B.Tech(5th Sem)

Java Lab File

Bhagwan Parshuram Institute of Technology

Submitted by :	Khushi
Class	CSE A
Roll No	02120802719
Submitted to :	Dr. Bhawna

INDEX

EXPERIMENT 1

Aim- Write a program for hello world that also displays the current date and time .

Code

```
import java.time.format.DateTimeFormatter;

import java.time.LocalDateTime;



public class Hello{

 public static void main(String[] args) {

     System.out.println("Hey , I'm Khushi");

     DateTimeFormatter dtf= DateTimeFormatter.ofPattern("dd/mm/yyyy hh:mm:ss");

     LocalDateTime now=LocalDateTime.now();

     System.out.println(dtf.format(now));




   }

}
```

```
TERMINAL    DEBUG CONSOLE    PROBLEMS 2    OUTPUT

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Drs-MacBook-Air:java drvntiwari$ javac Hello.java
Drs-MacBook-Air:java drvntiwari$ java Hello
Hey , I'm Khushi
05/44/2021 06:44:21
Drs-MacBook-Air:java drvntiwari$
```

EXPERIMENT 2

Aim- Write a program to implement the stack and queue concept .

Code

- Queue

```
public class Q {
	protected int[] data;
	protected int front;
	protected int size;

	public Q() {
	    data = new int[5];
	    front = 0;
	    size = 0;
	}

	public Q(int cap) {
	    data = new int[cap];
	    front = 0;
	    size = 0;
	}

	public void enqueue(int item) throws Exception {

	    if (isFull()) {
	        throw new Exception("Queue is Full.");
	    }
```

```
        int idx = (front + size) % data.length;
        data[idx] = item;

        size++;
    }

    public int dequeue() throws Exception {

        if (isEmpty()) {
            throw new Exception("Queue is Empty.");
        }

        int temp = data[front];
        data[front] = 0;

        front = (front + 1) % data.length;
        size--;

        return temp;

    }

    public int getFront() throws Exception {

        if (isEmpty()) {
            throw new Exception("Queue is Empty.");
        }

        int temp = data[front];
        return temp;
```

```
    }

    public int size() {
        return size;
    }

    public boolean isEmpty() {
        return size == 0;
    }

    public boolean isFull() {
        return size == data.length;
    }

    public void display() {

        System.out.println("---------------");

        for (int i = 0; i < size; i++) {
            int idx = (i + front) % data.length;
            System.out.print(data[idx] + " ");
        }

        System.out.println(".");
        System.out.println("---------------");

    }

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

```
            Q queue = new Q();
            queue.enqueue(10);
            queue.enqueue(20);
            queue.enqueue(30);
            queue.enqueue(40);
            queue.enqueue(50);

            queue.display();

            System.out.println(queue.dequeue());
            System.out.println(queue.dequeue());
            queue.display();
            queue.enqueue(60);
            queue.enqueue(70);

            queue.display();
        }
    }
```

```
Drs-MacBook-Air:java drvntiwari$ javac Q.java
Drs-MacBook-Air:java drvntiwari$ java Q
--------------
10 20 30 40 50 .
--------------
10
20
--------------
30 40 50 .
--------------
--------------
30 40 50 60 70 .
--------------
Drs-MacBook-Air:java drvntiwari$
```

- Stack

```
import java.util.Scanner;

public class Stack {


   protected int[] data;

   protected int tos;


   public Stack() { //constructor

       data = new int[5];

       tos = -1;

   }

   public Stack(int cap) {

       data = new int[cap];

       tos = -1;

   }


   public void push(int item) throws Exception {


       if (isFull()) {

           throw new Exception("Stack is Full.");

       }


       tos++;

       data[tos] = item;

   }


   public int pop() throws Exception {


       if (isEmpty()) {

           throw new Exception("Stack is Empty.");
```

```
    }

    int temp = data[tos];

    data[tos] = 0;
    tos--;

    return temp;

}

public int peek() throws Exception {

    if (isEmpty()) {
        throw new Exception("Stack is Empty.");
    }

    int temp = data[tos];
    return temp;

