step 3    If user selects option 1 then get the string from user and add at last position of the list.
Step 4    If user selects option 2
          Step 4.1    Get the string and position from the user
          Step 4.2    Check the position exceeds from list size
          Step 4.3    If exceeds then prompt an error “Sorry Position is not in the list limit” and
                      goto step 2.
          Step 4.4    Else create temporary list
          Step 4.5    Add all the elements of the list upto current position one by one.
          Step 4.6    Add current element to the list
          Step 4.7    Add remaining elements one by one.
          Step 4.8    Assign temporary list as source list and goto step 2
Step 5    If user selects option 3 then get the string to be searched from the user.
          Step 5.1    Check every string one by one for string match
          Step 5.2    If match found and display the string and position and goto step 2
          Step 5.3    Else prompt an error message “String not found in the list” and goto step 2
Step 6    If user selects option 4 the display all the strings one by one. After that goto step 2.
Step 7    If user selects option 5 then proceed following
          Step 7.1    Get the character from user
          Step 7.2    Get all the string one by one and match first letter with character entered by
                      the user
          Step 7.3    If character matches display the string
          Step 7.4    Else move to next string.
          Step 7.5    At the end goto step 2
Step 8    If user selects option 6 exit from process and stop processing
```

Coding:

StringList.java

```
package com.raja.oopslab.stringlist;

import java.util.ArrayList;

public class StringList {
        ArrayList<String> listOfStrings;

        public StringList() {
                listOfStrings = new ArrayList<String>();
        }

        public boolean addString(String aString) {
                boolean result = listOfStrings.add(aString);
                return result;
        }

        public void insertStringAt(int position, String aString) {
                int list_size = listOfStrings.size();
                int cur_pos = 0;
                ArrayList<String> templist = new ArrayList<String>();
                if (position > list_size)
                        System.out.println("Sorry Position is not in the list limit");
                else {
                        for (String elememt : listOfStrings) {

                                if (cur_pos == position)
                                        templist.add(aString);

                                templist.add(elememt);
                                cur_pos++;
                        }
                }
                listOfStrings = templist;
        }

        public void displayList() {
                int i = 0;
                for (String element : listOfStrings){
                        System.out.println(i+"."+element);
                        i++;
                }
        }

        public void searchString(String aString) {

                int cur_pos = 0;
                int foundAt = -1;


                for (String element : listOfStrings) {
                        if (element.equalsIgnoreCase(aString)) {
                                foundAt = cur_pos;
                                break;
                        } else
                        cur_pos++;
                }
                if (foundAt >= 0)
                        System.out.println("Your string " + aString + " found at position " +
cur_pos);
                else
                        System.out.println("String not found in the list");
        }

        public void displayStringsBeginWith(String aLetter) {
                System.out.println("The String's Begins with Letter ["+aLetter+"]\n");
                for (String element : listOfStrings) {
                        String firstLetter = String.valueOf(element.charAt(0));
                        if (aLetter.equals(firstLetter))
                                System.out.println(element);
                }
        }
}
```

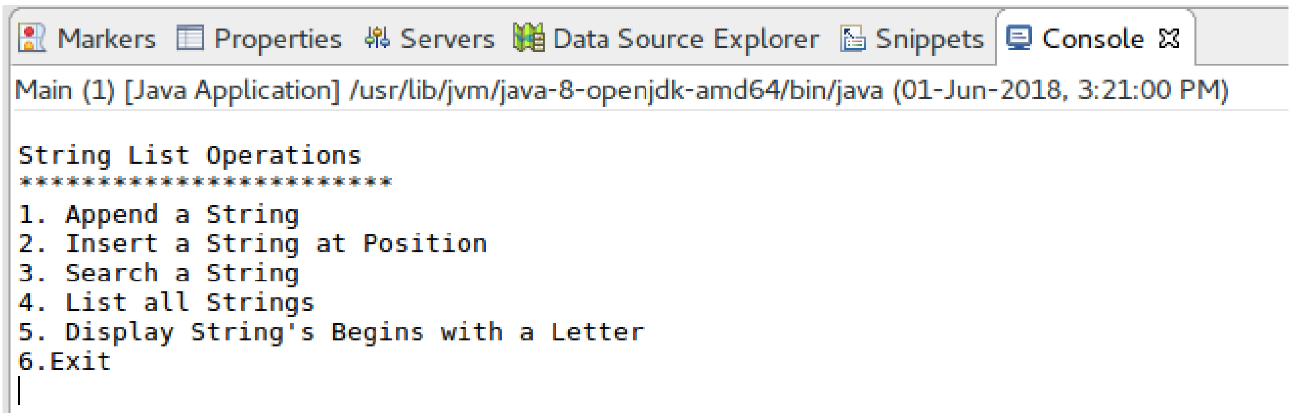
Main.java

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.Scanner;

import com.raja.oopslab.stringlist.StringList;

public class Main {
        public static void main(String[] args) {
                StringList stringList = new StringList();
                Scanner userInput = new Scanner(System.in);
                String aString = null;
                int choice = 0;
                int position = 0;
                while (choice != 6) {
                        System.out.println("\nString List Operations");
                        System.out.println("************************");
                        System.out.println(
                                        "1. Append a String\n"
                                        + "2. Insert a String at Position\n"
                                        + "3. Search a String\n"
                                        + "4. List all Strings\n"
                                        + "5. Display String's Begins with a Letter\n"
                                        + "6.Exit");

                        choice = userInput.nextInt();

                        switch (choice) {
                        case 1:
                                System.out.println("Enter a String: ");
                                stringList.addString(userInput.next());
                                break;
                        case 2:
                                System.out.println("Enter a String: ");
                                aString = userInput.next();
                                System.out.println("Enter a Position to Insert: ");
```

```
                                        position = userInput.nextInt();
                                        stringList.insertStringAt(position, aString);
                                        break;
                                case 3:
                                        System.out.println("Enter a String to Be Searched: ");
                                        aString = userInput.next();
                                        stringList.searchString(aString);
                                        break;
                                case 4:
                                        System.out.println("\nList Contains");
                                        System.out.println("-------------");
                                        stringList.displayList();
                                        break;
                                case 5:
                                        System.out.println("Enter a letter");
                                        stringList.displayStringsBeginWith(userInput.next());
                                        break;
                                case 6:
                                        System.out.println("\n\nThank You !!!");
                                        userInput.close();
                                        System.exit(0);
                                        break;
                                default:
                                        System.out.println("Please enter valid input");
                                        break;
                                }
                        }
                }
        }
```

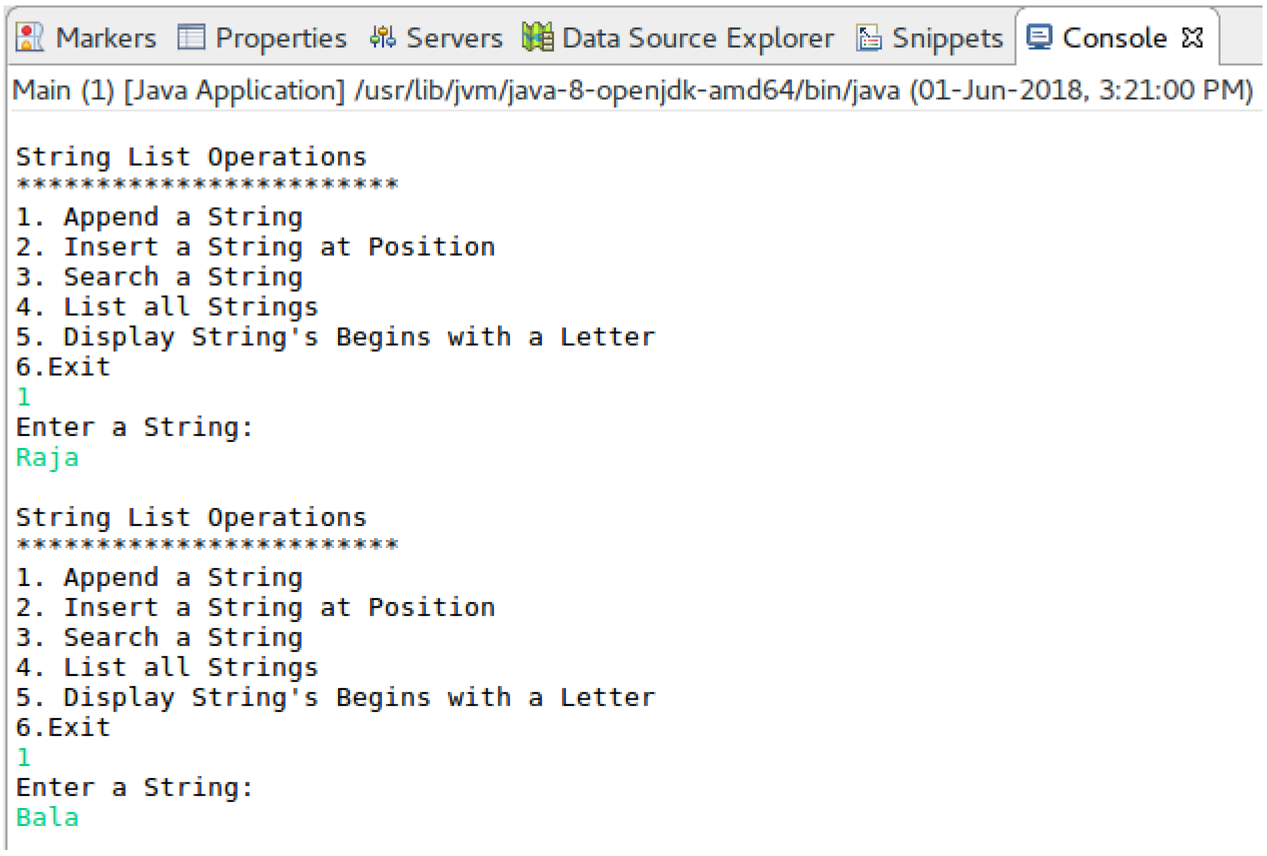
Output:

List Choices:



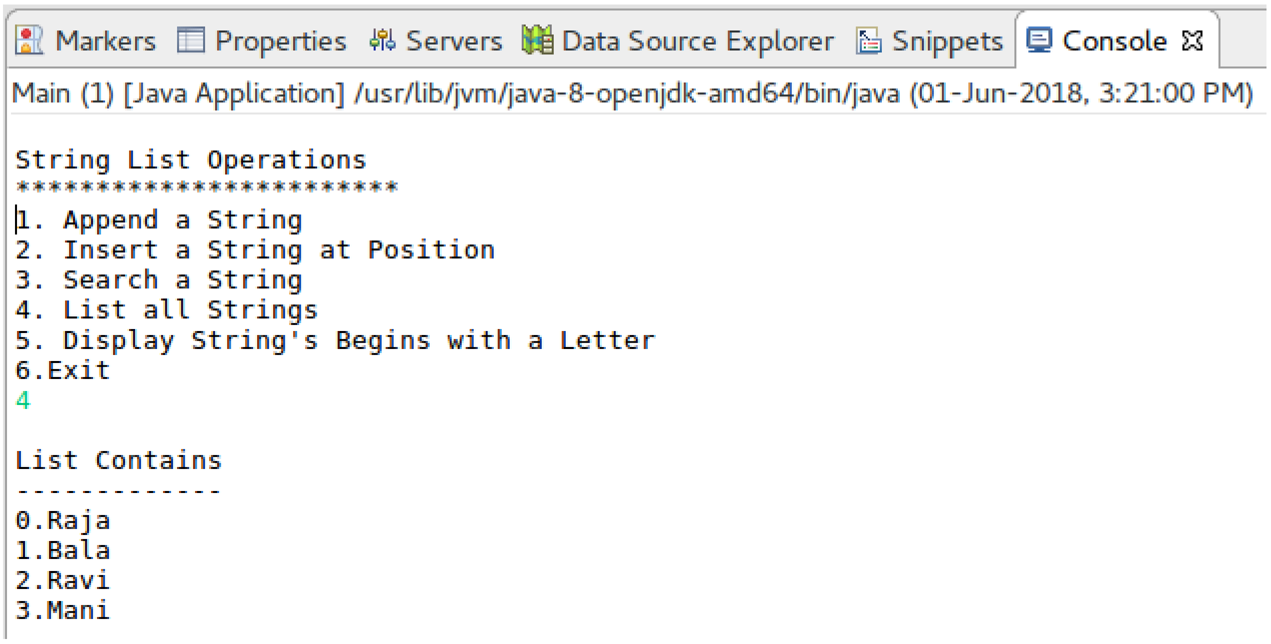
```
Main (1) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 3:21:00 PM)

String List Operations
*************************
1. Append a String
2. Insert a String at Position
3. Search a String
4. List all Strings
5. Display String's Begins with a Letter
6.Exit
```

Append a string:



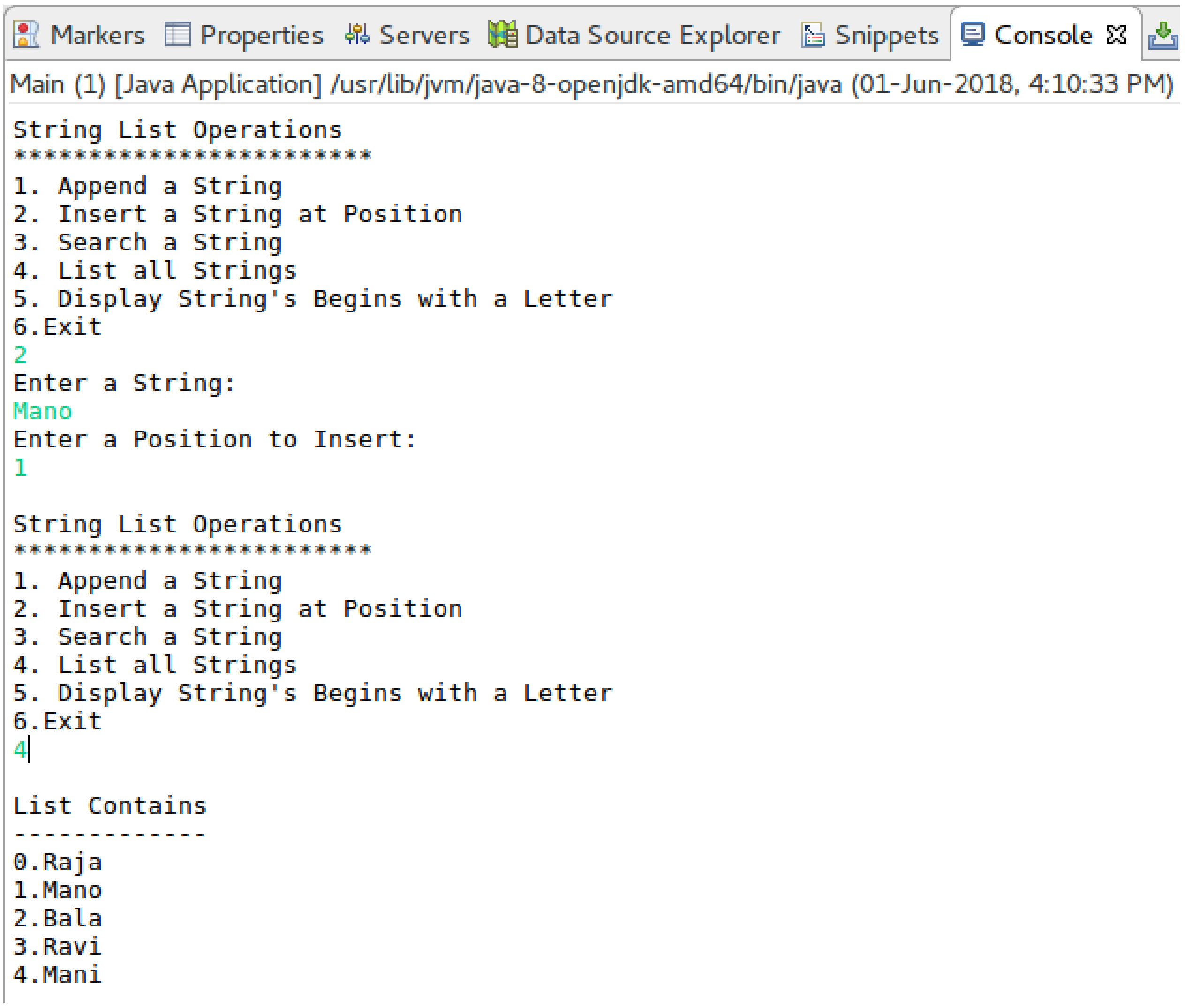
```
Main (1) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 3:21:00 PM)

String List Operations
*************************
1. Append a String
2. Insert a String at Position
3. Search a String
4. List all Strings
5. Display String's Begins with a Letter
6.Exit
1
Enter a String:
Raja

String List Operations
*************************
1. Append a String
2. Insert a String at Position
3. Search a String
4. List all Strings
5. Display String's Begins with a Letter
6.Exit
1
Enter a String:
Bala
```

