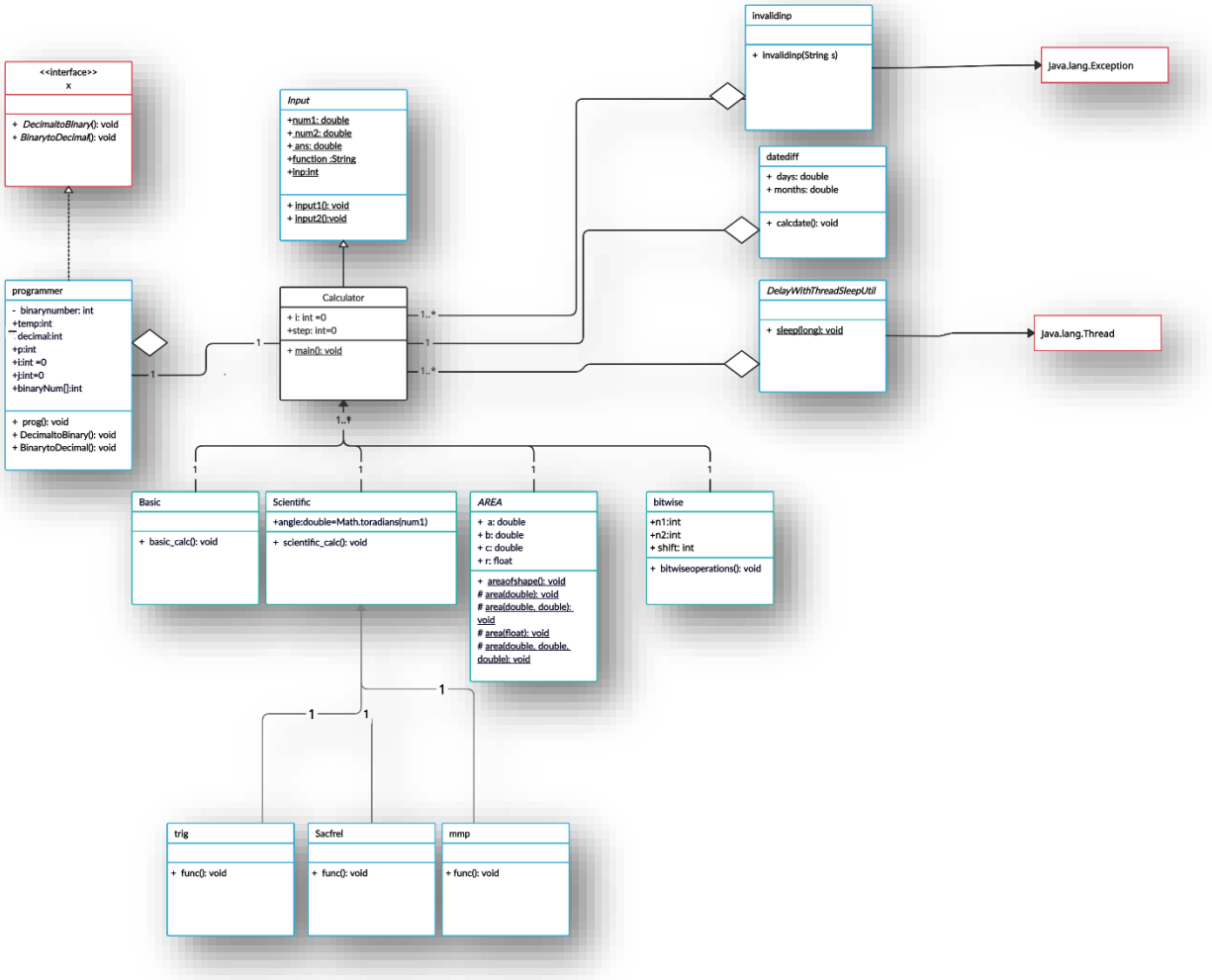
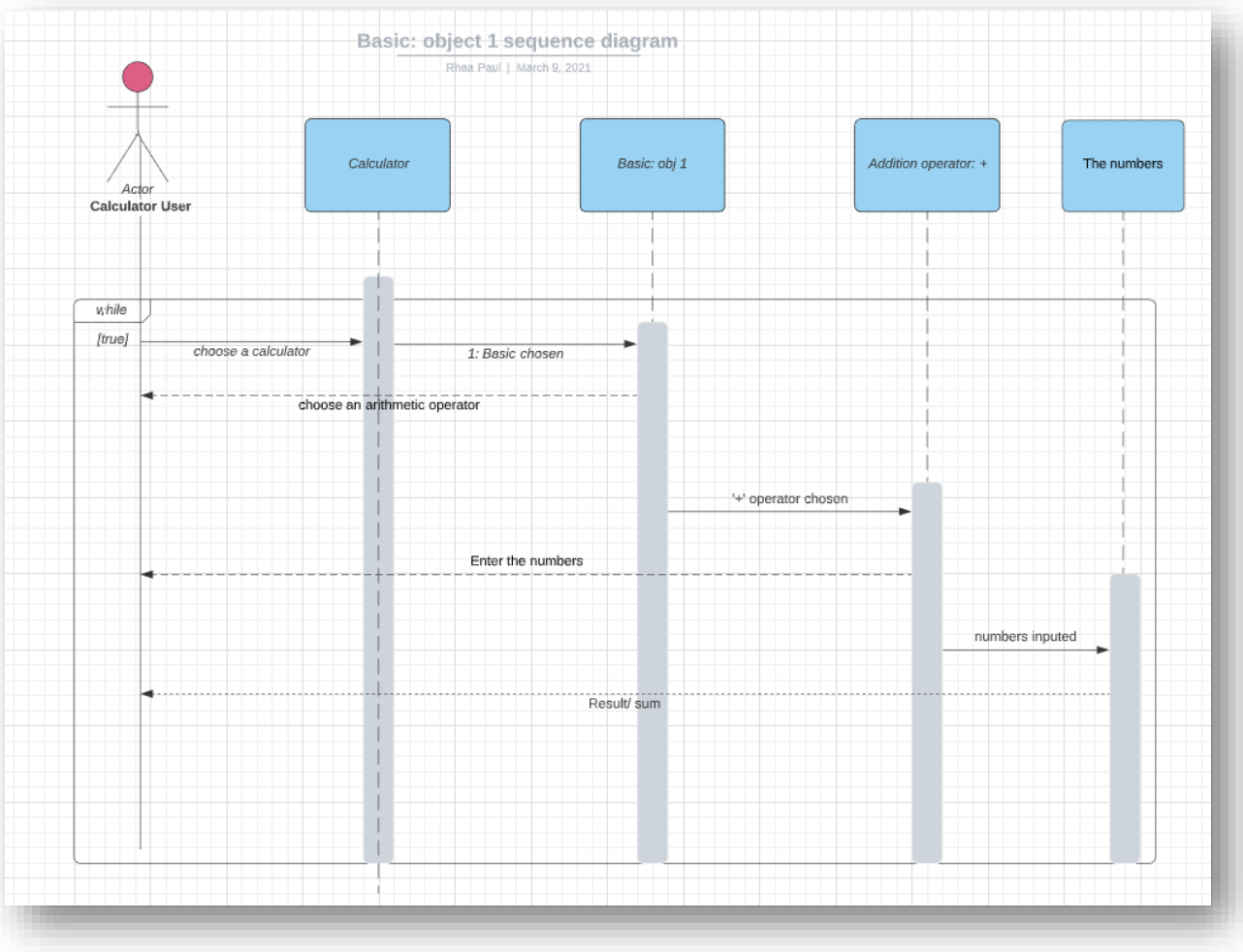
Index