}

public int size() {
    return tos + 1;
}

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

```
public boolean isFull() {
    return size() == data.length;
}

public void display() {

    System.out.println("-----------------");

    System.out.print("(top) ");

    for (int i = tos; i >= 0; i--) {
        System.out.print(data[i] + " ");
    }

    System.out.println(" ");
    System.out.println("-----------------");

}
public static void main(String[] args)
{
    Scanner scan = new Scanner(System.in);
    System.out.println("--------- Stack Operations ---------\n");
    System.out.print("Enter the size of stack: ");
    int n = scan.nextInt();

    Stack s = new Stack(n);

    System.out.println("1. push");
    System.out.println("2. pop");
    System.out.println("3. peek");
```

```
    char ch;
    do{
        System.out.print("\nEnter choice: ");
        int choice = scan.nextInt();
        switch (choice)
        {
        case 1 :
            System.out.print("Enter integer element to push: ");
            try {
                s.push( scan.nextInt() );
            }
            catch (Exception e) {
                System.out.println("Error : " + e.getMessage());
            }
            break;

        case 2 :
            try {
                System.out.println("Popped Element = " + s.pop());
            }
            catch (Exception e) {
                System.out.println("Error : " + e.getMessage());
            }
            break;

        case 3 :
            try {
                System.out.println("Peek Element = " + s.peek());
            }
```

```
                catch (Exception e) {
                    System.out.println("Error : " + e.getMessage());
                }
                break;

            default :
                System.out.println("Wrong Entry \n ");
                break;
            }
            /* display stack */
            s.display();
            System.out.print("\nDo you want to continue (Type y or n): \n");
            ch = scan.next().charAt(0);

        } while (ch == 'Y' || ch == 'y');
        scan.close();
    }
}
```

```
Drs-MacBook-Air:java drvntiwari$ javac Stack.java
Drs-MacBook-Air:java drvntiwari$ java Stack
--------- Stack Operations ---------

Enter the size of stack: 5
1. push
2. pop
3. peek

Enter choice: 1
Enter integer element to push: 11
----------------
(top) 11
----------------

Do you want to continue (Type y or n):
y

Enter choice: 1
Enter integer element to push: 22
----------------
(top) 22 11
----------------

Do you want to continue (Type y or n):
y

Enter choice: 13
Wrong Entry

----------------
(top) 22 11
----------------

Do you want to continue (Type y or n):
y

Enter choice: 2
Popped Element = 22
----------------
(top) 11
----------------

Do you want to continue (Type y or n):
y

Enter choice: 3
Peek Element = 11
----------------
(top) 11
----------------

Do you want to continue (Type y or n):
n
Drs-MacBook-Air:java drvntiwari$
```

EXPERIMENT 3

Aim-

1. **Write a program to implement dynamic initialization .**
2. **Write a program to implement java literals .**
3. **Write a program to implement scope of a variable.**

Code

1.

```
public class Dynamic {

 public static void main(String[] args) {

     double a =3.0 ,b = 4.0f;

     double c =Math.sqrt(a*a + b*b);

     double d = c*a;

     System.out.println("Hypotenuse is " + c);

     System.out.println("Value of d is " + d);

   }

}
```

```
Drs-MacBook-Air:java drvntiwari$ javac Dynamic.java
Drs-MacBook-Air:java drvntiwari$ java Dynamic
Hypotenue is 5.0
Value of d is 15.0
Drs-MacBook-Air:java drvntiwari$
```

2.

```java
public class Javaliterals {

   public static void main(String[] args) {

       int a = 10;

       int b= 0100;

       int c= 0xFace;

       int d = 0b1111;

       System.out.println(a);

       System.out.println(b);

       System.out.println(c);

       System.out.println(d);

   }

}
```

```
Drs-MacBook-Air:java drvntiwari$ javac Javaliterals.java
Drs-MacBook-Air:java drvntiwari$ java Javaliterals
10
64
64206
15
Drs-MacBook-Air:java drvntiwari$
```

3.

```
public class Scope {

   int num=20;

   void m1()

   {int num=30;

   System.out.println("local number is " + num);

   System.out.println("global number is " + this.num);}

   public static void main(String[] args) {

    Scope st=new Scope() ;

     st.m1();



   }
}
```

```
Drs-MacBook-Air:java drvntiwari$ javac Scope.java
Drs-MacBook-Air:java drvntiwari$ java Scope
local number is 30
global number is 20
Drs-MacBook-Air:java drvntiwari$
```

EXPERIMENT 4

Aim- Different ways of taking input from user

1. **Using scanner class .**
2. **Using a buffer reader .**
3. **Using command line arguments.**

