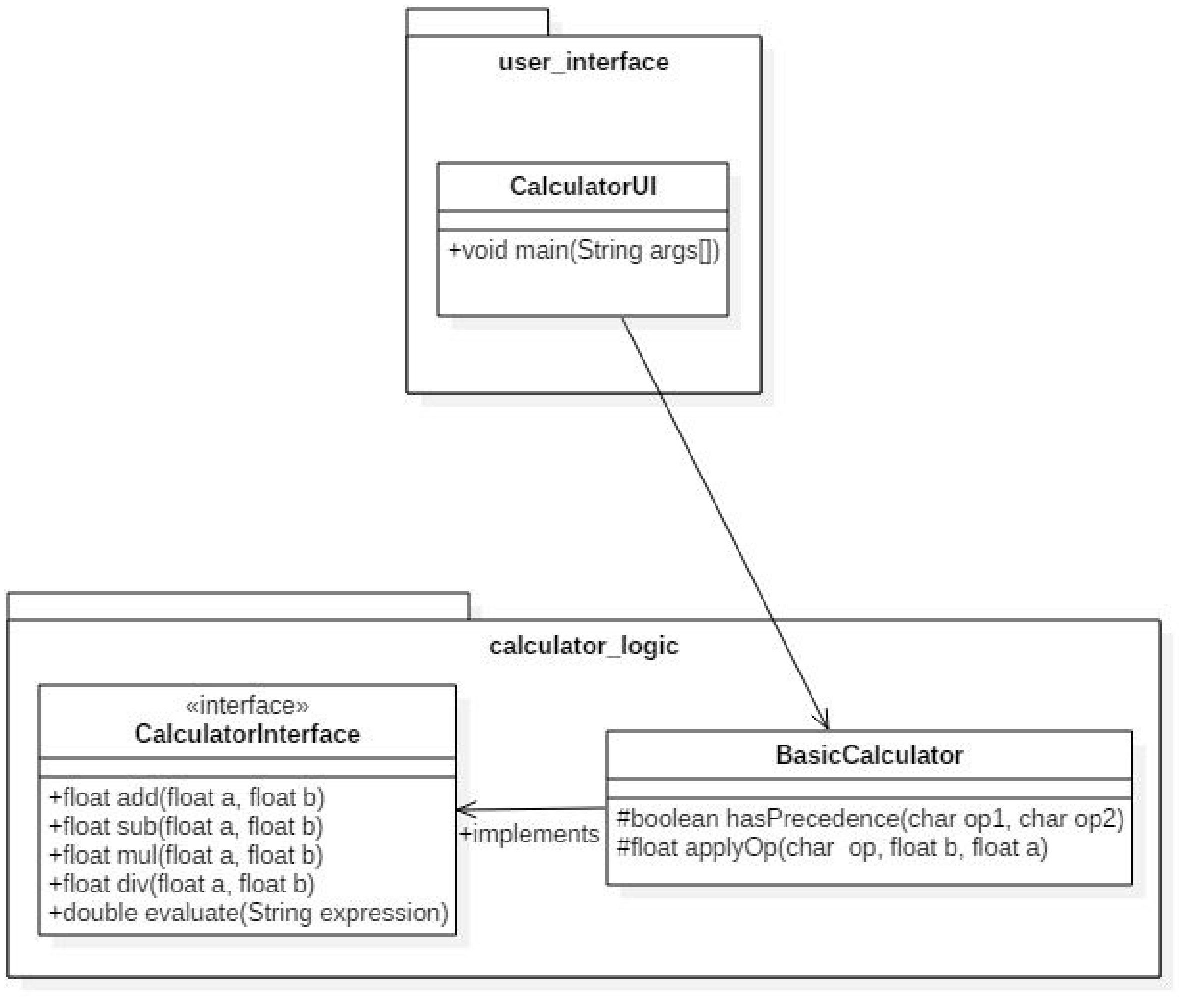
161080974_prashant 161080975_lalbahadur 161081981_freya

Tutorial No. 1

Q.1 implement calculator program using basic operations addition,subtraction,multiplication,division based on monolithic and client-server architecture.

Class diagram:

Monolithic:


user_interface
CalculatorUI
+void main(String args[])
calculator_logic
«interface»
CalculatorInterface
+float add(float a, float b)
+float sub(float a, float b)
+float mul(float a, float b)
+float div(float a, float b)
+double evaluate(String expression)
+implements
BasicCalculator
#boolean hasPrecedence(char op1, char op2)
#float applyOp(char op, float b, float a)