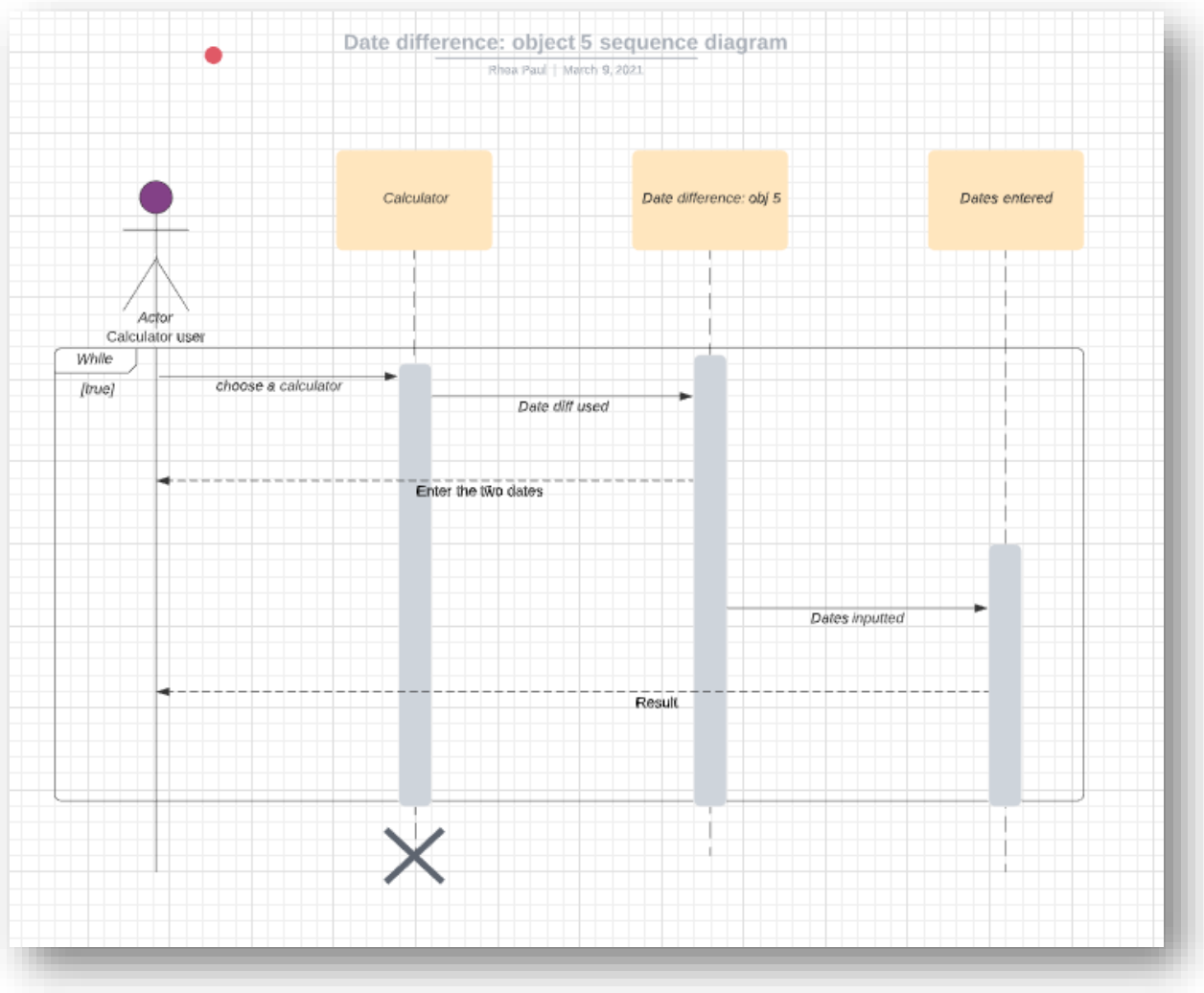
UML class diagram



Sequence diagram



Sequence diagram



Outputs for each case

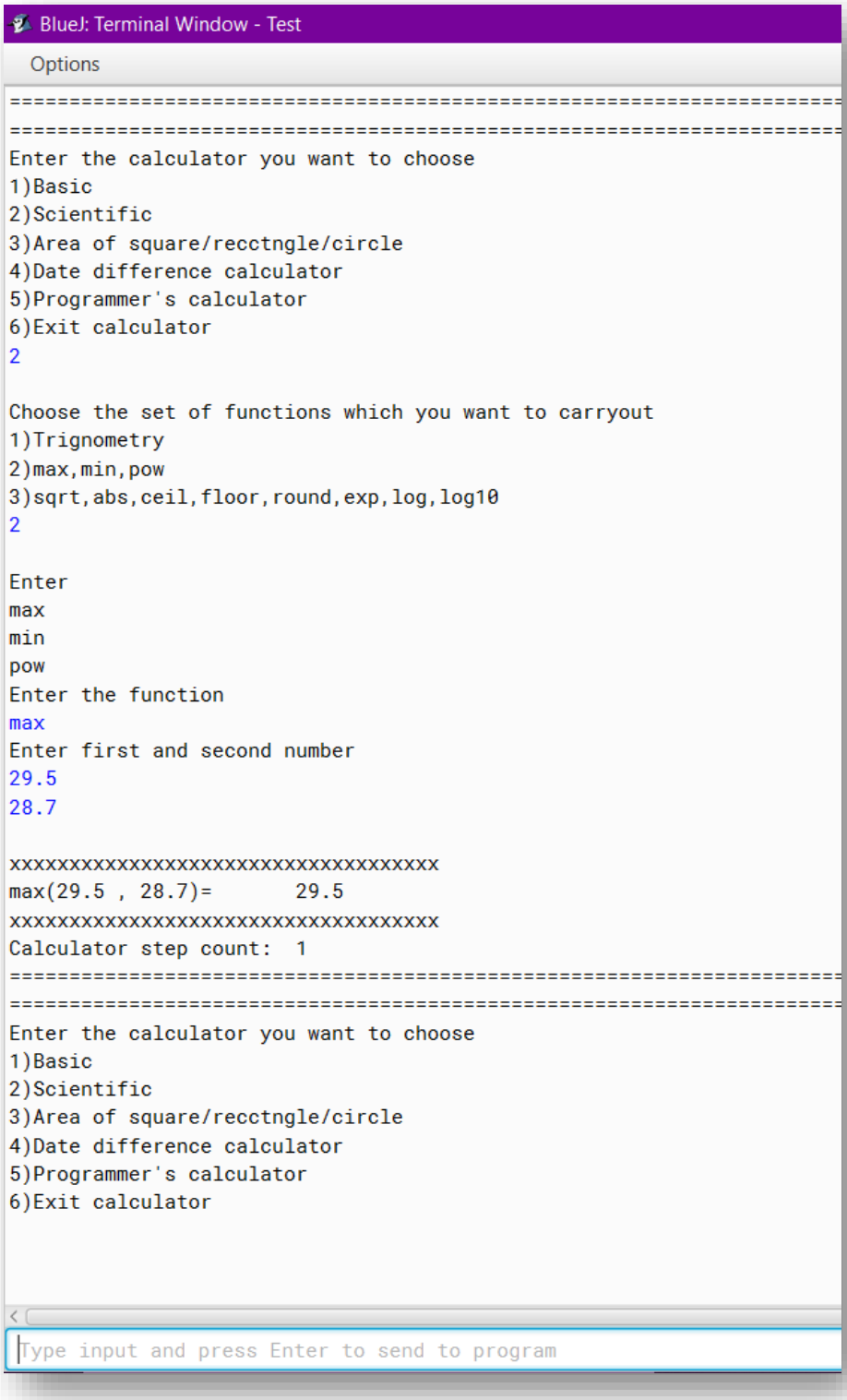
1. Basic calculator

BlueJ: Terminal Window - Test

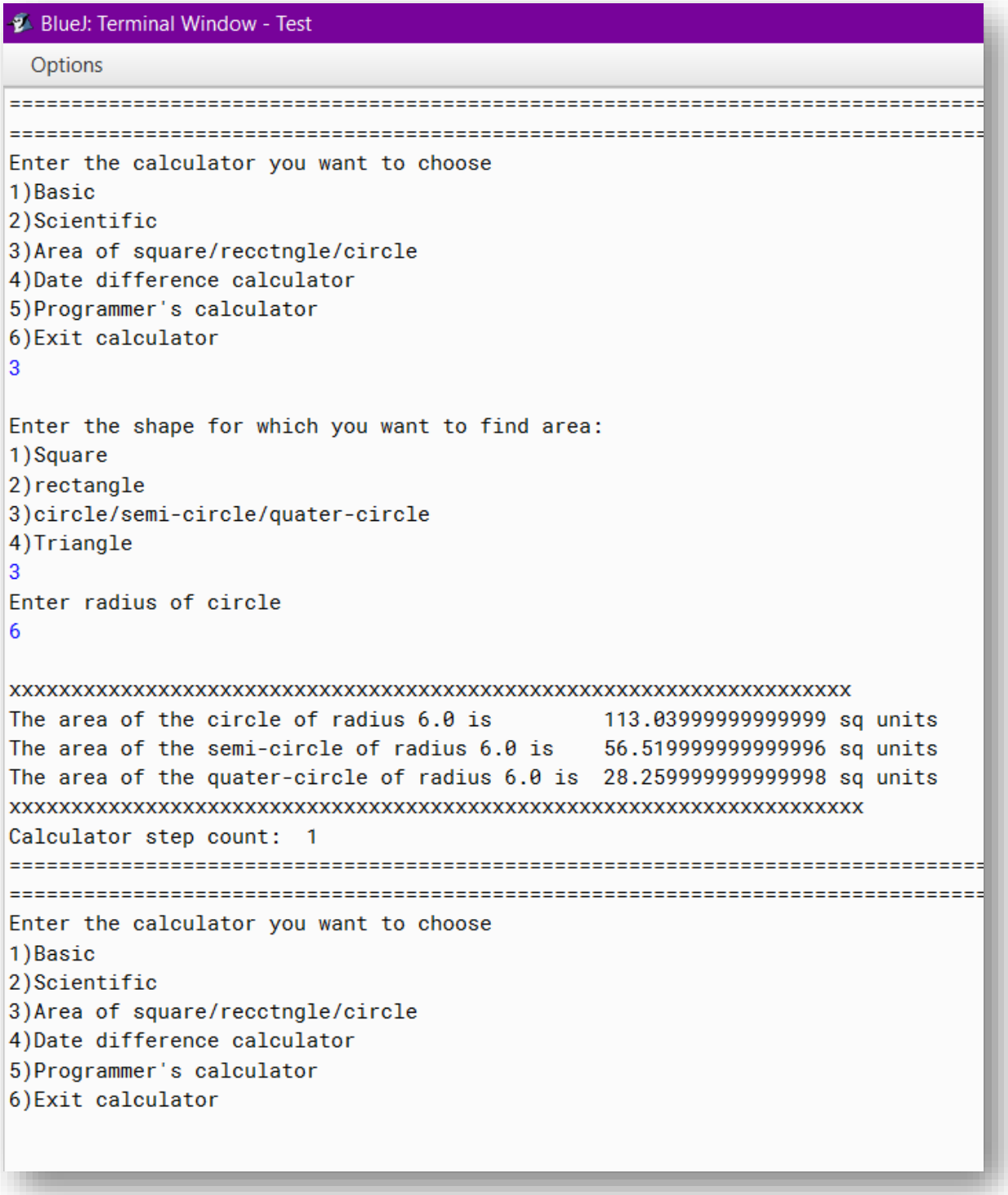
Options

```
================================================================================
================================================================================
Enter the calculator you want to choose
1)Basic
2)Scientific
3)Area of square/recctngle/circle
4)Date difference calculator
5)Programmer's calculator
6)Exit calculator
1

Enter arithmethic operator
+       -       *       /       %
Enter the function
+
Enter first and second number
20
25
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

20.0
+
25.0
=
45.0
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Calculator step count:  1
================================================================================
================================================================================
Enter the calculator you want to choose
1)Basic
2)Scientific
3)Area of square/recctngle/circle
4)Date difference calculator
5)Programmer's calculator
6)Exit calculator
```