Display List:



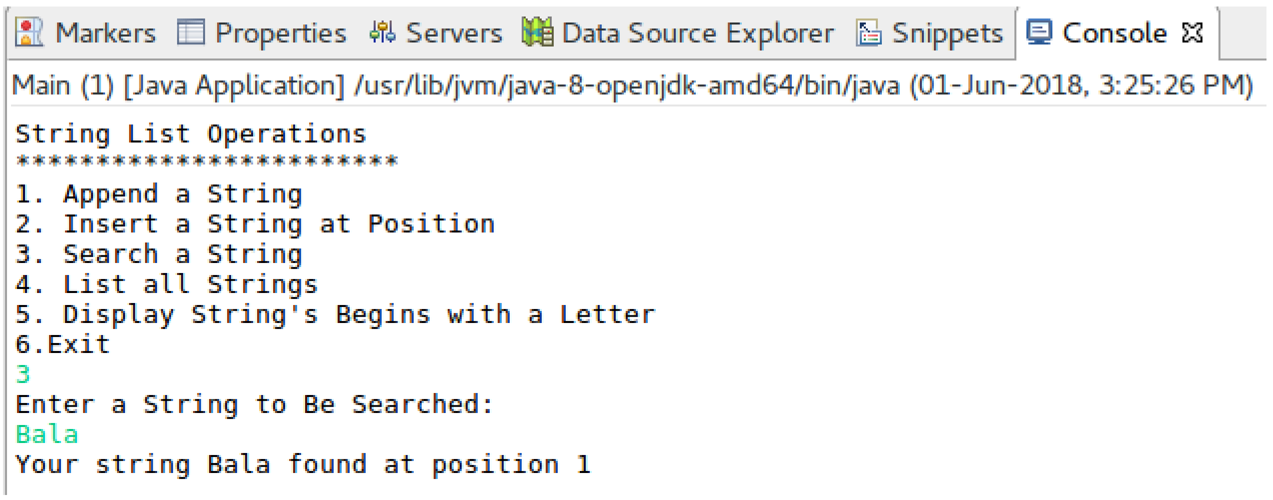
```
Main (1) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 3:21:00 PM)

String List Operations
************************
1. Append a String
2. Insert a String at Position
3. Search a String
4. List all Strings
5. Display String's Begins with a Letter
6.Exit
4

List Contains
-------------
0.Raja
1.Bala
2.Ravi
3.Mani
```

Insert At Position:



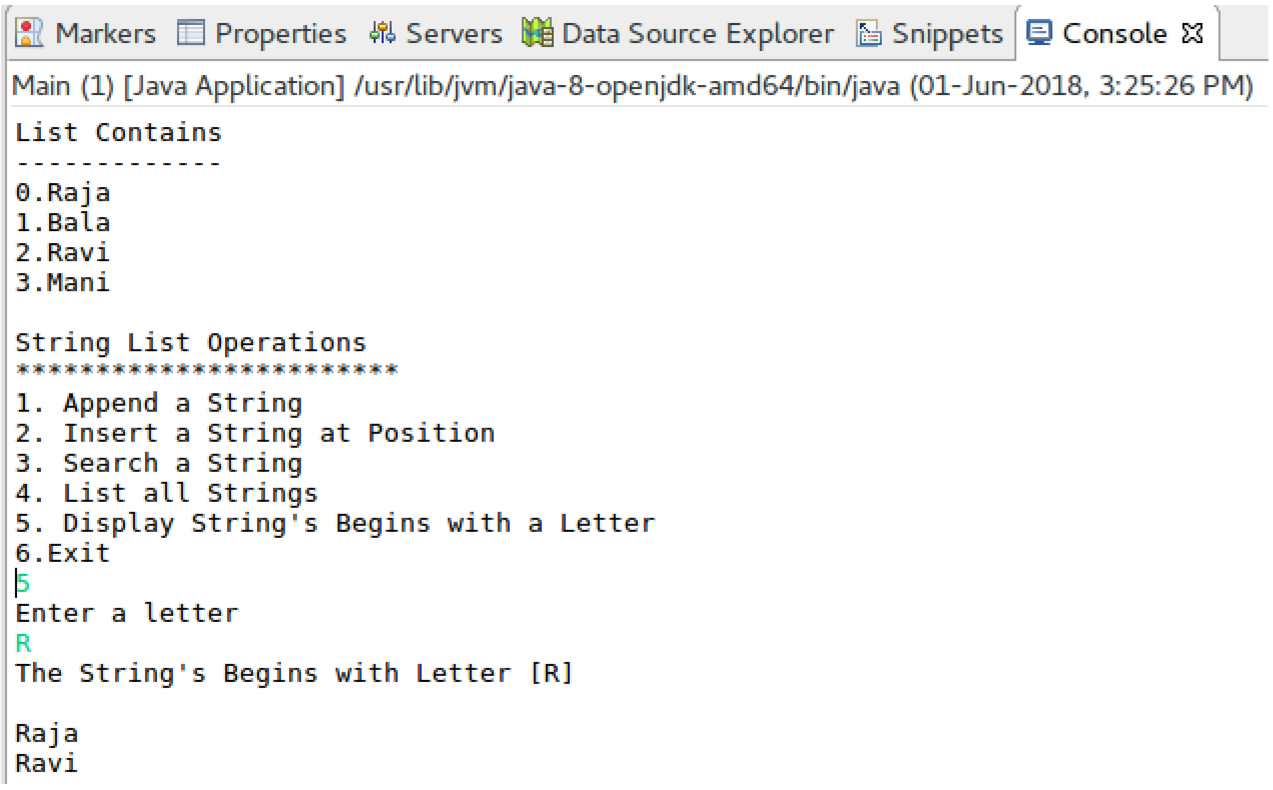
```
Main (1) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 4:10:33 PM)
String List Operations
************************
1. Append a String
2. Insert a String at Position
3. Search a String
4. List all Strings
5. Display String's Begins with a Letter
6.Exit
2
Enter a String:
Mano
Enter a Position to Insert:
1

String List Operations
************************
1. Append a String
2. Insert a String at Position
3. Search a String
4. List all Strings
5. Display String's Begins with a Letter
6.Exit
4

List Contains
-------------
0.Raja
1.Mano
2.Bala
3.Ravi
4.Mani
```

Search a String:



```
Main (1) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 3:25:26 PM)
String List Operations
************************
1. Append a String
2. Insert a String at Position
3. Search a String
4. List all Strings
5. Display String's Begins with a Letter
6.Exit
3
Enter a String to Be Searched:
Bala
Your string Bala found at position 1
```

List the string's that begins with a letter "R":



```
Main (1) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (01-Jun-2018, 3:25:26 PM)
List Contains
-------------
0.Raja
1.Bala
2.Ravi
3.Mani

String List Operations
************************
1. Append a String
2. Insert a String at Position
3. Search a String
4. List all Strings
5. Display String's Begins with a Letter
6.Exit
5
Enter a letter
R
The String's Begins with Letter [R]

Raja
Ravi
```

Result:

The java console application for string list operations was developed and tested successfully.

Ex.No: 6	**Java Application to Find the Area of different Shapes**
Date:	

Aim:

To create a Java console application to find the area of different shapes using abstract class concept in java.

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 5    If user selects Circle
          Step 5.1    Get the Radius of the Circle
          Step 5.2    Compute Area = 3.14 *  Radius *  Radius
          Step 5.3    Display Area
Step 6    If user selects exit end the process
Step 7    Stop the Process
```

Coding:

Shape.java

```
package com.raja.oopslanb.shapes;

public abstract class Shape {
	double length = 0.0;
	double hight = 0.0;
	public abstract void printArea();
}
```

Rectangle.java

```
package com.raja.oopslanb.shapes;

import java.util.Scanner;

public class Rectangle extends Shape {
	double area = 0.0;

	@Override
	public void printArea() {
		System.out.println("\nRectangle");
		System.out.println("---------\n");
		Scanner input = new Scanner(System.in);
		System.out.println("Enter Length of Rectangle : ");
		this.length = input.nextDouble();
		System.out.println("Enter Breadth of Rectangle : ");
		this.hight = input.nextDouble();
		this.area = this.length * this.hight;
		System.out.println("Area of the Rectangle is : " + this.area);
	}

}
```

Triangle.java

```
package com.raja.oopslanb.shapes;

import java.util.Scanner;

public class Triangle extends Shape {
	double area = 0.0;

	@Override
	public void printArea() {
		System.out.println("\nTriangle");
		System.out.println("---------\n");
		Scanner input = new Scanner(System.in);
		System.out.println("Enter Length of Triangle : ");
		this.length = input.nextDouble();
		System.out.println("Enter Hight of Triangle : ");
		this.hight = input.nextDouble();
		this.area = 0.5 * this.length * this.hight;
		System.out.println("Area of the Triangle is : " + this.area);
	}
}
```

Circle.java

```
package com.raja.oopslanb.shapes;

import java.util.Scanner;

public class Circle extends Shape {
	double area = 0.0;

	@Override
	public void printArea() {
		System.out.println("\nCircle");
		System.out.println("-------\n");
		Scanner input = new Scanner(System.in);
		System.out.println("Enter Radius of Circle : ");
		this.length = input.nextDouble();
		this.area = Math.PI * this.length * this.length;
		System.out.println("Area of the Circle is : " + this.area);
	}

}
```

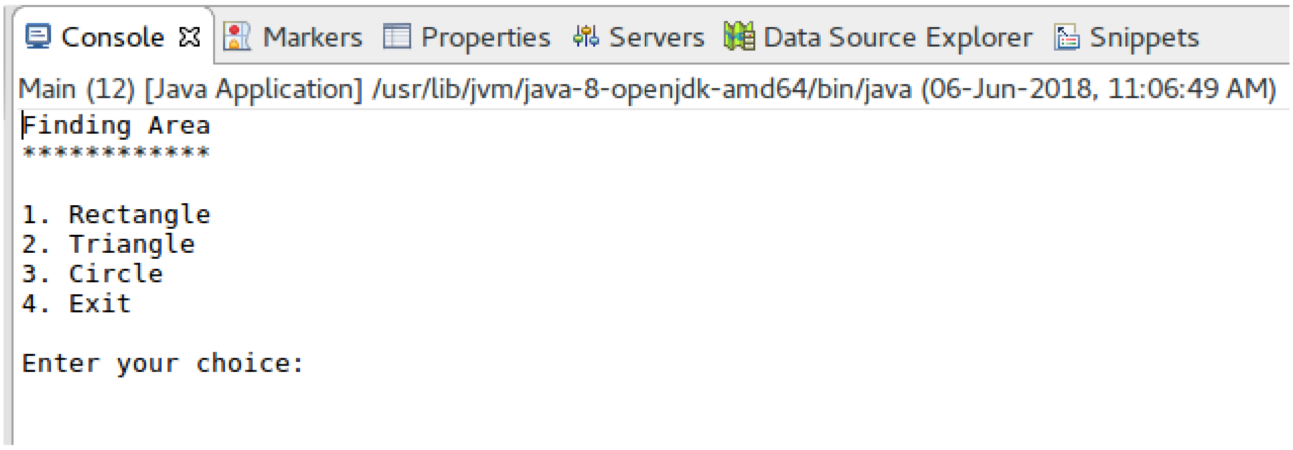
Main.java

```
import com.raja.oopslanb.shapes.Rectangle;
import com.raja.oopslanb.shapes.Shape;
import com.raja.oopslanb.shapes.Triangle;

import java.util.Scanner;

import com.raja.oopslanb.shapes.Circle;

public class Main {
	public static void main(String[] args) {
		Scanner userInput = new Scanner(System.in);
		int choice = 0;
		do {
			System.out.println("Finding Area");
			System.out.println("************");
			System.out.println(
						"\n1. Rectangle" + "\n2. Triangle" + "\n3. Circle" + "\n4. Exit"
+ "\n\nEnter your choice: ");
			choice = userInput.nextInt();
			switch (choice) {
			case 1:
				Shape rt = new Rectangle();
				rt.printArea();
				break;
			case 2:
				Shape tr = new Triangle();
				tr.printArea();
				break;
			case 3:
				Shape cr = new Circle();
```