Code

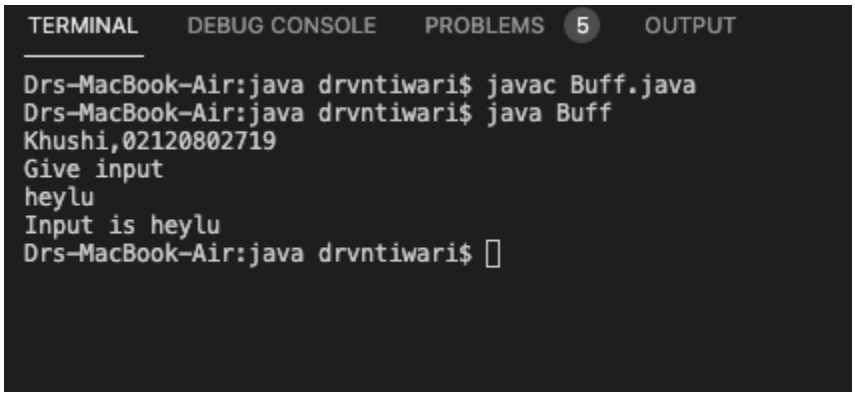
1.

```
import java.util.Scanner;

public class Scan {

   public static void main(String[] args) {

        Scanner s = new Scanner(System.in);

        System.out.println("Khushi,02120802719");

        System.out.println("Give input");

        int a= s.nextInt();

        System.out.println("Input is " + a);



   }


}
```

```
Drs-MacBook-Air:java drvntiwari$ javac Scan.java
Drs-MacBook-Air:java drvntiwari$ java Scan
Khushi,02120802719
Give input
10
Input is 10
Drs-MacBook-Air:java drvntiwari$
```

2.

```java
import java.io.*;


public class Buff {


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

       BufferedReader s = new BufferedReader(

           new InputStreamReader(System.in));

       System.out.println("Khushi,02120802719");

       System.out.println("Give input");

       String a= s.readLine();

       System.out.println("Input is " +a);


   }

}
```



3.

```
public class Cmd {

   public static void main(String[] args) {

       {

           if (args.length > 0) {

               System.out.println(

                   "The command line arguments are:");


               for (String val : args)

                   System.out.println(val);

           }

           else

               System.out.println("No command line arguments found.");

       }

   }

   }
```

```
Drs-MacBook-Air:java drvntiwari$ javac Cmd.java
Drs-MacBook-Air:java drvntiwari$ java Cmd Hey,I'm Khushi
> Hey,I'm Khushi
The command line arguments are:
Hey,Im Khushi
Hey,Im
Khushi
Drs-MacBook-Air:java drvntiwari$ java Cmd
No command line arguments found.
Drs-MacBook-Air:java drvntiwari$
```

EXPERIMENT 5

Aim- Write a program to show dynamic polymorphism.

Code

```
import java.util.*;

public class Poly extends B{

   public static void main(String[] args) {

       Scanner s = new Scanner(System.in);

       System.out.println("Khushi,02120802719");

       System.out.println("Give inputs");

       int a= s.nextInt();

       int b= s.nextInt();

       A ob ;

       B ob1 = new B(a, b);

       ob = ob1;

       ob1.print();


   }

}


class A {


   int a, b;


   A(int a, int b) {

       this.a = a;

       this.b = b;

   }
```

```
    A() {
    }

    int greater() {
        int c = a > b ? a : b;
        return c;
    }

    void print() {
        System.out.println("a :" + this.a);
        System.out.println("b :" + this.b);
    }

}

class B extends A {

    int c;

    B(int a, int b ){
        super.a = a;
        super.b = b;
    }

    B(){
    }
    int greater() {
        int c = a > b ? a : b;
        return c;
    }
```

```
   void print() {

        super.print();

        System.out.println(greater() + " is greater.");

   }

}
```

```
Drs-MacBook-Air:java drvntiwari$ javac Poly.java
Drs-MacBook-Air:java drvntiwari$
Drs-MacBook-Air:java drvntiwari$ java Poly
Khushi,02120802719
Give inputs
35 40
a :35
b :40
40 is greater.
Drs-MacBook-Air:java drvntiwari$
```

EXPERIMENT 6

Aim- Write a program to print a calendar.