2. Scientific calculator



3. area calculator



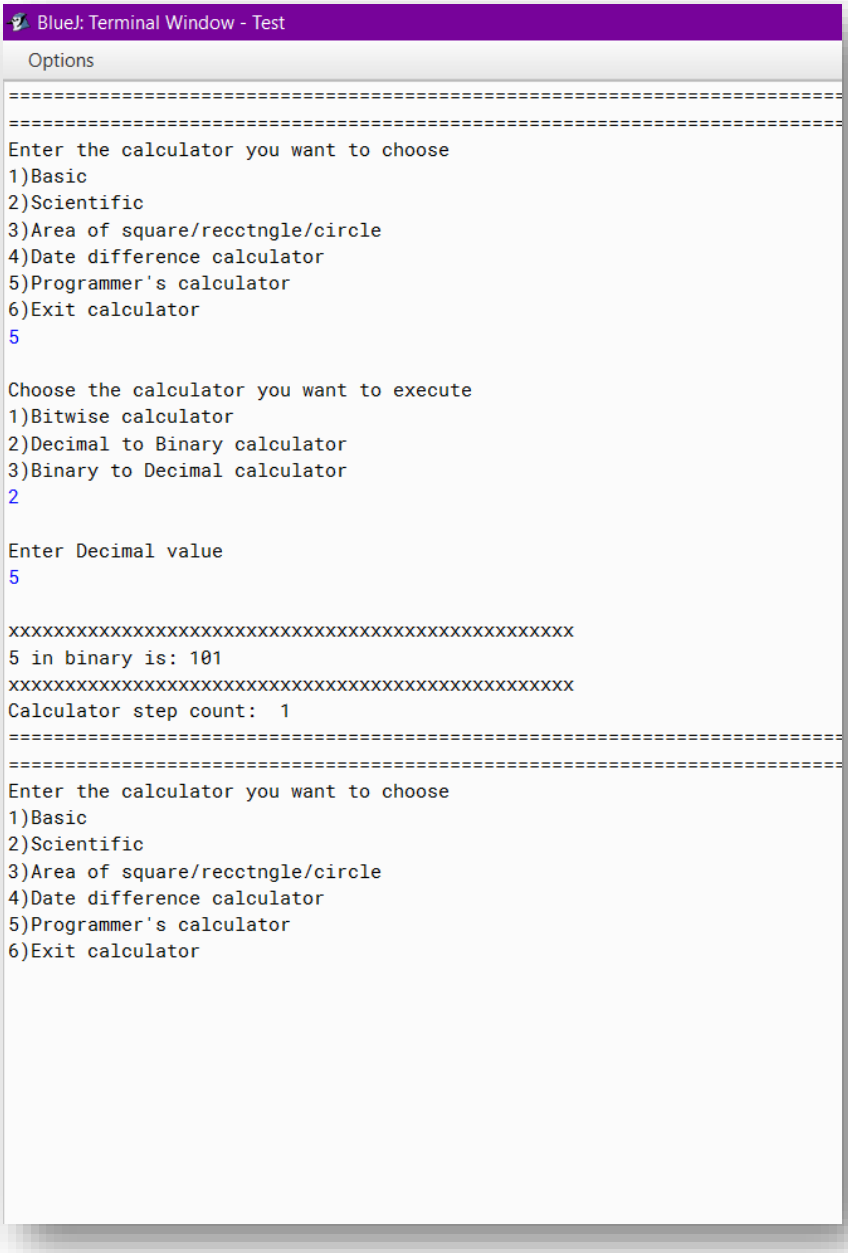
4.date difference calculator

BlueJ: Terminal Window - Test

Options

```
================================================================================
================================================================================
Enter the calculator you want to choose
1)Basic
2)Scientific
3)Area of square/recctngle/circle
4)Date difference calculator
5)Programmer's calculator
6)Exit calculator
4

Enter first date        format:dd(enter)mm(enter)yy
22
02
2002
Enter second date       format:dd(enter)mm(enter)yy
25
07
2020
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
days between the two dates are:         6725.5
months between the two dates are:       221.0
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Calculator step count:  1
================================================================================
================================================================================
Enter the calculator you want to choose
1)Basic
2)Scientific
3)Area of square/recctngle/circle
4)Date difference calculator
5)Programmer's calculator
6)Exit calculator
```