```
					cr.printArea();
					break;
				case 4:
					System.out.println("\n\nThank You !!!");
					userInput.close();
					break;
				default:
					System.out.println("Please enter valid input");
					break;
				}
			} while (choice != 4);
		}
	}
```

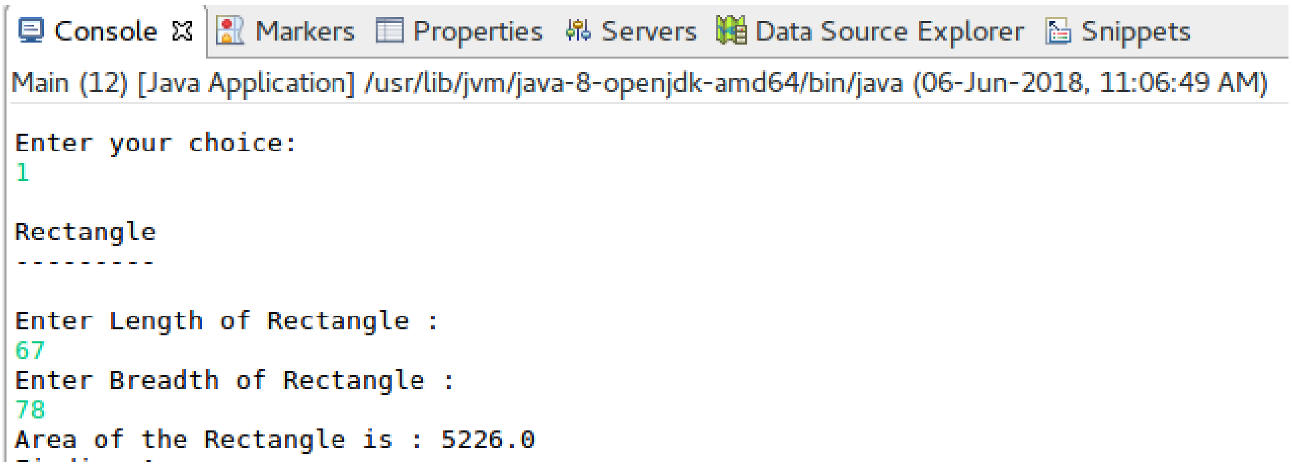
Output:

Choice



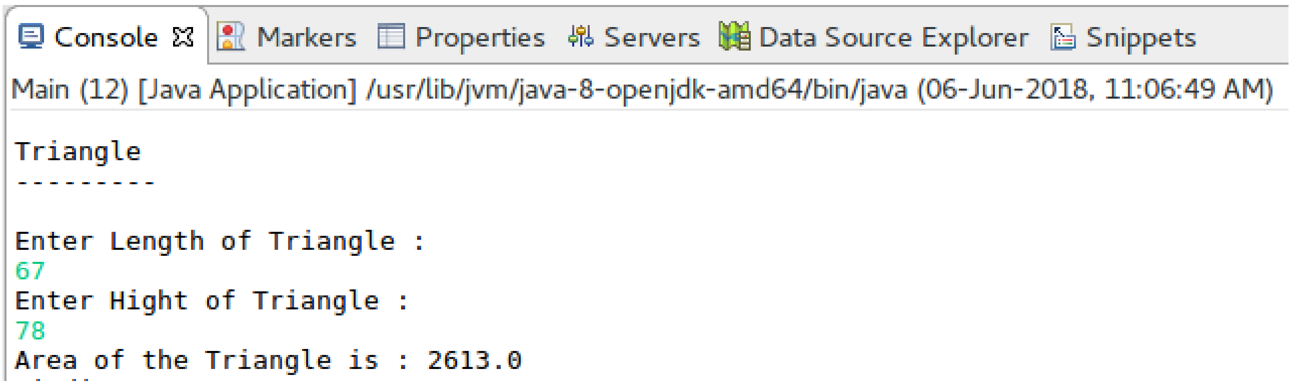
```
Main (12) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06-Jun-2018, 11:06:49 AM)
Finding Area
************

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

Enter your choice:
```

Rectangle



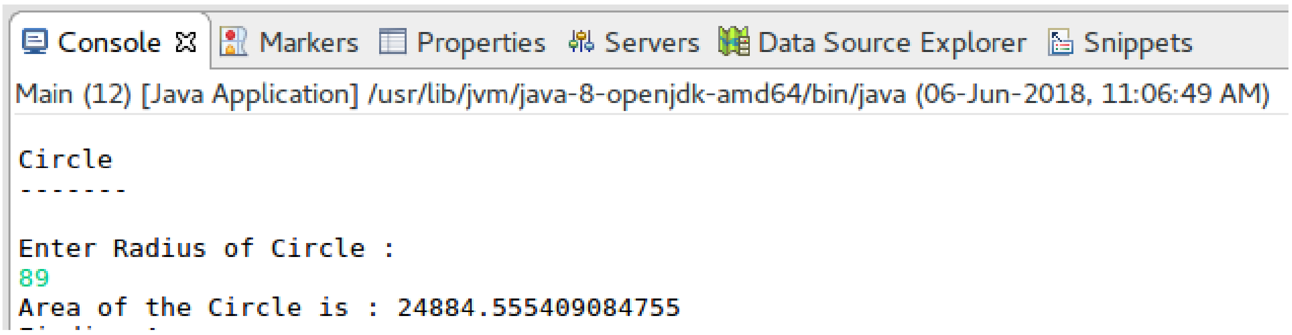
```
Main (12) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06-Jun-2018, 11:06:49 AM)

Enter your choice:
1

Rectangle
---------

Enter Length of Rectangle :
67
Enter Breadth of Rectangle :
78
Area of the Rectangle is : 5226.0
```

Triangle



```
Main (12) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06-Jun-2018, 11:06:49 AM)

Triangle
---------

Enter Length of Triangle :
67
Enter Hight of Triangle :
78
Area of the Triangle is : 2613.0
```

Circle



```
Main (12) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06-Jun-2018, 11:06:49 AM)

Circle
-------

Enter Radius of Circle :
89
Area of the Circle is : 24884.555409084755
```

Result:

The Java console application to find the area of different shapes using abstract class concept in java was developed and tested successfully.

Ex.No: 7	**Bank Transaction System with User Exceptions**
Date:	

Aim:

To create a Java console application for Banking transaction system that helps the users to do their credit and debit transactions and it rises user defined exception while encountering errors in transaction and also it should be solved using exception handling techniques.

Algorithm:

```
Step 1    Start the Process
Step 2    Prompt user with List Operation Choices
          1. Add Money 2. Get Money 3. Details 4. Exit
          Get the choice from user.
Step 3    If user selects Add Money
          Step 3.1    Get the amount to be added to balance from the user
          Step 3.2    If the amount is less than 0 then throw Invalid Credit Exception and goto step
                      3.4
          Step 3.3    Else add amount to balance and goto step 2
          Step 3.4    Prompt user with "Enter valid credit amount" and goto step 3.1
Step 4    If user selects Get Money
          Step 4.1    Get the amount to be debited to balance from the user
          Step 4.2    If the amount is greater than existing balance then throw Invalid Debit
                      Exception and goto step 4.4
          Step 4.3    Else deduct amount from balance and goto step 2
          Step 4.4    Prompt user with "Enter valid debit amount" and goto step 4.1
Step 5    If user selects Details then display Customer Details [Name, Account Number, Current
          Balance]
Step 6    If user wants to exit display "Thank You !!!" and end process
Step 7    Stop the Process
```

Coding:

Customer.java

```
package com.raja.oopslab.exception.bank;

import java.util.Scanner;

public class Customer {
	String name;
	int accNo;
	int balance;

	public Customer(String name, int accNo) {
		this.name = name;
		this.accNo = accNo;
		this.balance = 0;
	}

	public void creditTransaction(int amount) {
		Scanner input = new Scanner(System.in);
		try {
			if (amount < 0)
				throw new InvalidCredit();

			else
				balance = balance + amount;
		} catch (InvalidCredit e) {
			amount = input.nextInt();
			creditTransaction(amount);
		}

	}

	public void debitTransaction(int amount) {
		Scanner input = new Scanner(System.in);
		try {
			if (amount > balance)
				throw new InvalidDebit();

			else
				balance = balance - amount;
		} catch (InvalidDebit e) {
			amount = input.nextInt();
			debitTransaction(amount);
		}

	}

	public void displayDetails(){
		System.out.println("Customer Details");
		System.out.println("****************");
		System.out.println("Customer Name : "+this.name);
		System.out.println("Customer AccNo : "+this.accNo);
		System.out.println("Customer Current Balance : "+this.balance);
	}

}
```

InvalidCredit.java

```
package com.raja.oopslab.exception.bank;

public class InvalidCredit extends Exception {
	public InvalidCredit() {
		System.out.print("Please enter valid credit amount");
	}
}
```

InvalidDebit.java

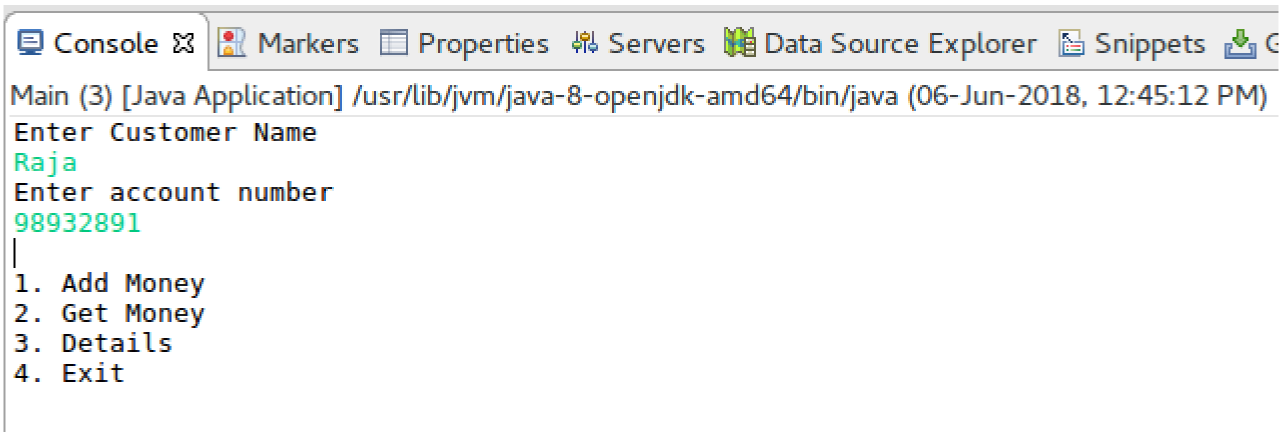
```
package com.raja.oopslab.exception.bank;

public class InvalidDebit extends Exception {
	public InvalidDebit() {
		System.out.print("Please enter valid debit amount");
	}
}
```

Main.java

```
import java.util.Scanner;
import com.raja.oopslab.exception.bank.*;
public class Main {
	public static void main(String[] args) {
		Scanner input = new Scanner(System.in);
		String name;
		int acc_no;
		System.out.println("Enter Customer Name");
		name = input.next();
		System.out.println("Enter account number");
		acc_no = input.nextInt();
		Customer aCustomer = new Customer(name, acc_no);
		int choice = 0;
		while(choice != 4){
			System.out.println("\n1. Add Money\n2. Get Money\n3. Details\n4. Exit");
			choice = input.nextInt();
			switch(choice){
			case 1:
				System.out.println("Enter the amount");
				aCustomer.creditTransaction(input.nextInt());
				break;
			case 2:
				System.out.println("Enter the amount");
				aCustomer.debitTransaction(input.nextInt());
				break;
			case 3:
				aCustomer.displayDetails();
				break;
			case 4:
				System.out.println("Thank You !!!");
				break;
			}
		}
	}
}
```

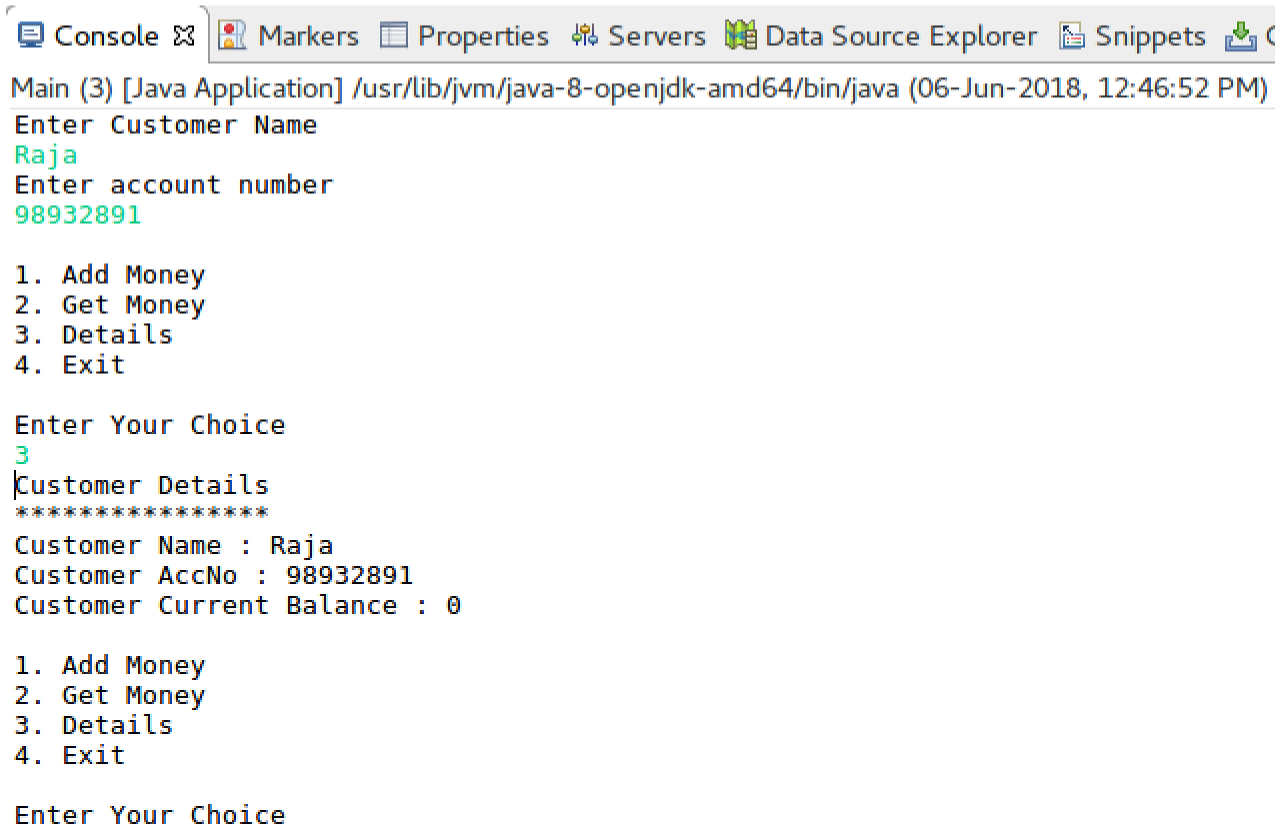
Output:

Choice:



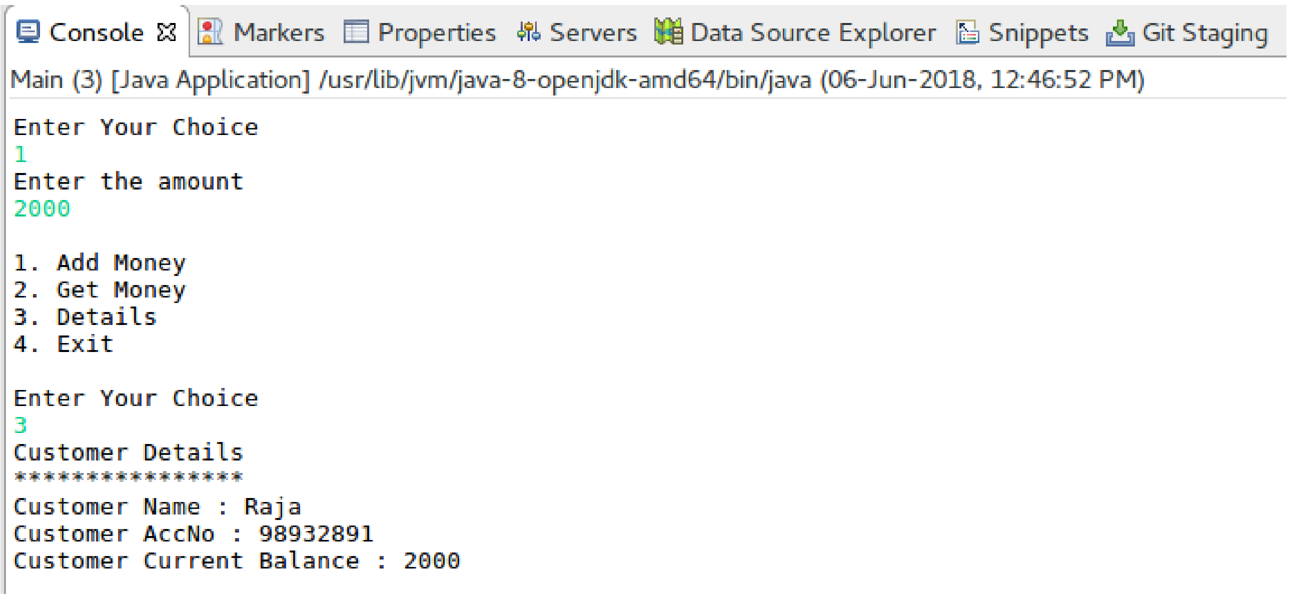
```
Main (3) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06-Jun-2018, 12:45:12 PM)
Enter Customer Name
Raja
Enter account number
98932891