Code

```
import java.util.Scanner;


public class Calendar {

	public static void main(String[] args) {

	Scanner scn = new Scanner(System.in);

	System.out.print("\nEnter full year (e.g., 2001): ");

	int year = scn.nextInt();

	System.out.print("Enter month in number between 1 and 12: ");

	int month = scn.nextInt();


	if (month < 1 || month > 12)

	    System.out.println("Wrong input!");

	else

	    printMonth(year, month);


	scn.close();

   }



   static void printMonth(int year, int month) {

	printMonthTitle(year, month);

	printMonthBody(year, month);

   }


   static void printMonthTitle(int year, int month) {
```

```
        System.out.println("\n        " + getMonthName(month) + " " + year);
        System.out.println("============================");

        System.out.println(" Sun Mon Tue Wed Thu Fri Sat");
    }
    static String getMonthName(int month) {
        String monthName = null;
        switch (month) {
            case 1:
                monthName = "January";
                break;
            case 2:
                monthName = "February";
                break;
            case 3:
                monthName = "March";
                break;
            case 4:
                monthName = "April";
                break;
            case 5:
                monthName = "May";
                break;
            case 6:
                monthName = "June";
                break;
            case 7:
                monthName = "July";
                break;
            case 8:
```

```
            monthName = "August";
            break;
        case 9:
            monthName = "September";
            break;
        case 10:
            monthName = "October";
            break;
        case 11:
            monthName = "November";
            break;
        case 12:
            monthName = "December";
    }
    return monthName;
}

static void printMonthBody(int year, int month) {

    int startDay = getStartDay(year, month);
    int numberOfDaysInMonth = getNumberOfDaysInMonth(year, month);
    int i = 0;
    for (i = 0; i < startDay; i++)
        System.out.print("    ");
    for (i = 1; i <= numberOfDaysInMonth; i++) {
        if (i < 10)
            System.out.print("   " + i);
        else
            System.out.print("  " + i);
        if ((i + startDay) % 7 == 0)
```

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

    static int getStartDay(int year, int month) {

        int startDay1800 = 3;
        int totalNumberOfDays = getTotalNumberOfDays(year, month);
        return (totalNumberOfDays + startDay1800) % 7;
    }



    static int getTotalNumberOfDays(int year, int month) {
        int total = 0;

        for (int i = 1800; i < year; i++)
            if (isLeapYear(i))
                total = total + 366;
            else
                total = total + 365;

        for (int i = 1; i < month; i++)
            total = total + getNumberOfDaysInMonth(year, i);
        return total;
    }

    static int getNumberOfDaysInMonth(int year, int month) {
        if (month == 1 || month == 3 || month == 5 || month == 7 || month == 8 || month
== 10 || month == 12)
            return 31;
```

```
        if (month == 4 || month == 6 || month == 9 || month == 11)
            return 30;
        if (month == 2)
            return isLeapYear(year) ? 29 : 28;
        return 0;
    }

    static boolean isLeapYear(int year) {
        return year % 400 == 0 || (year % 4 == 0 && year % 100 != 0);
    }
}
```

```
Drs-MacBook-Air:java drvntiwari$ javac Calendar.java
Drs-MacBook-Air:java drvntiwari$ java Calendar

Enter full year (e.g., 2001): 2001
Enter month in number between 1 and 12: 11

        November 2001
==========================
 Sun Mon Tue Wed Thu Fri Sat
                   1   2   3
   4   5   6   7   8   9  10
  11  12  13  14  15  16  17
  18  19  20  21  22  23  24
  25  26  27  28  29  30
Drs-MacBook-Air:java drvntiwari$
```

EXPERIMENT 7

Aim- Write a program to implement different kinds of inheritance.

Code

- Single Inheritance

```
import java.io.*;

class A{

   A(){

        System.out.println("Constuctor of Class A");

   }


}


class B extends A{

   B(){

      super();

       System.out.println("Constructor of class B");  }

}
public class Single {

   public static void main(String[] args) {

   B ob1=new B();  }

}
```

```
Drs-MacBook-Air:java drvntiwari$ javac Single.java
Drs-MacBook-Air:java drvntiwari$ java Single
Constuctor of Class A
Constructor of class B
Drs-MacBook-Air:java drvntiwari$
```

- MultiLevel Inheritance