5. programmers calculator



Code

```
package EndSemProj;
import java.util.Scanner;
public class calculator extends input
{
   public static void main(String[] args)
   {
      Scanner reader=new Scanner(System.in);
      int i=0,step=0;
      basic obj1=new basic();
      scientific obj2=new scientific();

      datediff obj5=new datediff();
      programmer obj9=new programmer();
      while (true)
      {
         System.out.println
("=================================================
==================================================
==================================================
==================================================");
         System.out.println
("=================================================
==================================================
==================================================
==================================================");
         System.out.println ("Enter the calculator you want to
choose\n1)Basic\n2)Scientific\n3)Area of square/recctngle/circle\n4)Date
difference calculator\n5)Programmer's calculator\n6)Exit calculator");
         inp=reader.nextInt();
         try
         {
            if(inp>6||inp<1)
            {
                  throw new invalidinp("\nYour input has to be between 1 and 6");
            }
            switch(inp)
            {
               case 1:obj1.basic_calc();
                  break;
```

```
                case 2:obj2.scientific_calc();
                  break;
                case 3:AREA.areaofshape();
                      break;
                case 4:obj5.calcdate();
                      break;
                case 5:obj9.prog();
                      break;
                case 6:
System.out.println("\n\n**********************************************
******");
                      System.out.println("**                                    **");
                      System.out.print("**\tThank you for using the calclator ");
                      System.out.println("\t**");
                      System.out.print("**\tTotal steps:\t"+step);
                      System.out.println("                **");
                      System.out.println("**                                    **");

System.out.println("**********************************************
**");
                      System.exit(0);
                      break;
              }
          }
          catch(invalidinp t)
          {
             System.out.println(t);
          }
          finally
          {
             System.out.println("Calculator step count:\t"+ ++step);
          }
          i++;
          if (i%2==0)
          {
              DelayWithThreadSleepUtil.sleep(15000);
              System.out.print('\u000C');
          }
          }
       }
}
abstract class input
```

```
{
   public static double num1,num2,ans;
   public static String function;
   static int inp;
   public  static void input1()
   {
        Scanner sc1=new Scanner(System.in);
        System.out.println("Enter the function");
        function=sc1.nextLine();
        System.out.println("Enter the number");
        num1 =sc1.nextDouble();
   }
   public  static  void input2()
   {
      Scanner sc2=new Scanner(System.in);
      System.out.println("Enter the function");
      function= sc2.next();
      System.out.println("Enter first and second number");
      num1 = sc2.nextDouble();
      num2 = sc2.nextDouble();
   }
}
class basic extends calculator
{
   public  void basic_calc()
     {
    System.out.println ("\nEnter arithmethic operator\n+\t-\t*\t/\t%");
         input.input2();
    switch(function)
    {
      case "+": ans = num1 + num2;
        break;
      case "-": ans = num1 - num2;
        break;
      case "*": ans = num1 * num2;
        break;
      case "/": ans = num1 / num2;
        break;
      case "%": ans = num1 % num2;
        break;
      default: System.out.println("Error! Enter correct operator\n");
    }
```

```
      System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
      System.out.println("\n"+num1+"\n"+function+"\n"+num2+ "\n=\n"+ans);
      System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
       }
}
class scientific extends calculator
{
   public void scientific_calc()
   {
      Scanner reader1=new Scanner(System.in);
      System.out.println ("\nChoose the set of functions which you want to
carryout\n1)Trignometry\n2)max,min,pow\n3)sqrt,abs,ceil,floor,round,exp,log,l
og10");
      inp=reader1.nextInt();
      trig trigobj=new trig();
      mmp mmpobj=new mmp();
      sacfrel sacfrelobj=new sacfrel();
      try
         {
            if(inp>3||inp<1)
            {
                  throw new invalidinp("\nYour input has to be between 1 and 3");
            }
         }
         catch(invalidinp t)
         {
            System.out.println(t);
         }
      switch(inp)
      {
         case 1:trigobj.func();
        break;
         case 2:mmpobj.func();
        break;
         case 3:sacfrelobj.func();
        break;
      }
   }
}
Abstract class AREA extends calculator
{
   public static void areaofshape()
```

```
    {
        double a,b,c;
        float r;
        System.out.println("\nEnter the shape for which you want to find
area:\n1)Square\n2)rectangle\n3)circle/semi-circle/quater-circle\n4)Triangle");
        Scanner reader2=new Scanner(System.in);
        inp=reader2.nextInt();
        switch(inp)
        {
            case 1:System.out.println("Enter length of side of square");
                   a=reader2.nextDouble();
                   AREA.area(a);
                    break;
            case 2:System.out.println("Enter length and breadth of rectangle");
                   a=reader2.nextDouble();
                   b=reader2.nextDouble();
                   AREA.area(a,b);
                    break;
            case 3:System.out.println("Enter radius of circle");
                   r=reader2.nextFloat();
                   AREA.area(r);
                    break;
            case 4:System.out.println("Enter the three sides of a triangle");
                   a=reader2.nextDouble();
                   b=reader2.nextDouble();
                   c=reader2.nextDouble();
                   AREA.area(a,b,c);
                    break;
            default: System.out.println("Wrong input");
        }
    }
        protected static void area(double a)
        {

System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxx");
            System.out.println("the area of the square of length "+a+" is
"+Math.pow(a, 2)+" sq units");

System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx");
        }
```

```
    protected static void area(double a, double b)
    {

System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
        System.out.println("The area of the rectangle of length "+a+" and
breadth "+b+" is "+a*b+" sq units");

System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
    }
    protected static void area(float r)
    {

System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxx");
        System.out.println("The area of the circle of radius "+r+" is\t\t"+(3.14 *
r * r)+" sq units");