1. Add Money
2. Get Money
3. Details
4. Exit
```

Display Details:



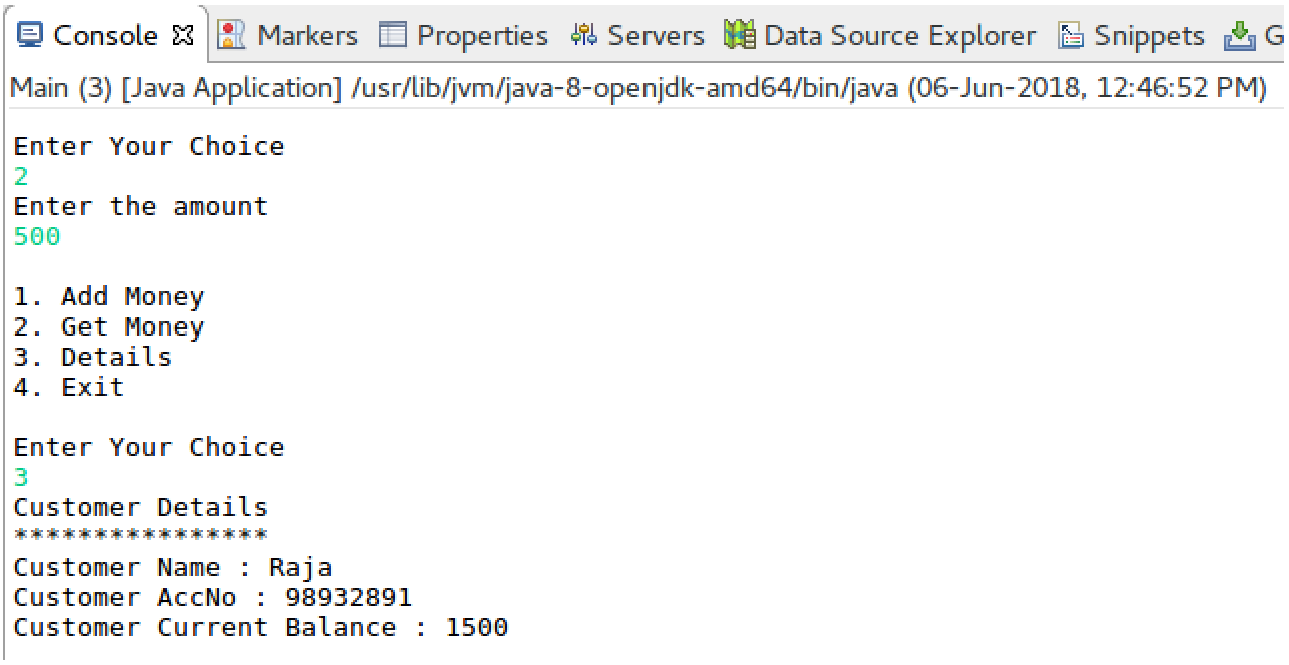
```
Main (3) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06-Jun-2018, 12:46:52 PM)
Enter Customer Name
Raja
Enter account number
98932891

1. Add Money
2. Get Money
3. Details
4. Exit

Enter Your Choice
3
Customer Details
****************
Customer Name : Raja
Customer AccNo : 98932891
Customer Current Balance : 0

1. Add Money
2. Get Money
3. Details
4. Exit

Enter Your Choice
```

Add Money:



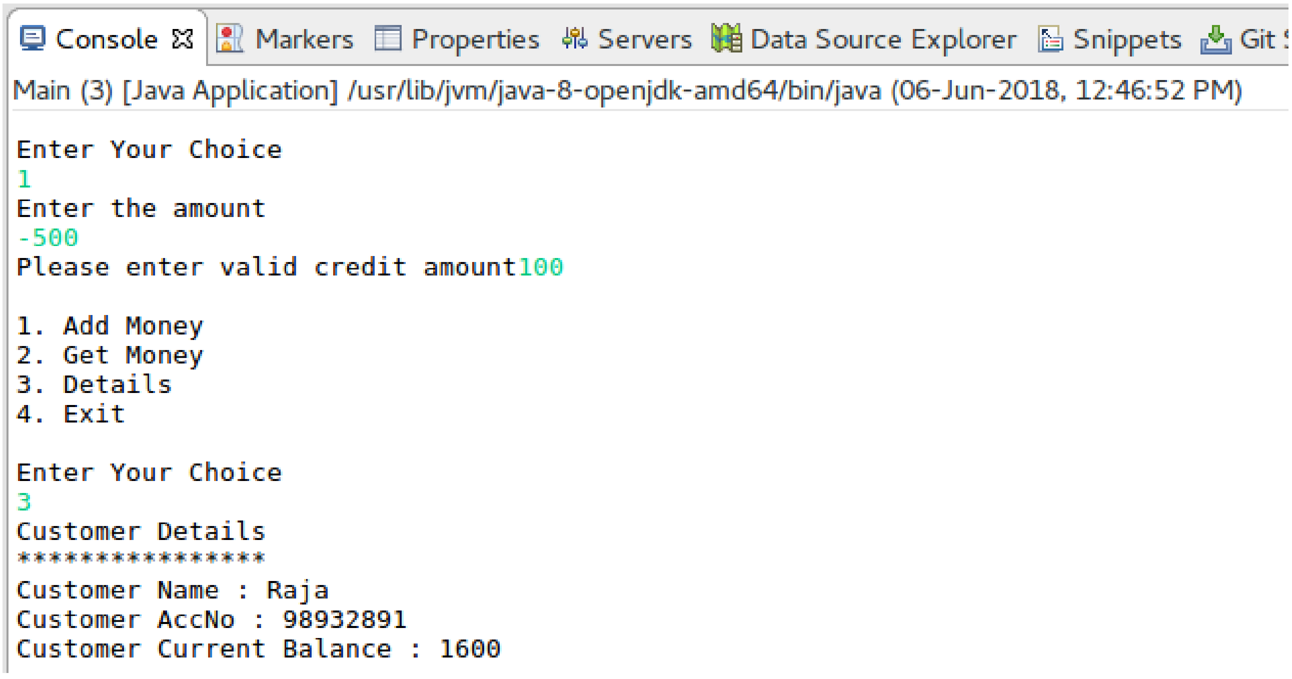
```
Main (3) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06-Jun-2018, 12:46:52 PM)
Enter Your Choice
1
Enter the amount
2000

1. Add Money
2. Get Money
3. Details
4. Exit

Enter Your Choice
3
Customer Details
****************
Customer Name : Raja
Customer AccNo : 98932891
Customer Current Balance : 2000
```

Get Money:



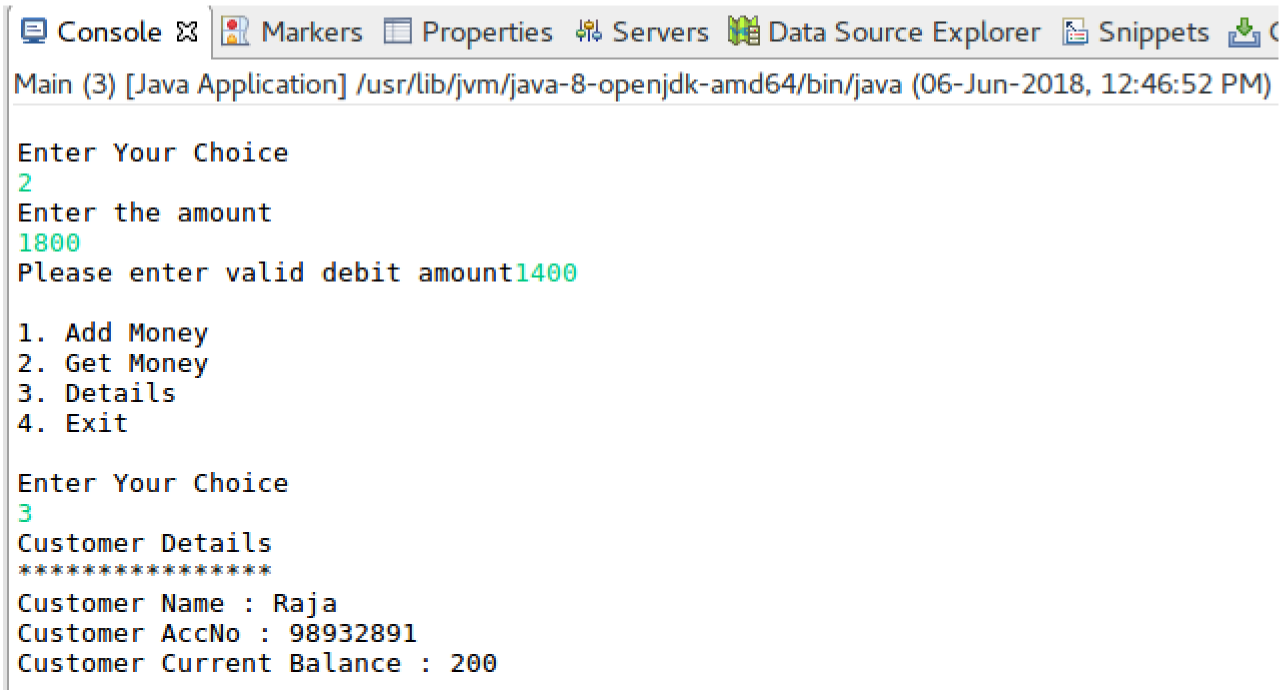
```
Main (3) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06-Jun-2018, 12:46:52 PM)
Enter Your Choice
2
Enter the amount
500

1. Add Money
2. Get Money
3. Details
4. Exit

Enter Your Choice
3
Customer Details
****************
Customer Name : Raja
Customer AccNo : 98932891
Customer Current Balance : 1500
```

Exception in Add Money:



```
Main (3) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06-Jun-2018, 12:46:52 PM)

Enter Your Choice
1
Enter the amount
-500
Please enter valid credit amount100

1. Add Money
2. Get Money
3. Details
4. Exit

Enter Your Choice
3
Customer Details
****************
Customer Name : Raja
Customer AccNo : 98932891
Customer Current Balance : 1600
```

Exception in Get Money:



```
Main (3) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06-Jun-2018, 12:46:52 PM)

Enter Your Choice
2
Enter the amount
1800
Please enter valid debit amount1400

1. Add Money
2. Get Money
3. Details
4. Exit

Enter Your Choice
3
Customer Details
****************
Customer Name : Raja
Customer AccNo : 98932891
Customer Current Balance : 200
```

Result:

The java console application that uses user defined exception handling techniques was developed and tested successfully.

Ex.No: 8	**Java Application for File Handling**
Date:	

Aim:

To create a Java console application to handle the files and find the file properties [Availability, Readable or Writeable or Both, Length of the File].

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
```

Coding:

UserFileHandler.java

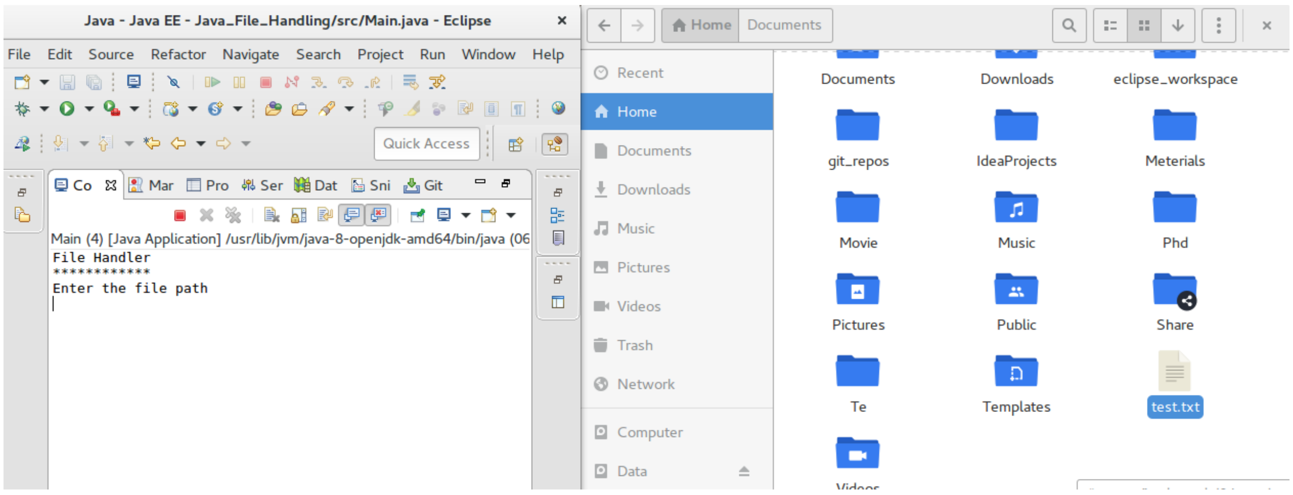
```
package com.raja.oopslab.files;

import java.io.File;
public class UserFileHandler{
        File aFile;
        boolean isReadable = false;
        boolean isWriteable = false;
        boolean isExists = false;
        int length = 0;
public UserFileHandler(String path) {
        aFile = new File(path);
        this.isExists = aFile.exists();
        this.isReadable = aFile.canRead();
        this.isWriteable = aFile.canWrite();
        this.length = (int) aFile.length();
}
public void fileDetails(){
        if(isExists){
                System.out.println("The File "+aFile.getName()+" is Available
at:"+aFile.getParent());
                if(isReadable && isWriteable)
                        System.out.println("File is Readable and Writeable");
                else if(isReadable)
                        System.out.println("File is Only Readable");
                else if(isWriteable)
                        System.out.println("File is Only Writeable");
                System.out.println("Total length of the file is :"+this.length+" bytes");
        }
        else
                System.out.println("File does not exists ");
}
}
```