```
class E {
   E() {
       System.out.println("Constuctor of Class A");
   }
}

class F extends E {
   F() {
       super();
       System.out.println("Constructor of class B");
   }
}

class C extends F{
   C() {
       super();
       System.out.println("Constructor of class C");
   }
}

public class Multi{
   public static void main(String[] args) {
       C ob1 = new C();
   }
}
```

```
Drs-MacBook-Air:java drvntiwari$ javac Multi.java
Drs-MacBook-Air:java drvntiwari$ java Multi
Constuctor of Class A
Constructor of class B
Constructor of class C
Drs-MacBook-Air:java drvntiwari$
```

- Hierarchical Inheritance

```
class X {

   X() {

       System.out.println("Constructor of Class X");

   }

}


class Y extends X {

   Y() {

       super();

       System.out.println("Constructor of class Y");

   }

}


class Z extends X{

   Z() {

       super();

       System.out.println("Constructor of class Z");

   }

}
```

```
public class Hier {

   public static void main(String[] args) {

       Z ob1 = new Z();

       Y ob2 = new Y();

   }

}
```

```
Drs-MacBook-Air:java drvntiwari$ java Hier
Constuctor of Class X
Constructor of class Z
Constuctor of Class X
Constructor of class Y
Drs-MacBook-Air:java drvntiwari$
```

- Multiple Inheritance

```
class J{

   J(){

       System.out.println("Constructor of Class J");

   }

}


interface K{

   void print();

}


class L  extends J implements K{

   L(){

      super();

       System.out.println("Constructor of class L");

   }

   @Override

   public void print() {

       System.out.println("Inside Interface K");
```

```
    }

}

public class Multiple {

    public static void main(String[] args) {

    L ob1=new L();

    ob1.print();

    }

}
```

```
Drs-MacBook-Air:java drvntiwari$ javac Multiple.java
Drs-MacBook-Air:java drvntiwari$ java Multiple
Constuctor of Class J
Constructor of class L
Inside Interface K
Drs-MacBook-Air:java drvntiwari$
```

EXPERIMENT 8

Aim- Write a program to implement interfaces.

Code

```
class Person {
   String name;
   int age;

   void display() {
       System.out.println("Name: " + name + "\nAge: " + age);
   }

   Person(int umar, String naam) {
       name = naam;
       age = umar;
   }
}

class Student extends Person {
   Student(int umar, String naam, int roll, String b) {
       super(umar, naam);
       rollno = roll;
       branch = b;
   }

   int rollno;
   String branch;
```

```
   void display() {
       super.display();
       ;
       System.out.println("Roll No: " + rollno + "\nBranch: " + branch);
   }
}

class Employee extends Person {
   Employee(int umar, String naam, int Ecn, String Dj) {
       super(umar, naam);
       Ecno = Ecn;
       Doj = Dj;
   }

   int Ecno;
   String Doj;

   void display() {
       super.display();
       System.out.println("EC no.: " + Ecno + "\nDate of joining: " + Doj);
   }
}

class Faculty extends Employee {
   Faculty(int umar, String naam, int Ecn, String Dj, String D) {
       super(umar, naam, Ecn, Dj);
       desig = D;
   }

   String desig;
```

```
   void display() {
       super.display();
       System.out.println("\nDesignation: " + desig);
   }
}
class Staff extends Employee {
   Staff(int umar, String naam, int Ecn, String Dj, String D) {
       super(umar, naam, Ecn, Dj);
       desig = D;
   }
   String desig;
   void display() {
       super.display();
       System.out.println("\nDesignation: " + desig);
   }
}
public class Interface {
   public static void main(String[] args) {
       Staff ob1 = new Staff(25, "Damon", 8001, "12/07/1845", "Vampire");
       Faculty ob2 = new Faculty(35, "Klaus", 1001, "22/06/0998", "Hybrid");
       Employee ob3 = new Employee(40, "Modfam", 4001, "10/08/2009");
       Student ob4 = new Student(19, "Khushi", 21, "CSE");
       Person ob5 = new Person(19, "Jake");
       System.out.println("\nClass Staff\n");
       ob1.display();
       System.out.println("\nClass Faculty\n");
       ob2.display();
       System.out.println("\nClass Employee\n");
       ob3.display();
```

```
        System.out.println("\nClass Student\n");

        ob4.display();