        System.out.println("The area of the semi-circle of radius "+r+" is\t"
+((3.14 * r * r)/2)+" sq units");
        System.out.println("The area of the quarter-circle of radius "+r+" is\t"
+((3.14 * r * r)/4)+" sq units");

System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxx");
    }
    protected static void area(double a, double b, double c)
    {
       double temp = (a+b+c);
       double s= temp/2;
       double triarea = Math.sqrt(s*(s-a)*(s-b)*(s-c));

System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
        System.out.println( "Area of triangle with length of sides  "+a+"," +b+ "
and " +c+" is : "+ triarea);

System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
    }
}
class datediff
```

```
{
   public void calcdate()
   {
         double days,months;
         Scanner reader3=new Scanner(System.in);
         System.out.println("\nEnter first date \tformat:dd(enter)mm(enter)yy");
         double d1=reader3.nextDouble();
         double m1=reader3.nextDouble();
         double y1=reader3.nextDouble();
         System.out.println("Enter second date \tformat:dd(enter)mm(enter)yy");
         double d2=reader3.nextDouble();
         double m2=reader3.nextDouble();
         double y2=reader3.nextDouble();
         days=Math.abs(y2-y1)*365+Math.abs(m2-m1)*30.5+Math.abs(d2-d1);
         months=Math.abs(y2-y1)*12+Math.abs(m2-m1);
         if (d1>31||d2>31||m1>12||m2>12||d1<0||d2<0||m1<1||m2<1)
         System.out.println("\nincorrect date input please try again");
         else
         {

System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx");
                  System.out.println("days between the two dates are:\t\t"+days);
               System.out.println("months between the two dates are:\t"+months);

System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx");
            }
         }
}
abstract class DelayWithThreadSleepUtil extends Thread
{
   public static void sleep(long millies) {
      try {
         Thread.sleep(millies);
      } catch (InterruptedException e) {
         System.out.println("Thread is interrupted");
         Thread.currentThread().interrupt();
      }
   }
}
interface x
```

```
{
   abstract public void DecimaltoBinary();
   abstract public void BinarytoDecimal();
}
class bitwise extends calculator
{
      public void bitwiseoperations()
   {
      System.out.println ("Enter operation\n&\t|\t^\t~\t<<\t>>\t>>>");
           System.out.println ("Enter the function");
           Scanner reader4=new Scanner(System.in);
           function=reader4.next();
           int n1,n2;
      switch(function)
           {
             case "&":
               case "^":
             case "|":System.out.println("Enter the two numbers");
                    n1=reader4.nextInt();
             n2=reader4.nextInt();
             switch(function)
                  {
                   case "&": ans = n1 & n2;
                          break;
                   case "|": ans = n1 | n2;
                   break;
               case "^": ans = n1 ^ n2;
                   break;
                  default: System.out.println("Error!\n");
            }

System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
              System.out.println(n1+""+function+""+n2+ "=\t"+ans);

System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
              break;
             case "~":
             case "<<":
             case ">>":
         case ">>>": System.out.println("Enter the number");
```

```
                n1=reader4.nextInt();
                    switch(function)
                {
                    case "~": ans =~n1;

System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx");
                                System.out.println(function+""+n1+"=\t"+ans);

System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx");
                                break;
                            case "<<":
                            case ">>":
                        case ">>>": System.out.println("Enter number to be shifted by");
                            int shift=reader4.nextInt();
                                switch(function)
                                {
                                case "<<": ans = n1<<shift;

                                            break;
                                 case ">>": ans = n1>>shift;

                                    break;
                                case ">>>": ans = n1>>>shift;

                                        break;
                                    default: System.out.println("Error!\n");
                                }

System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxx");
                                System.out.println(n1+""+function+""+shift+"=\t"+ans);

System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xx");
                                break;
                            default: System.out.println("Error!\n");
                        }
                break;
                    default: System.out.println("Error!\n");
                }
```

```
    }
}
class programmer implements x
{
    public int binaryNumber;
    public void prog()
    {
        System.out.println("\nChoose the calculator you want to
execute\n1)Bitwise calculator\n2)Decimal to Binary calculator\n3)Binary to
Decimal calculator");
        Scanner reader7=new Scanner(System.in);
        int input1=reader7.nextInt();
        bitwise bitwiseObject=new bitwise();
        programmer obj=new programmer();
        switch(input1)
        {
          case 1:bitwiseObject.bitwiseoperations();
                 break;
          case 2:obj.DecimaltoBinary();
                  break;
          case 3:obj.BinarytoDecimal();
                  break;
        }
    }
    public void BinarytoDecimal()
    {
        System.out.println("\nEnter binary");
        Scanner reader5=new Scanner(System.in);
        binaryNumber=reader5.nextInt();

System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxx");
        System.out.print(binaryNumber+" in decimal is:\t");
        int decimal = 0;
        int p = 0;
        while(true)
        {
          if(binaryNumber == 0)
          {
             break;
          }
          else
```

```
        {
            int temp = binaryNumber%10;
            decimal += temp*Math.pow(2, p);
            binaryNumber = binaryNumber/10;
            p++;
         }
       }
       System.out.println(decimal);

System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
    }
    public void DecimaltoBinary()
    {
       Scanner reader6=new Scanner(System.in);
       System.out.println("\nEnter Decimal value");
       int decimal=reader6.nextInt();
       int[] binaryNum = new int[100];
       int i = 0;

System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
       System.out.print(decimal+" in binary is:\t");
       while (decimal > 0)
       {
          binaryNum[i] = decimal % 2;
          decimal = decimal / 2;
          i++;
       }
       for (int j = i - 1; j >= 0; j--)
          System.out.print(binaryNum[j]);

System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
    }
}
class trig extends scientific
{
    public void func()
    {
```