Main.java

```
import java.io.File;
import java.util.Scanner;
import com.raja.oopslab.files.*;
public class Main {

        public static void main(String[] args) {
                String file_path = null;
                Scanner input = new Scanner(System.in);
                System.out.println("File Handler");
                System.out.println("************");
                System.out.println("Enter the file path");
                file_path = input.next();
                new UserFileHandler(file_path).fileDetails();
        }
}
```

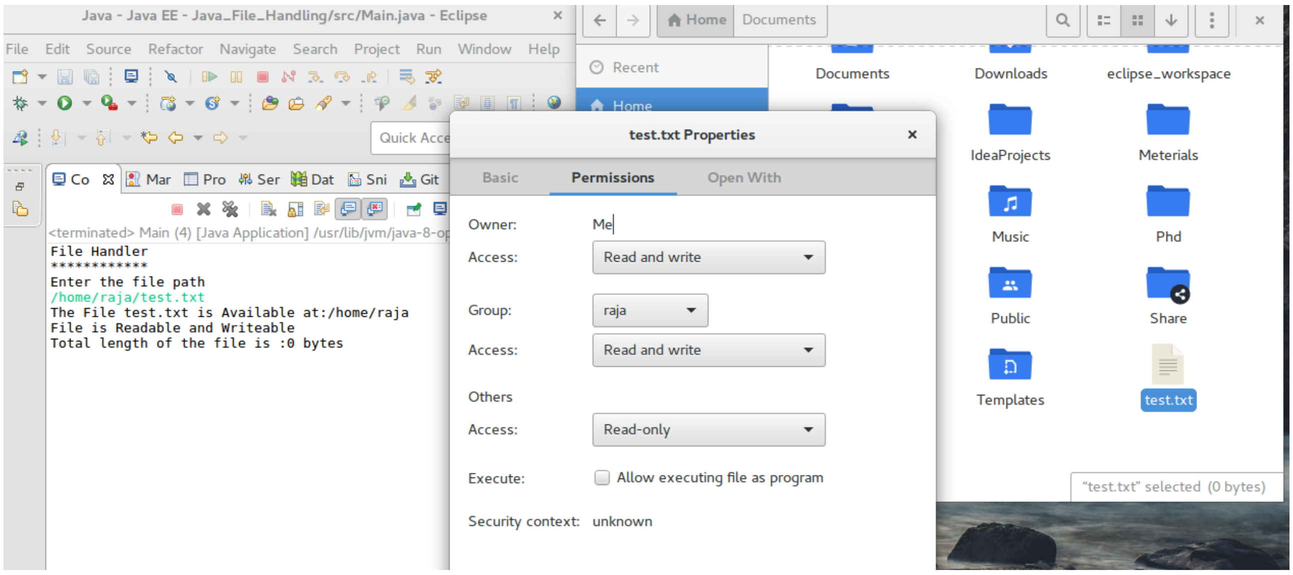
Output:

Availability of File:



```
Main (4) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06
File Handler
************
Enter the file path
```

File Read and Writeable:



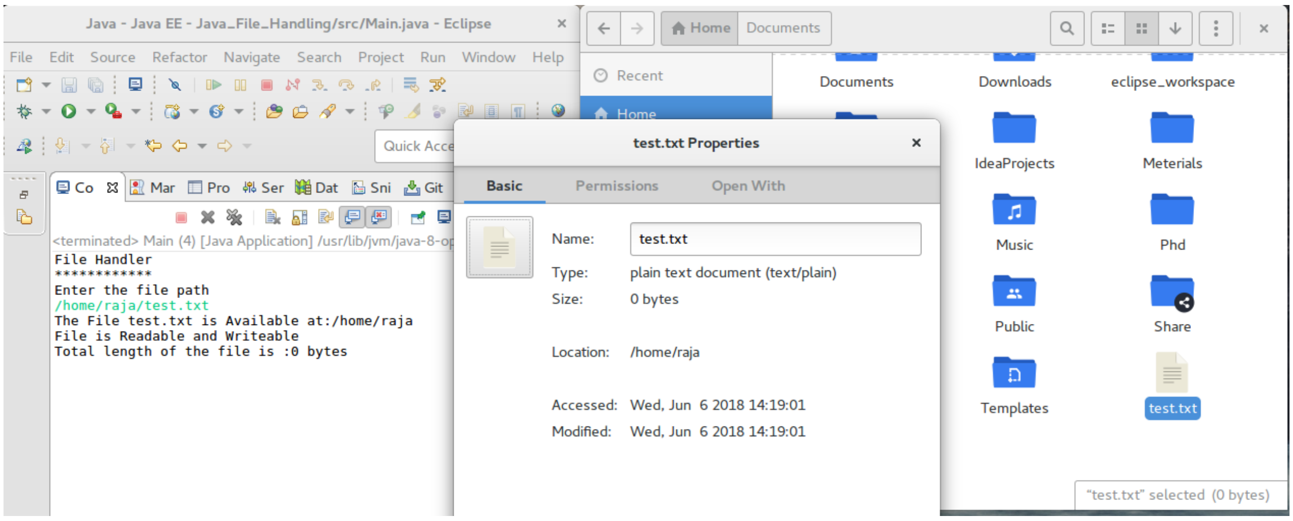
```
<terminated> Main (4) [Java Application] /usr/lib/jvm/java-8-op
File Handler
************
Enter the file path
/home/raja/test.txt
The File test.txt is Available at:/home/raja
File is Readable and Writeable
Total length of the file is :0 bytes
```

test.txt Properties

Owner: Me
Access: Read and write
Group: raja
Access: Read and write
Others
Access: Read-only
Execute: Allow executing file as program
Security context: unknown

"test.txt" selected (0 bytes)

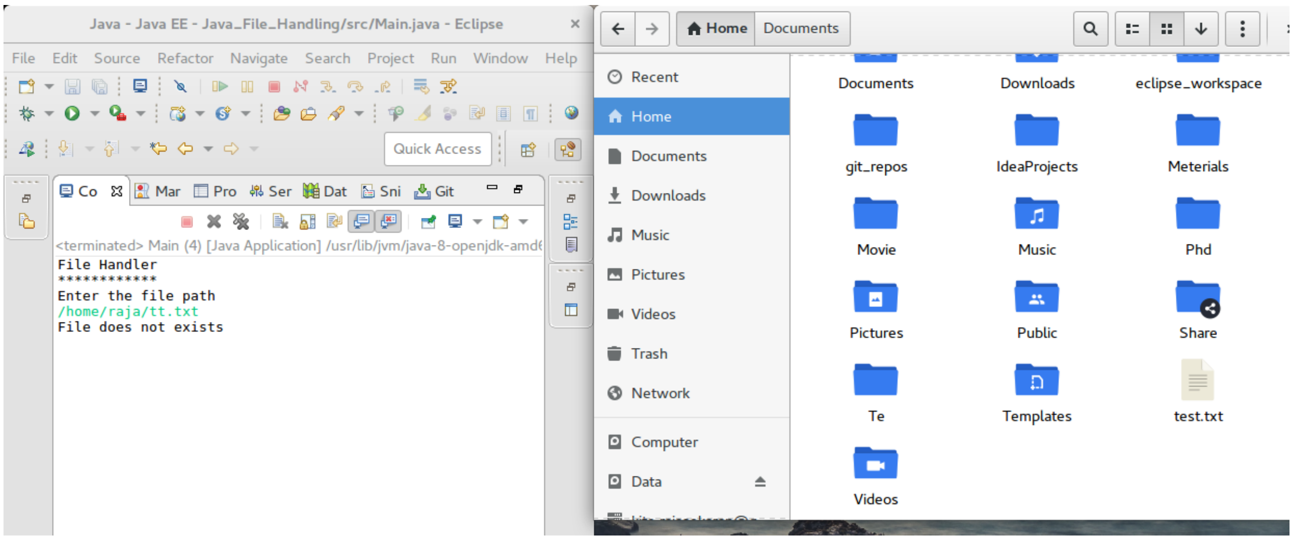
File Size:


Java - Java EE - Java_File_Handling/src/Main.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
test.txt Properties
Basic Permissions Open With
Home Documents
Recent
Documents Downloads eclipse_workspace
IdeaProjects Meterials
Music Phd
Public Share
Templates test.txt
"test.txt" selected (0 bytes)

```
<terminated> Main (4) [Java Application] /usr/lib/jvm/java-8-o
File Handler
************
Enter the file path
/home/raja/test.txt
The File test.txt is Available at:/home/raja
File is Readable and Writeable
Total length of the file is :0 bytes
```

Name: test.txt
Type: plain text document (text/plain)
Size: 0 bytes
Location: /home/raja
Accessed: Wed, Jun 6 2018 14:19:01
Modified: Wed, Jun 6 2018 14:19:01

File Not Exists:


Java - Java EE - Java_File_Handling/src/Main.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Home Documents
Recent
Home
Documents
Downloads
Music
Pictures
Videos
Trash
Network
Computer
Data
Documents Downloads eclipse_workspace
git_repos IdeaProjects Meterials
Movie Music Phd
Pictures Public Share
Te Templates test.txt
Videos

```
<terminated> Main (4) [Java Application] /usr/lib/jvm/java-8-openjdk-amd6
File Handler
************
Enter the file path
/home/raja/tt.txt
File does not exists
```

Result:

The java console application for handling files was developed and tested successfully.

Ex.No: 9	**Java Application for Multi threading**
Date:	

Aim:

To create a Java console application the uses the multi threading concepts in java. The Application has 3 threads one creates random number, one thread computes square of that number and another one computes the cube of that 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
```

Coding:

RandomNumberThread.java

```
package com.raja.oopslab.threading;

import java.util.Random;

public class RandomNumberThread extends Thread{
	Random num = new Random();
	int value;
	@Override
	public void run(){
		while(true){
			try {
				this.sleep(1000);
			} catch (InterruptedException e) {

			}
			value = num.nextInt(1000);
			System.out.println("RandomNumberThread generated a number "+value);
			if(value % 2 == 0)
				new SquareGenThread(value).start();
			else
				new CubeGenThread(value).start();
		}

	}
}
```

SquareGenThread.java

```
package com.raja.oopslab.threading;

public class SquareGenThread extends Thread{
	int number;
	int squre;
	public SquareGenThread(int number) {
		this.number = number;
	}
	@Override
	public void run(){
		try {
			this.sleep(3000);
		} catch (InterruptedException e) {

		}
		this.squre = this.number * this.number;
		System.out.println("SquareGenThread--> Square of "+number+" is "+squre);
	 }
}
```

CubeGenThread.java

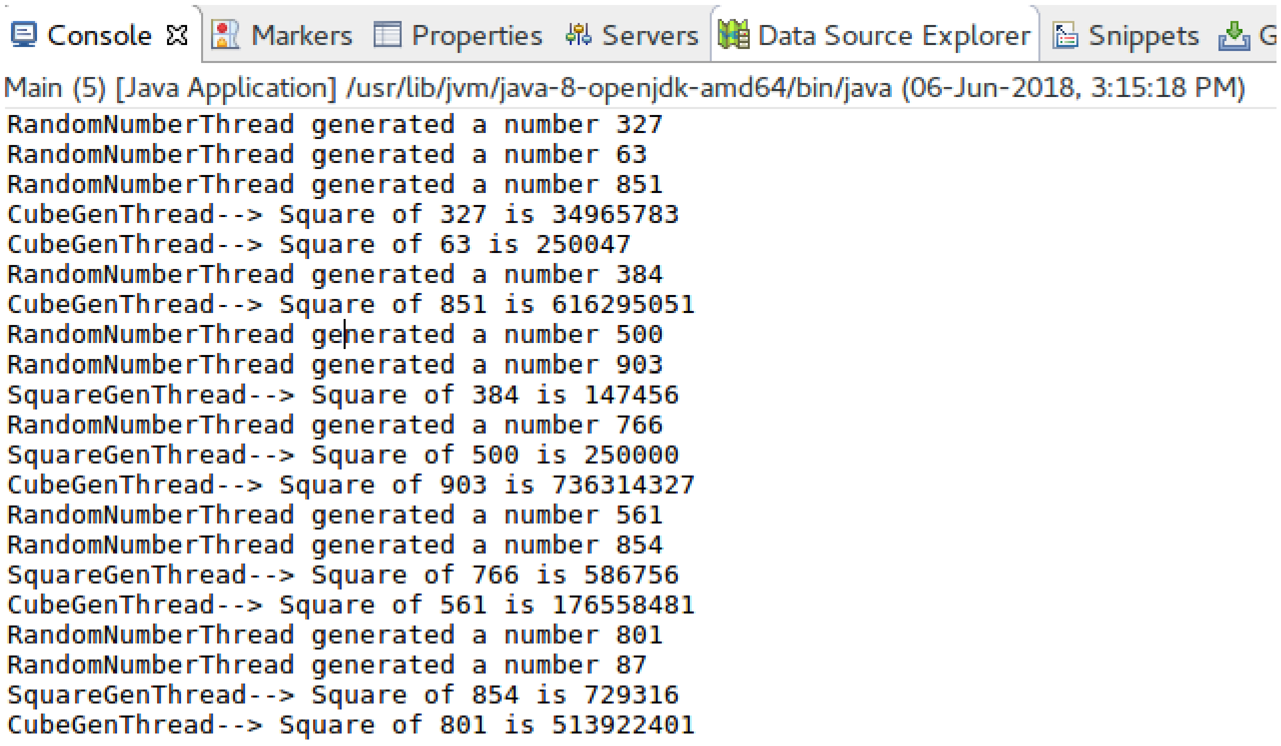
```
package com.raja.oopslab.threading;

public class CubeGenThread extends Thread{
	int number;
```

```
        int squre;
        public CubeGenThread(int number) {
                this.number = number;
        }
        @Override
        public void run(){
                try {
                        this.sleep(2000);
                } catch (InterruptedException e) {

                }
                this.squre = this.number * this.number * this.number;
                System.out.println("CubeGenThread--> Square of "+number+" is "+squre);
         }
}
```

Main.java