        System.out.println("\nClass Person\n");

        ob5.display();

    }

}
```

```
Drs-MacBook-Air:java drvntiwari$ javac Interface.java
Drs-MacBook-Air:java drvntiwari$ java Interface

Class Staff

Name: Damon
Age: 25
EC no.: 8001
Date of joining: 12/07/1845

Designation: Vampire

Class Faculty

Name: Klaus
Age: 35
EC no.: 1001
Date of joining: 22/06/0998

Designation: Hybrid

Class Employee

Name: Modfam
Age: 40
EC no.: 4001
Date of joining: 10/08/2009

Class Student

Name: Khushi
Age: 19
Roll No: 21
Branch: CSE

Class Person

Name: Jake
Age: 19
```

EXPERIMENT 9

Aim- Write a program to implement error handling using try catch blocks.

Code

```
import java.util.Scanner;


class CustomException extends Exception {
   String message;
   CustomException(String str) {
      message = str;
   }
   public String toString() {
      return ("Custom Exception Occurred : " + message);
   }
}
public class Ex {
   public static void main(String args[]) {
      try {
       Scanner s = new Scanner(System.in);
       System.out.println("Khushi,02120802719");
       System.out.println("Give input");
       int a= s.nextInt();
       int b= s.nextInt();
       if(a>b){
           System.out.println(a-b);
       }
       else{
         throw new CustomException("You made an error.");
```

```
        }

    }

        catch(CustomException e) {

            System.out.println(e);

        }

    }
}
```

```
Drs-MacBook-Air:java drvntiwari$ javac Ex.java
Drs-MacBook-Air:java drvntiwari$ java Ex
Khushi,02120802719
Give input
4
3
1
Drs-MacBook-Air:java drvntiwari$ java Ex
Khushi,02120802719
Give input
5
6
Custom Exception Occurred : You made an error.
Drs-MacBook-Air:java drvntiwari$
```

EXPERIMENT 10

Aim- Write a program to print a calculator using multithreading.

Code

```
import java.util.Scanner;
class Add extends Thread{
   int a,b;
   Add(int x,int y){
       a=x;
       b=y;
   }
   public void run(){

       System.out.println(a+b);
   }
}
class Sub extends Thread{
   int a,b;
   Sub(int x,int y){
       a=x;
       b=y;
   }
   public void run(){

       System.out.println(a-b);
   }
}
class Mul extends Thread{
   int a,b;
```

```
   Mul(int x,int y){
       a=x;
       b=y;
   }
   public void run(){

       System.out.println(a*b);
   }
}
class Div implements Runnable{
   float a,b;
   Div(int x,int y){
       a=x;
       b=y;
   }
   @Override public void run(){

       System.out.println(a/b);
   }
}
class Power implements Runnable{
   int a,b;
   Power(int x,int y){
       a=x;
       b=y;
   }
   @Override public void run(){
       System.out.println(Math.pow(a, b));
   }
}
```

```
class Rem implements Runnable{
   int a,b;
   Rem(int x,int y){
       a=x;
       b=y;
   }
   @Override public void run(){
       System.out.println(a%b);
   }
}
public class Calculator{
   public static void main(String[] args) {
       Scanner scn = new Scanner(System.in);
       System.out.println("ENTER TWO VALUES FOR ARITHMETIC  OPERATIONS");
       int a= scn.nextInt();
       int b= scn.nextInt();
       Add ob1=new Add(a,b);
       Sub ob2=new Sub(a,b);
       Mul ob3=new Mul(a,b);
       Div ob4=new Div(a,b);
       Power ob5=new Power(a,b);
       Rem ob6=new Rem(a,b);
       ob1.start();
       ob2.start();
       ob3.start();
       ob4.run();
       ob5.run();
       ob6.run();
   }
}
```

```
Drs-MacBook-Air:java drvntiwari$ java Calculator
Drs-MacBook-Air:java drvntiwari$ javac Calculator.java
Drs-MacBook-Air:java drvntiwari$ java Calculator
ENTER TWO VALUES FOR ARITHEMATIC  OPERATIONS
4
2
6
2.0
16.0
0
8
2
Drs-MacBook-Air:java drvntiwari$ java Calculator
ENTER TWO VALUES FOR ARITHEMATIC  OPERATIONS
5
2
3
10
2.5
25.0
1
7
```