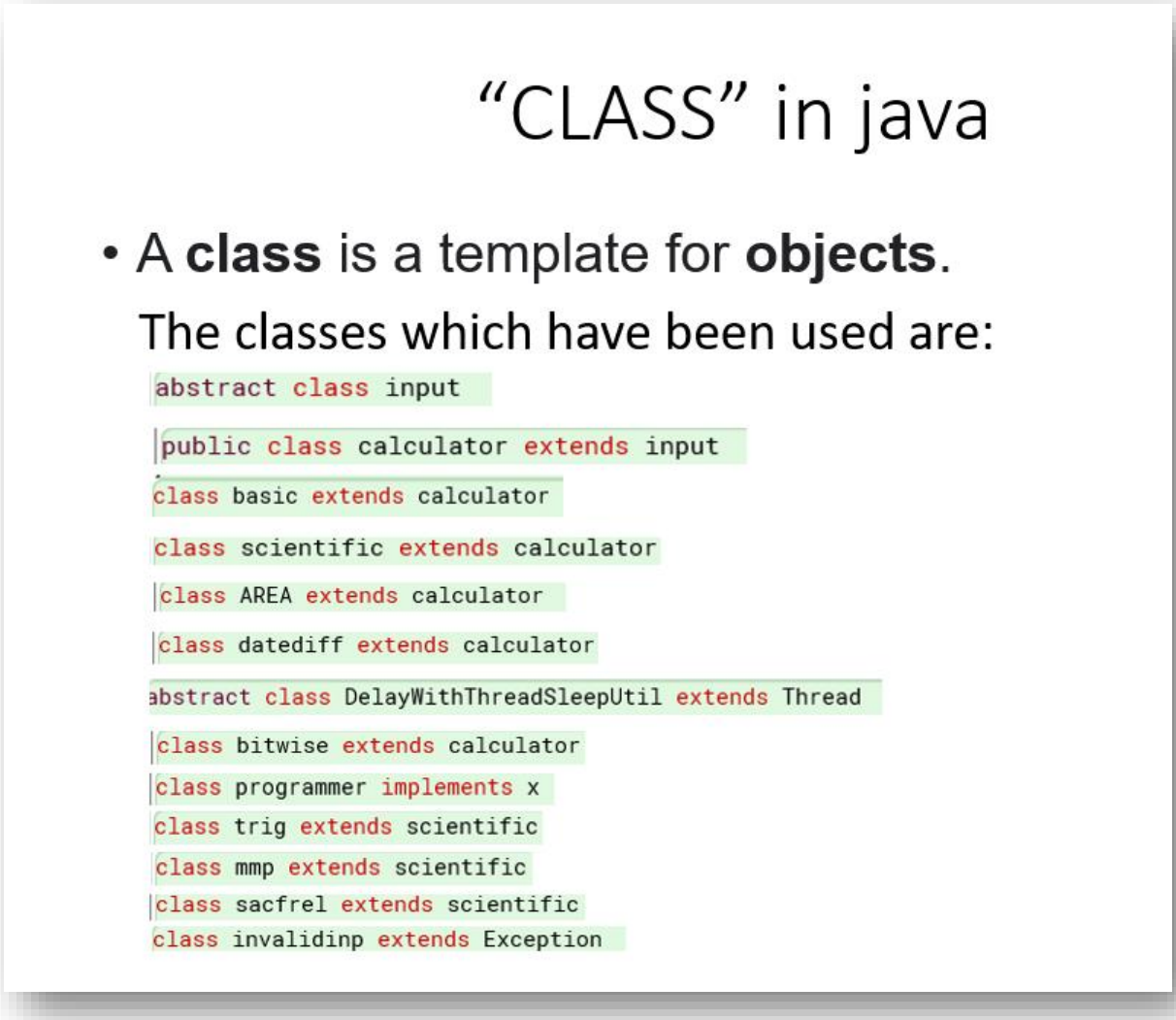
```
      System.out.println ("\nEnter which
function\nsin(degree)\tcos(degree)\ttan(degree)\tcosec(degree)\tsec(degree)\tcot
(degree)\ttoradian(degree)\ttodegree(radian)");
        input.input1();
        double angle=Math.toRadians(num1);
      switch(function)
      {

         case "sin": ans = Math.sin(angle);
            break;
         case "cos": ans = Math.cos(angle);
            break;
         case "tan": ans = Math.tan(angle);
            break;
         case "cosec": ans = 1/(Math.sin(angle));
            break;
        case "sec": ans = 1/(Math.cos(angle));
            break;
        case "cot": ans = 1/(Math.tan(angle));
            break;
         case "todegree": ans = Math.toDegrees(num1);
            break;
        case "toradian": ans = angle;
            break;
         default: System.out.println("Error! Enter correct operator\n\n");
      }
      System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
      System.out.println(function+"("+num1+")=\t"+ans);
      System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
    }
}
class mmp extends scientific
{
   public void func()
   {
      System.out.println ("\nEnter\nmax\nmin\npow");
      input.input2();
      switch(function)
        {
          case "max": ans = Math.max(num1,num2);
             break;
          case "min": ans = Math.min(num1,num2);
```

```
            break;
          case "pow": ans = Math.pow(num1,num2);
            break;
         default: System.out.println("Error! Enter correct operator");
       }
      System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
      System.out.println(function+"("+num1+" , "+num2+")=\t"+ans);
      System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
    }
}
class sacfrel extends scientific
{
   public void func()
   {
      System.out.println ("\nEnter
operation\nsqrt\tabs\tceil\tfloor\tround\texp\tlog\tlog10 ");
       input.input1();
     switch(function)
     {
       case "sqrt": ans = Math.sqrt(num1);
          break;
       case "abs": ans = Math.abs(num1);
          break;
       case "ceil": ans = Math.ceil(num1);
          break;
       case "floor": ans = Math.floor(num1);
          break;
      case "round": ans = Math.round(num1);
          break;
      case "exp": ans = Math.exp(num1);
          break;
      case "log": ans = Math.log(num1);
          break;
      case "log10": ans = Math.log10(num1);
          break;
       default: System.out.println("\n\nError! Enter correct operator");
     }
      System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
      System.out.println(function+"("+num1+")=\t"+ans);
      System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
    }
}
```

```
class invalidinp extends Exception
{
   public invalidinp(String s)
   {
      super(s);
   }
}
```