```
import java.util.Random;
import com.raja.oopslab.threading.RandomNumberThread;
public class Main {
        public static void main(String[] args) {
                new RandomNumberThread().start();
        }
}
```

Output:



```
Main (5) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06-Jun-2018, 3:15:18 PM)
RandomNumberThread generated a number 327
RandomNumberThread generated a number 63
RandomNumberThread generated a number 851
CubeGenThread--> Square of 327 is 34965783
CubeGenThread--> Square of 63 is 250047
RandomNumberThread generated a number 384
CubeGenThread--> Square of 851 is 616295051
RandomNumberThread generated a number 500
RandomNumberThread generated a number 903
SquareGenThread--> Square of 384 is 147456
RandomNumberThread generated a number 766
SquareGenThread--> Square of 500 is 250000
CubeGenThread--> Square of 903 is 736314327
RandomNumberThread generated a number 561
RandomNumberThread generated a number 854
SquareGenThread--> Square of 766 is 586756
CubeGenThread--> Square of 561 is 176558481
RandomNumberThread generated a number 801
RandomNumberThread generated a number 87
SquareGenThread--> Square of 854 is 729316
CubeGenThread--> Square of 801 is 513922401
```

Result:

The java console application for multithreading was developed and tested successfully.

Ex.No: 10	**Java Application for Generic Max Finder**
Date:	

Aim:

To create a Java console application that finds the maximum in a array based on the type of the elements using generic functions in java.

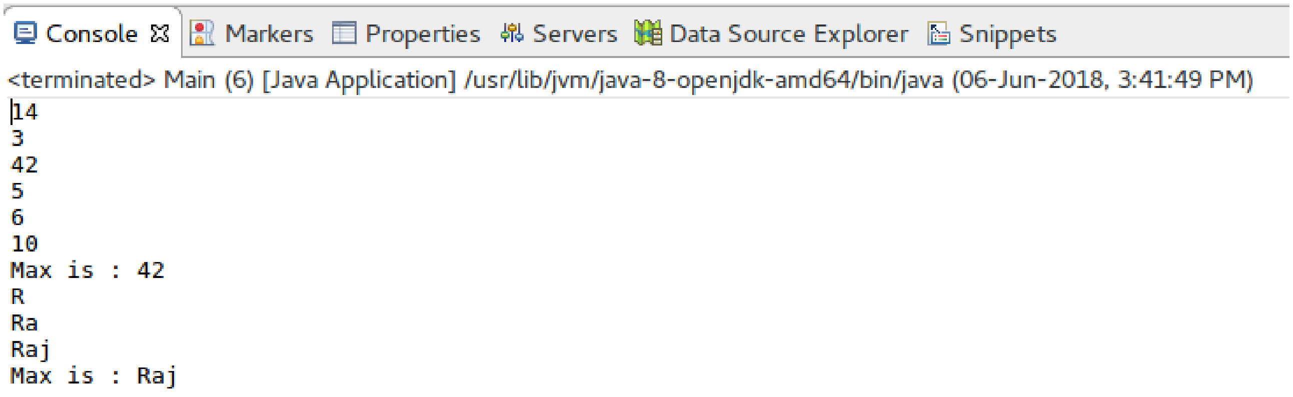
Algorithm:

```
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    Goto step 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    Goto step 3.1 until all the elements has been processed.
Step 5    Stop the Process
```

Coding:

```
class GenericMax {
        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:



```
<terminated> Main (6) [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (06-Jun-2018, 3:41:49 PM)
14
3
42
5
6
10
Max is : 42
R
Ra
Raj
Max is : Raj
```

Result:

The java console application for finding generic max of given elements was developed and tested successfully.

Ex.No: 11	**Calculator Application using java AWT packages**
Date:	

Aim:

To create a Java GUI application that mimics the basic and advanced functionalities of a scientific calculator.

Algorithm:

```
Step 1    Start the Process
Step 2    Display Text View and Number Pad and Option Pads
Step 3    If user presses any number get the existing numbers in Text View add them up and
          display
Step 4    If user press Operators
          Step 4.1    Get the Text View content as Operant 1 and Set the display to null.
          Step 4.2    If user pressed "+" button set operator as plus
          Step 4.3    If user pressed "-" button set operator as minus
          Step 4.4    If user pressed "x" button set operator as multiply
          Step 4.5    If user pressed "/" button set operator as divide
          Step 4.6    Goto step 3
Step 5    If user pressed "=" button then proceed following steps.
          Step 5.1    Get the Text View content as Operant 2 and Set the display to null.
          Step 5.2    If operator is "plus" then display Text View as Operant 1 + Operant 2
          Step 5.3    If operator is "minus" then display Text View as Operant 1 - Operant 2
          Step 5.4    If operator is "multiply" then display Text View as Operant 1 * Operant 2
          Step 5.5    If operator is "divide" then display Text View as Operant 1 / Operant 2
          Step 5.6    Goto step 4
Step 6    If advanced button pressed
          Step 6.1    Change Operant Types [+,-,x,/ into sin, cos, tan, log] and goto step 2
Step 7    If user pressed any of the operator
          Step 7.1    Get the Text View content as Operant 1 and Set the display to null.
          Step 7.2    If user pressed "sin" button set display the sin value of Operant 1
          Step 7.3    If user pressed "cos" button set display the cos value of Operant 1
          Step 7.4    If user pressed "tan" button set display the tan value of Operant 1
          Step 7.5    If user pressed "log" button set display the log value of Operant 1
          Step 7.6    Goto step 7
Step 8    If advanced pressed again then revert the button changes and return back to normal
Step 9    Repeat the process until user presses the close button then Stop the Process.
```

Coding:

```
import java.awt.BorderLayout;
import java.awt.Button;
import java.awt.Font;
import java.awt.Frame;
import java.awt.GridLayout;
import java.awt.Panel;
import java.awt.TextField;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;

class Numpan extends Panel implements ActionListener{

        Button n0,n1,n2,n3,n4,n5,n6,n7,n8,n9,point,equal;
        Button plus,minus,multiply, divide;
        Button m_plus,m_minus,clear,advanced;
        TextField display;
        String op1,op2,result;
        String op_flag;
        String data;
        double dop1,dop2,dresult;
        boolean flag_advanced=true;
        public Numpan(TextField display) {
                this.display = display;
                setLayout(new GridLayout(0,4));

                n0 = new Button("0");
                n0.setActionCommand("zero");
                n0.addActionListener(this);

                n1 = new Button("1");
                n1.setActionCommand("one");
                n1.addActionListener(this);

                n2 = new Button("2");
                n2.setActionCommand("two");
                n2.addActionListener(this);

                n3 = new Button("3");
                n3.setActionCommand("three");
                n3.addActionListener(this);

                n4 = new Button("4");
                n4.setActionCommand("four");
                n4.addActionListener(this);

                n5 = new Button("5");
                n5.setActionCommand("five");
                n5.addActionListener(this);

                n6 = new Button("6");
                n6.setActionCommand("six");
                n6.addActionListener(this);
```

```
n7 = new Button("7");
n7.setActionCommand("seven");
n7.addActionListener(this);

n8 = new Button("8");
n8.setActionCommand("eight");
n8.addActionListener(this);

n9 = new Button("9");
n9.setActionCommand("nine");
n9.addActionListener(this);

point = new Button(".");
point.setActionCommand("point");
point.addActionListener(this);

equal = new Button("=");
equal.setActionCommand("equal");
equal.addActionListener(this);

plus = new Button("+");
plus.setActionCommand("plus");
plus.addActionListener(this);

minus = new Button("-");
minus.setActionCommand("minus");
minus.addActionListener(this);

multiply = new Button("x");
multiply.setActionCommand("multiply");
multiply.addActionListener(this);

divide = new Button("/");
divide.setActionCommand("divide");
divide.addActionListener(this);

m_plus = new Button("M+");
m_plus.setActionCommand("m_plus");
m_plus.addActionListener(this);

m_minus = new Button("M-");
m_minus.setActionCommand("m_minus");
m_minus.addActionListener(this);

clear = new Button("C");
clear.setActionCommand("clear");
clear.addActionListener(this);

advanced = new Button("ADV");
advanced.setActionCommand("advanced");
advanced.addActionListener(this);

add(m_plus);
add(m_minus);
add(clear);
add(advanced);
add(n1);
add(n2);
```

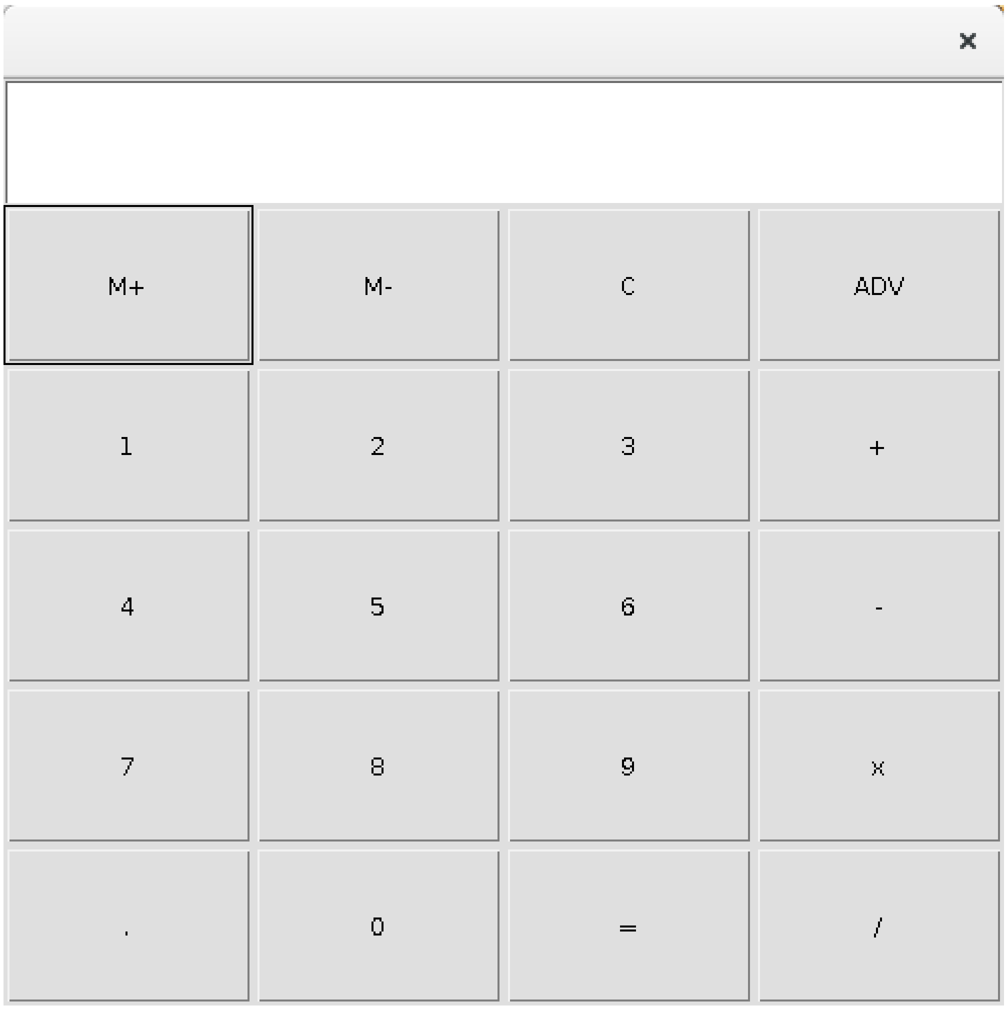
```
        add(n3);
        add(plus);
        add(n4);
        add(n5);
        add(n6);
        add(minus);
        add(n7);
        add(n8);
        add(n9);
        add(multiply);
        add(point);
        add(n0);
        add(equal);
        add(divide);
}
public String getDisplayText(){
        return display.getText().toString();
}
public void setDisplay(String text){
        display.setText(text);
}
public void clearDisplay(){
        System.out.println("Clear Called");
        setDisplay("");
        data = "";
}

public void changeAdvanced(boolean toAdvanced){
        if(toAdvanced){
                plus.setLabel("sin");
                plus.setActionCommand("sin");
                //System.out.println("cos in");
                minus.setLabel("cos");
                minus.setActionCommand("cos");
                //System.out.println("cos out");
                multiply.setLabel("tan");
                multiply.setActionCommand("tan");
                divide.setLabel("log");
                divide.setActionCommand("log");

        }
        else{
                plus.setLabel("+");
                plus.setActionCommand("plus");
                minus.setLabel("-");
                minus.setActionCommand("minus");
                multiply.setLabel("x");
                multiply.setActionCommand("multiply");
                divide.setLabel("/");
                divide.setActionCommand("divide");
        }
}
@Override
public void actionPerformed(ActionEvent e) {

        data = getDisplayText();

        switch(e.getActionCommand()){
```

```
case "zero":
        setDisplay(data+"0");
        break;
case "one":
        setDisplay(data+"1");
        break;
case "two":
        setDisplay(data+"2");
        break;
case "three":
        setDisplay(data+"3");
        break;
case "four":
        setDisplay(data+"4");
        break;
case "five":
        setDisplay(data+"5");
        break;
case "six":
        setDisplay(data+"6");
        break;
case "seven":
        setDisplay(data+"7");
        break;
case "eight":
        setDisplay(data+"8");
        break;
case "nine":
        setDisplay(data+"9");
        break;

case "plus":
        op1 = data;
        op_flag = "plus";
        clearDisplay();
        break;
case "minus":
        op1 = data;
        op_flag = "minus";
        clearDisplay();
        break;
case "multiply":
        op1 = data;
        op_flag = "multiply";
        clearDisplay();
        break;
case "divide":
        op1 = data;
        op_flag = "divide";
        clearDisplay();
        break;
case "clear":
        clearDisplay();
        break;
case "advanced":
        if(flag_advanced){
                changeAdvanced(true);
                flag_advanced = false;
                }
                else{
                        changeAdvanced(false);
                        flag_advanced = true;
                }
                break;

        case "sin":
                op1 = data;
                setDisplay(String.valueOf(Math.sin(Double.valueOf(op1))));
                break;
        case "cos":
                op1 = data;
                setDisplay(String.valueOf(Math.cos(Double.valueOf(op1))));
                break;
        case "tan":
                op1 = data;
                setDisplay(String.valueOf(Math.tan(Double.valueOf(op1))));
                break;
        case "log":
                op1 = data;
                setDisplay(String.valueOf(Math.log(Double.valueOf(op1))));
                break;
        case "equal":
                switch(op_flag){
                case "plus":
                        op2 = data;
                        clearDisplay();
                        dop1 = Double.parseDouble(op1);
                        dop2 = Double.parseDouble(op2);
                        dresult = dop1 + dop2;
                        result = String.valueOf(dresult);
                        setDisplay(result);
                        op_flag = "";
                        break;
                case "minus":
                        op2 = data;
                        clearDisplay();
                        dop1 = Double.parseDouble(op1);
                        dop2 = Double.parseDouble(op2);
                        dresult = dop1 - dop2;
                        result = String.valueOf(dresult);
                        setDisplay(result);
                        op_flag = "";
                        break;

                case "multiply":
                        op2 = data;
                        clearDisplay();
                        dop1 = Double.parseDouble(op1);
                        dop2 = Double.parseDouble(op2);
                        dresult = dop1 * dop2;
                        result = String.valueOf(dresult);
                        setDisplay(result);
                        op_flag = "";
                        break;

                case "divide":
```

```
                                op2 = data;
                                clearDisplay();
                                dop1 = Double.parseDouble(op1);
                                dop2 = Double.parseDouble(op2);
                                dresult = dop1 / dop2;
                                result = String.valueOf(dresult);
                                setDisplay(result);
                                op_flag = "";
                                break;


                        }
                }
        }
}

class Calculator extends Frame {
        TextField display;
        public Calculator() {
                display = new TextField();
                display.setFont(new Font("Times New Roman", Font.BOLD, 50));
                setLayout(new BorderLayout());
                add(new Numpan(display),BorderLayout.CENTER);
                add(display,BorderLayout.NORTH);
                setVisible(true);
                setSize(500,500);
                addWindowListener(new WindowAdapter() {
                        @Override
                        public void windowClosing(WindowEvent e) {
                                dispose();
                        }
                });
        }

}
public class Main {

        public static void main(String[] args) {
                new Calculator();
        }

}
```

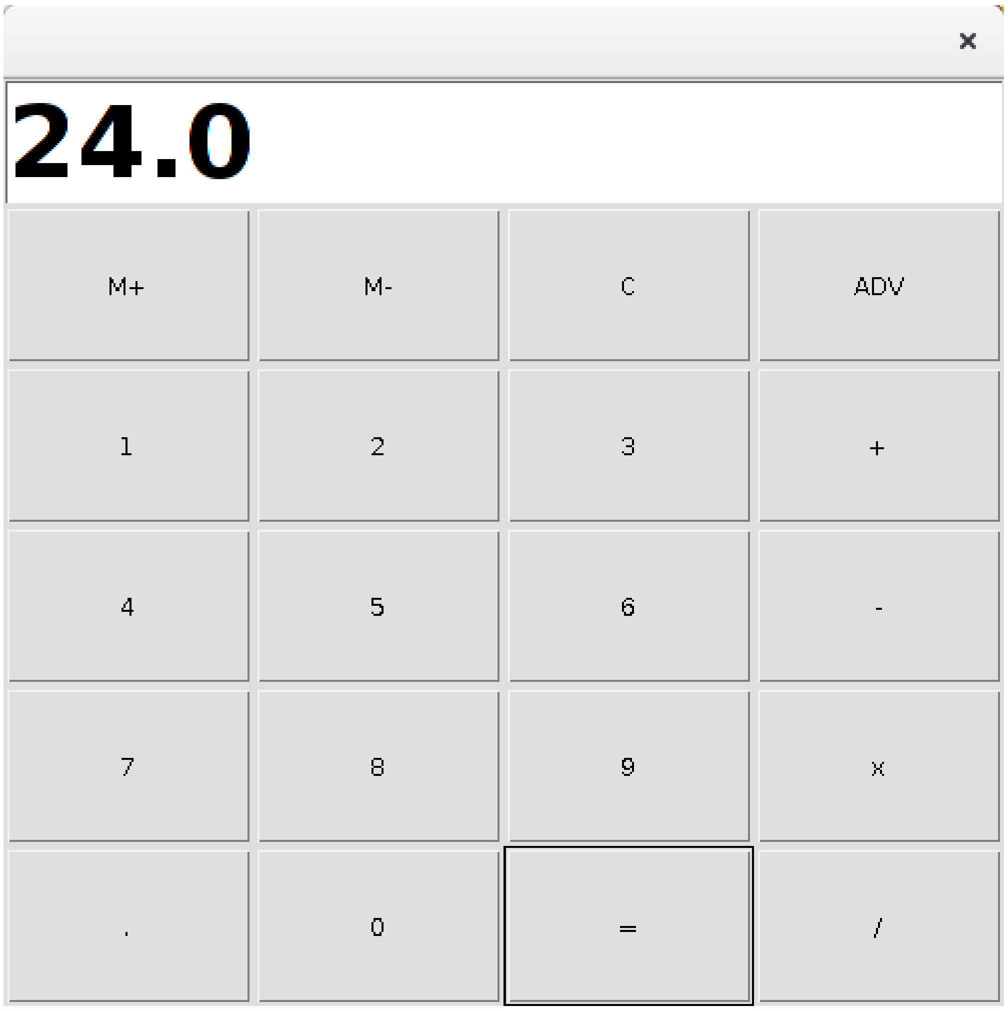
Output:

Calculator:

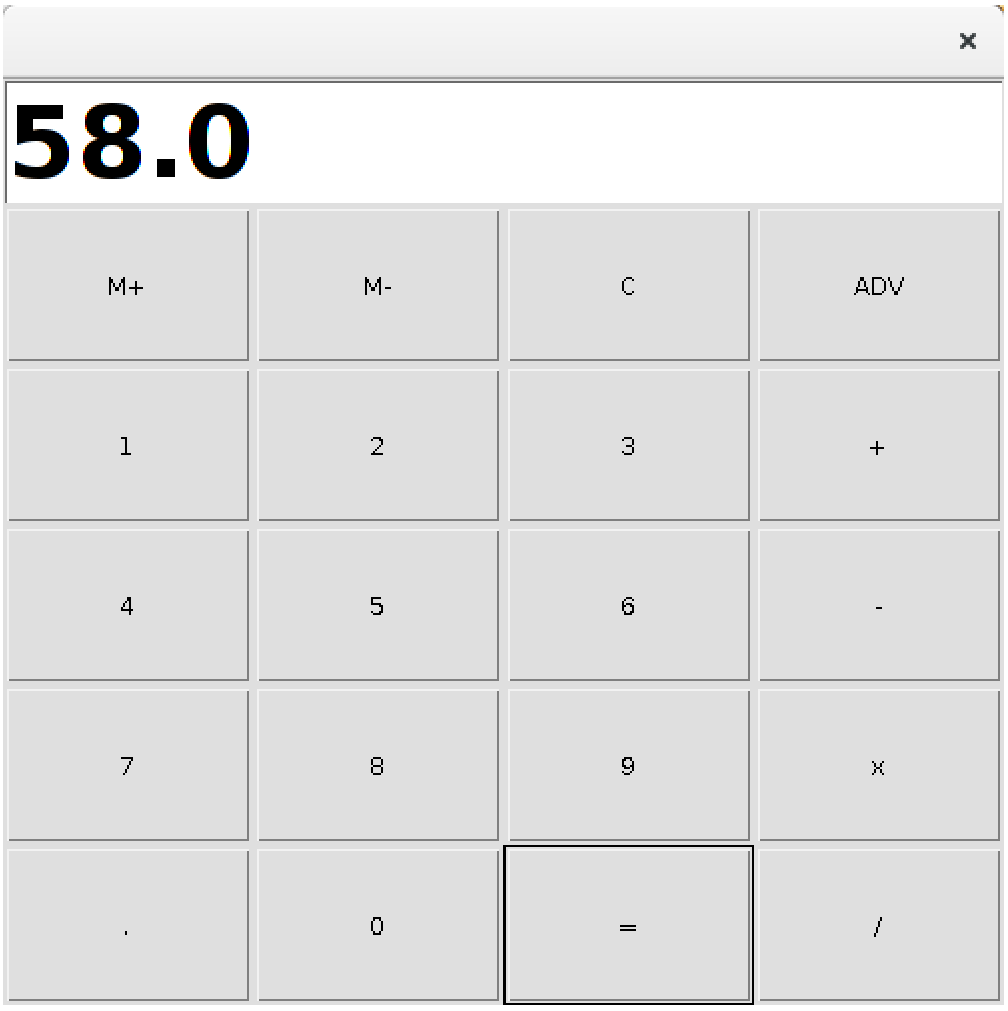


Basic Operations

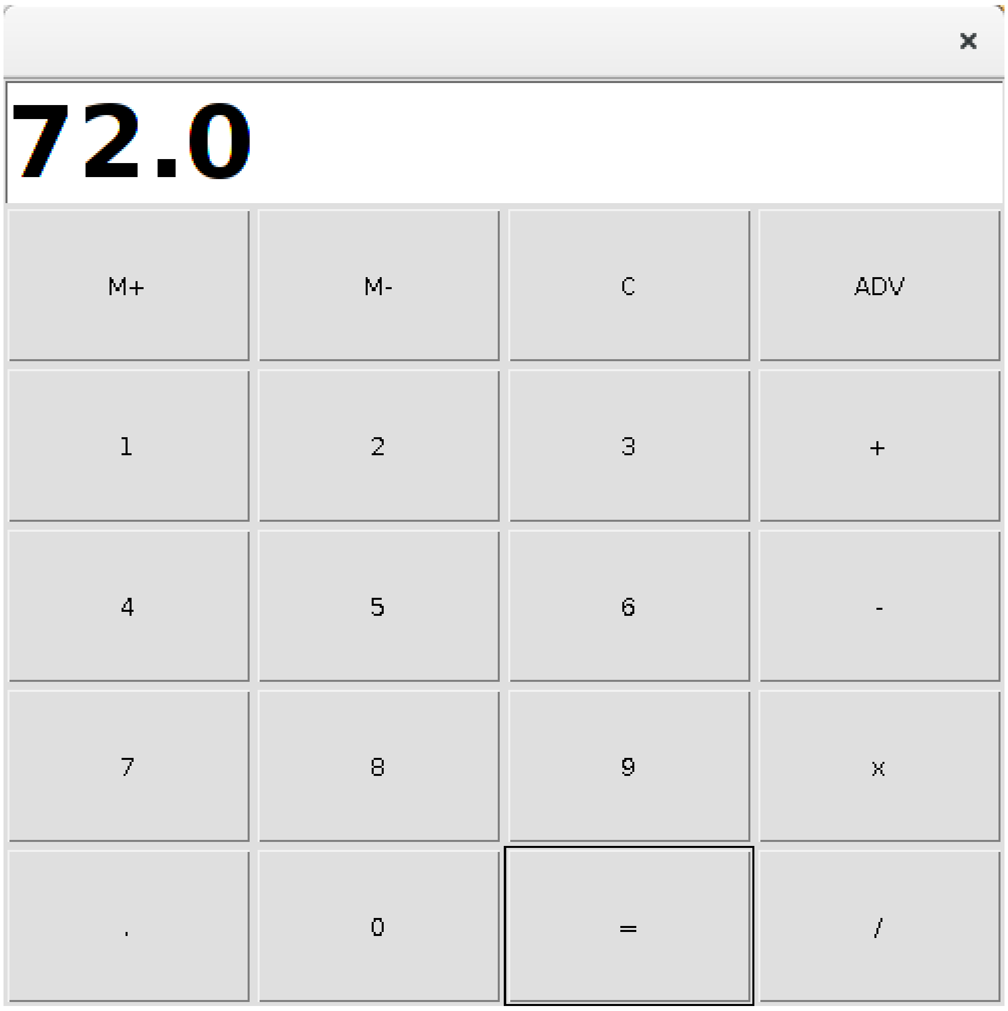
Addition [12 + 12]


24.0

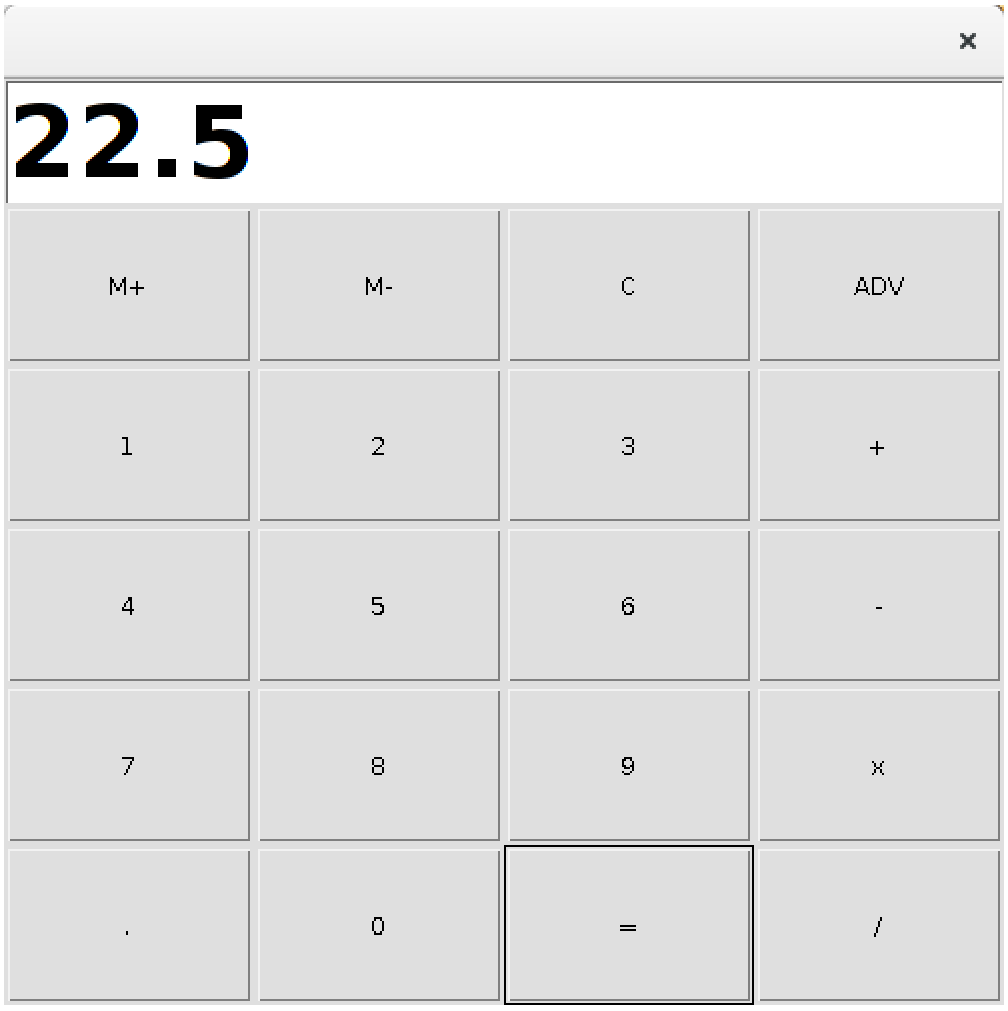
Subtraction [90 – 32]:



Multiplication [36 X 2]