Concepts used are:

Class



Object

"OBJECTS" in java

- An entity that has state and behavior is known as an object
- An object is an instance of a class
- An object has three characteristics:
    1. State.
    2. Behavior
    3. Identity

```
basic obj1=new basic();
scientific obj2=new scientific();
AREA obj3=new AREA();
datediff obj5=new datediff();
programmer obj9=new programmer();
bitwise bitwiseObject=new bitwise();
programmer obj=new programmer();
trig trigobj=new trig();
mmp mmpobj=new mmp();
sacfrel sacfrelobj=new sacfrel();
```

Access specifier

"Access Specifiers" in java

1. private
2. default
3. Protected
4. public

But, the classes and interfaces themselves can have only two access specifiers when declared outside any other class.

- 1) public
- 2) default

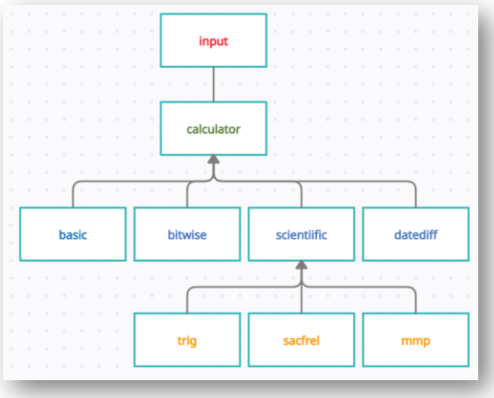
```
protected static void area(double a)
protected static void area(double a, double b)
protected static void area(float r)
protected static void area(double a, double b, double c)
public class calculator extends input

private int binaryNumber;
```

- Protected access specifier has been used for the method are which is undergoing method overloading
- Most methods and classes are public in the program
- Variable binaryNumber has private accessspecifier

Inheritance

“Inheritance” in java

- Inheritance is an important pillar of OOP. It is the mechanism in java by which one class can inherit the features of another class.



The program uses multiple, hierarchical inheritance

Interface

“Interface” in java

An interface is a reference type in Java. It is similar to class. It is a collection of abstract methods. A class implements an interface, thereby inheriting the abstract methods of the interface.

```
interface x
{
    abstract public void DecimaltoBinary();
    abstract public void BinarytoDecimal();
}

class programmer implements x

public void BinarytoDecimal()
public void DecimaltoBinary()
```

In the program, class programmer implements interface “x” thereby inheriting the abstract methods BinarytoDecimal and DecimaltoBinary

Exception handling

"Exception Handling" in java

- When an error occurs, Java will normally stop and generate an error message
- Errors/Exceptions can be handled using try and catch block
- Finally keyword lets us execute a statement regardless of error

```
try
{
    if(inp>8||inp<1)
    {
        throw new invalidinp("\nYour input has to be between 1 and 8");
    }
catch(invalidinp t)
{
    System.out.println(t);
}
finally
{
    System.out.println("Calculator step count:\t"+ ++step);
}
```

```
try
{
    if(inp>3||inp<1)
    {
        throw new invalidinp("\nYour input has to be between 1 and 3");
    }
}
catch(invalidinp t)
{
    System.out.println(t);
}
class invalidinp extends Exception
{
    public invalidinp(String s)
    {
        super(s);
    }
}
```

- We use exception handling to prevent error on invalid input by user while selecting an option to choose from
- Finally used to increment the step counter

Method overriding

"Method overriding" in java

Method overriding is one of the way by which java achieve Run Time Polymorphism. The version of a method that is executed will be determined by the object that is used to invoke it. If an object of a parent class is used to invoke the method, then the version in the parent class will be executed, but if an object of the subclass is used to invoke the method, then the version in the child class will be executed.

```
trig trigobj=new trig();
mmp mmpobj=new mmp();
sacfrel sacfrelobj=new sacfrel();
```

```
switch(inp)
{
    case 1:trigobj.func();
    break;
    case 2:mmpobj.func();
    break;
    case 3:sacfrelobj.func();
    break;
}
```

Case 1 executes trigonometry functions after method call
Case 2 executes max,min,pow functions after method call
Case 3 executes sqrt,abs,ceil,floor,round function after method call

Super

“Super keyword” in java

- Super Is used to call the method or constructor of parent class.
- In this case it used to call the constructor of the parent class ie. Exception class

```
class invalidinp extends Exception
{
    public invalidinp(String s)
    {
        super(s);
    }
}
```

Method overloading (polymorphism)

Method Overloading

Method overloading allows different methods to have the same name, but different signatures where the signature can differ by the number of input parameters or type of input parameters or both. Overloading is related to compile-time (or static) polymorphism.

```
protected static void area(double a)
{
    System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
    System.out.println("the area of the square of length "+a+" is "+Math.pow(a, 2)+" sq units");
    System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
}

protected static void area(double a, double b)
{
    System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
    System.out.println("The area of the rectangle of length "+a+" and breadth "+b+" is "+a*b+" sq units");
    System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
}

protected static void area(float r)
{
    System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
    System.out.println("The area of the circle of radius "+r+" is\t\t"+(3.14 * r * r)+" sq units");
    System.out.println("The area of the semi-circle of radius "+r+" is\t" +((3.14 * r * r)/2)+" sq units");
    System.out.println("The area of the quater-circle of radius "+r+" is\t" +((3.14 * r * r)/4)+" sq units");
    System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
}

protected static void area(double a, double b, double c)
{
    double temp = (a+b+c);
    double s= temp/2;
    double triarea = Math.sqrt(s*(s-a)*(s-b)*(s-c));
    System.out.println("\nxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
    System.out.println( "Area of triangle with lenght of sides  "+a+"," +b+ " and " +c+" is : "+ triarea);
    System.out.println("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
}
```

Method “area” is being overloaded for each different shape area is to be calculated for.

Threads

Threads

- Threads allows a program to operate more efficiently by doing multiple things at the same time.
- Threads can be used to perform complicated tasks in the background without interrupting the main program.
- We use the method sleep(long millisecond) to make the running thread to block for at least the specified number of milliseconds.

```
abstract class DelayWithThreadSleepUtil extends Thread
{
    public static void sleep(long millies)
    {
        try {
            Thread.sleep(millies);
        }
        catch (InterruptedException e)
        {
         System.out.println("Thread has been interupted");
        }
    }
}
```

```
if (i%2==0)
{
    DelayWithThreadSleepUtil.sleep(15000);
    System.out.print('\u000C');
}
```

Constructors

Constructors

- A constructor in Java is a special method that is used to initialize objects.
- The constructor is called when an object of a class is created.
- It can be used to set initial values for object attributes:

```
class invalidinp extends Exception
{
    public invalidinp(String s)
    {
        super(s);
    }
}
```

Invalidinp is the constructor used for exception handling