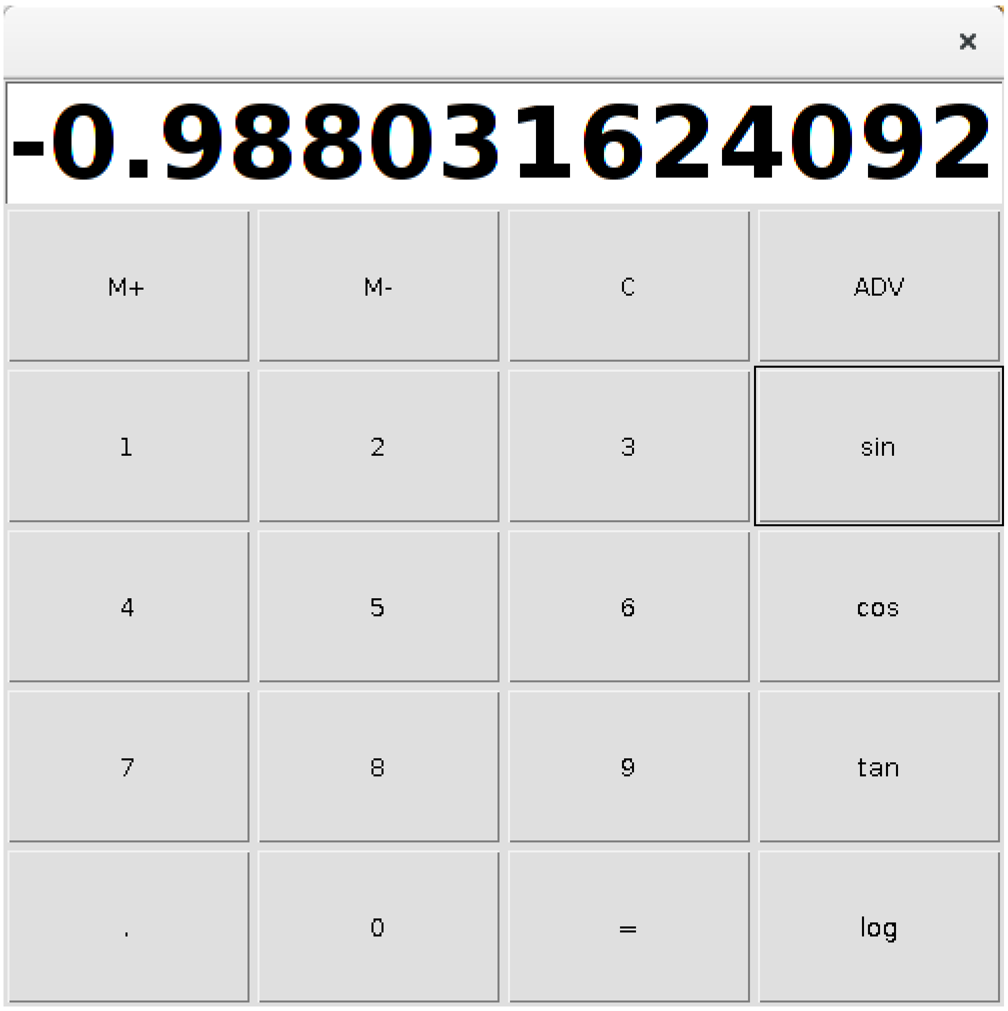

72.0

Division [45 / 2]

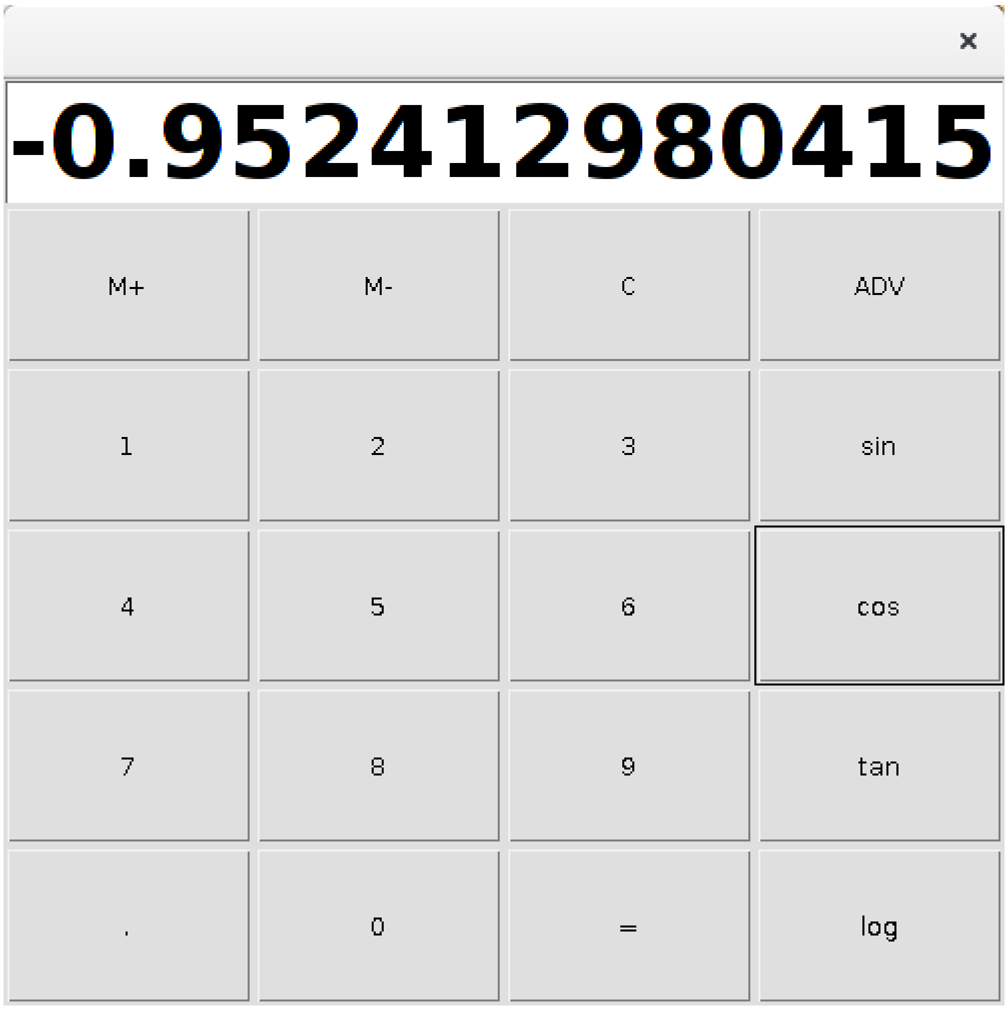


Scientific Operations:

Sin 30

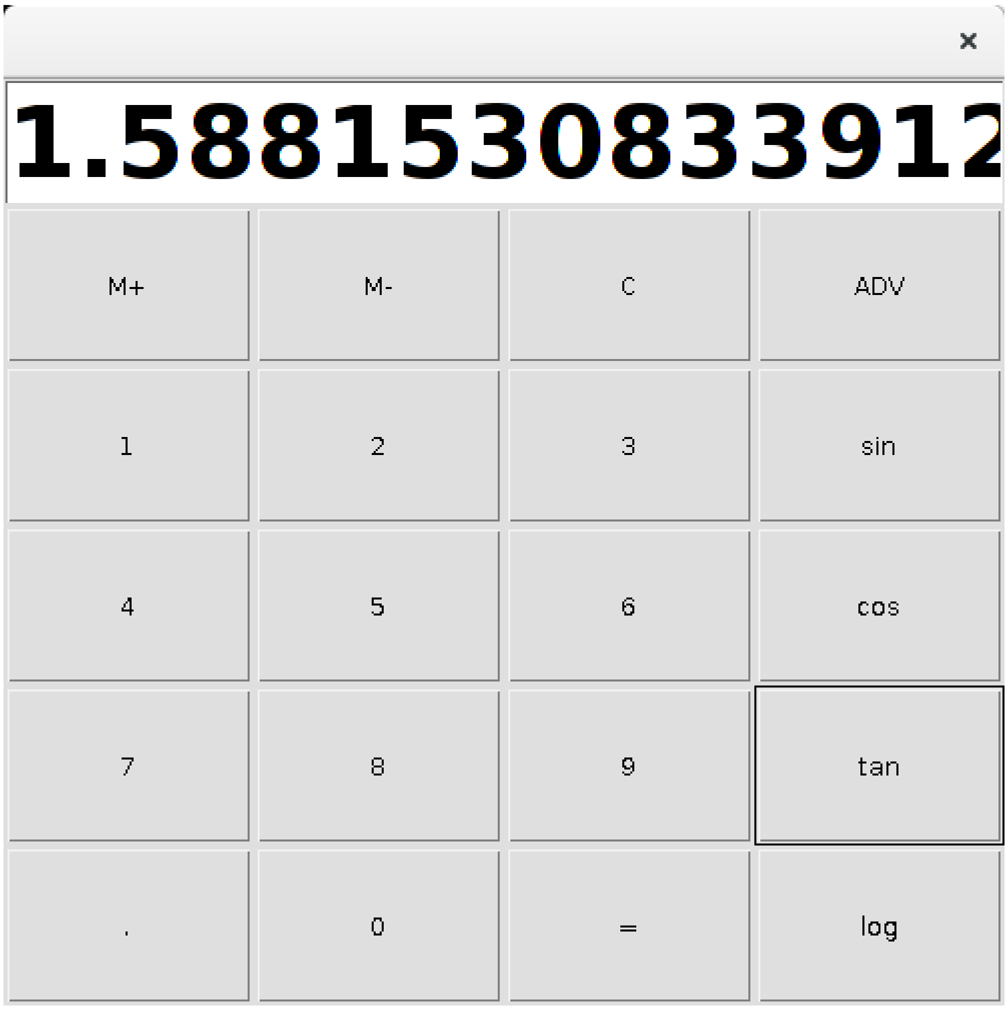

-0.988031624092

Cos 40:



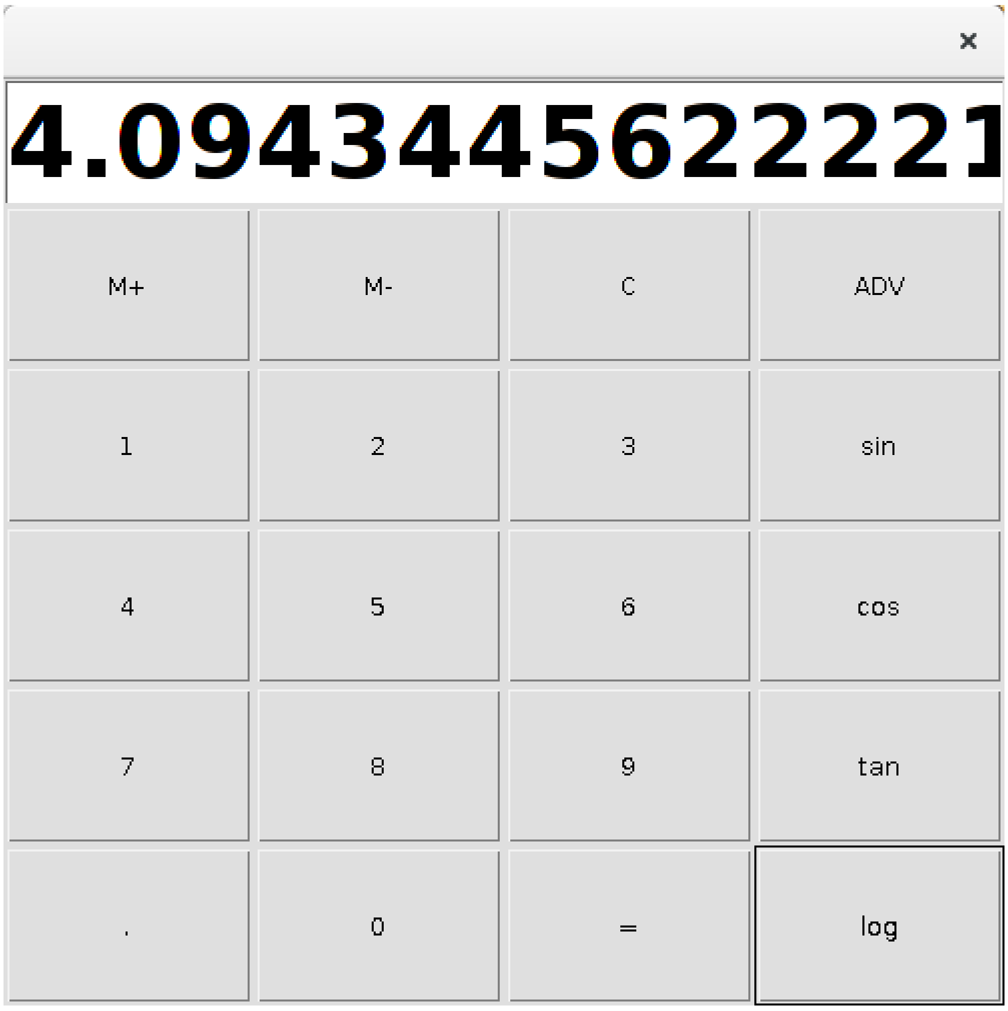
```
-0.952412980415
```

Tan 23



```
1.58815308833912
```

Log 60



Result:

The java GUI application for calculator with basic and advanced functionalities was developed and tested successfully.

Important Links			
S No	**Experiment**	**Source**	**Demo**
1	Electricity Bill Calculator	https://github.com/rajasekaranap/CS8383_Java_EB_Bill_Calculator	https://youtu.be/1uSOIZDz-AA
2	Unit Converters Application using packages	https://github.com/rajasekaranap/CS8383_Java_Converters	https://youtu.be/8HBADevR56M
3	Employee Payroll System	https://github.com/rajasekaranap/CS8383_Java_Employee_Payroll	https://youtu.be/8KuDnx8hxxQ
4	Java Application for ADT Stack using Interfaces	https://github.com/rajasekaranap/CS8383_Java_StackADT	https://youtu.be/KsWfHmRvDaU
5	Java Application for String Operations using ArrayList	https://github.com/rajasekaranap/CS8383_Java_String_List	https://youtu.be/-3Ge0TpVUvI
6	Java Application to Find the Area of different Shapes	https://github.com/rajasekaranap/CS8383_Java_Abstract_Area	https://youtu.be/mVI01UCZ1cw
7	Bank Transaction System with User Exceptions	https://github.com/rajasekaranap/CS8383_Java_User_Exception_Bank	https://youtu.be/jknMYRQw-ac
8	Java Application for File Handling	https://github.com/rajasekaranap/CS8383_Java_User_FileHandler	https://youtu.be/9VhvcH3G4ws
9	Java Application for Multi threading	https://github.com/rajasekaranap/CS8383_Java_Multi_Threading	https://youtu.be/1W8-ce1trPU
10	Java Application for Generic Max Finder	https://github.com/rajasekaranap/CS8383_Java_Generic_Max	https://youtu.be/L0KjoXi6FwM
11	Calculator Application using java AWT packages	https://github.com/rajasekaranap/CS8383_Java_AWT_Calculator_Advanced	https://youtu.be/QQMWrG_NhFY

THANK YOU