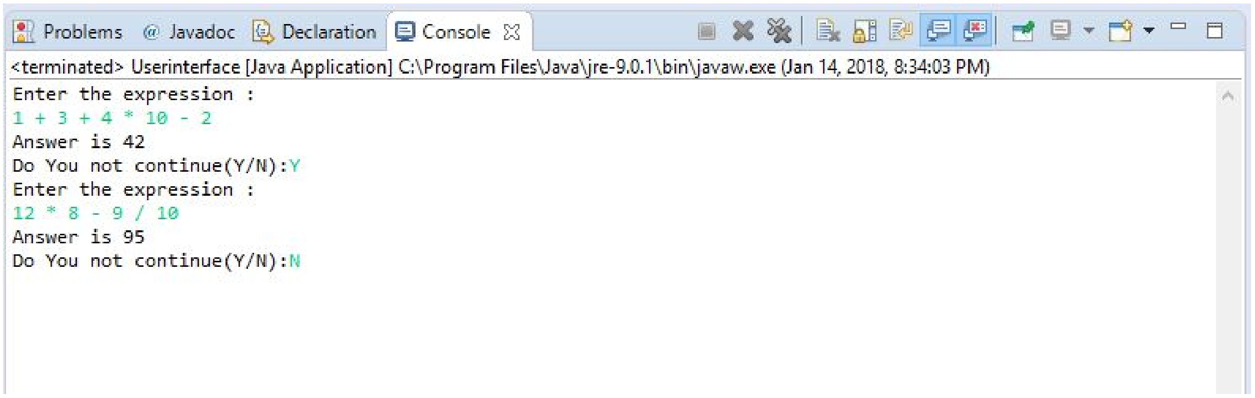
Implementation:

```
//userinterface
package user_interface;
import calculator_logic.BasicCalculator;
import java.util.Scanner;
public class CalculatorUI {
        public static void main(String[] args) {
            Scanner scan=new Scanner(System.in);
                String c="Y";
                while(c.equals("Y"))
                {
                System.out.println("Enter the expression : ");
                String exp;
                exp=scan.nextLine();
       BasicCalculator cal=new BasicCalculator();
                double res=cal.evaluate(exp);
                System.out.println("Answer is "+res);
                System.out.print("Do You not continue(Y/N):");
                c=scan.nextLine();
                }
                scan.close();
        }
}


//business logic
//interfaces
package calculator_logic;

public interface CalculatorInterface {
        public float add(float a,float b);
        public float sub(float a,float b);
        public float mul(float a,float b);
        public float div(float a,float b);
        public double evaluate(String expression);

}


//concrete implementation
package calculator_logic;
import java.util.Stack;
 public class BasicCalculator implements CalculatorInterface {
        @Override
        public float add(float a, float b) {
                // TODO Auto-generated method stub
                return a+b;
        }

        @Override
        public float sub(float a, float b) {
                // TODO Auto-generated method stub
                return a-b;
        }

        @Override
        public float mul(float a, float b) {
                // TODO Auto-generated method stub
                return a*b;
        }

        @Override
        public float div(float a, float b) {
                // TODO Auto-generated method stub
                return b/a;
        }
        public double evaluate(String expression)
        {
                Stack<Float> values;
                Stack<Character> ops;
                char[] tokens;
                tokens = expression.toCharArray();
                values = new Stack<>();
                ops = new Stack<Character>();
                for (int i = 0; i < tokens.length; i++)
                {
                        if (tokens[i] == ' ')
                                continue;
                        if (tokens[i] >= '0' && tokens[i] <= '9')
                        {
                                StringBuffer sbuf = new StringBuffer();
                                while (i < tokens.length && tokens[i] >= '0' &&
tokens[i] <= '9')
                                        sbuf.append(tokens[i++]);
                                values.push(Float.parseFloat(sbuf.toString()));
                        }
                        else if (tokens[i] == '(')
                                ops.push(tokens[i]);
                        else if (tokens[i] == ')')
                        {
                                while (ops.peek() != '(')
                                values.push(applyOp(ops.pop(), values.pop(),
values.pop()));
                                ops.pop();
                        }
                        else if (tokens[i] == '+' || tokens[i] == '-' ||
                                        tokens[i] == '*' || tokens[i] == '/')
                        {
```

```
                while (!ops.empty() && hasPrecedence(tokens[i],
ops.peek()))
                values.push(applyOp(ops.pop(), values.pop(),
values.pop()));
                ops.push(tokens[i]);
            }
        }
        while (!ops.empty())
            values.push(applyOp(ops.pop(), values.pop(),
values.pop()));
        return values.pop();
    }
    protected static boolean hasPrecedence(char op1, char op2)
    {
        if (op2 == '(' || op2 == ')')
            return false;
        if ((op1 == '*' || op1 == '/') && (op2 == '+' || op2 == '-'))
            return false;
        else
            return true;
    }
    protected float applyOp(char op, float b, float a)
    {
        switch (op)
        {
        case '+':
            return add(a,b);
        case '-':
            return sub(a,b);
        case '*':
            return mul(a,b);
        case '/':
            if (b != 0)
```

```
                        return div(a,b);
                }
                return 0;
        }

}
```

output:

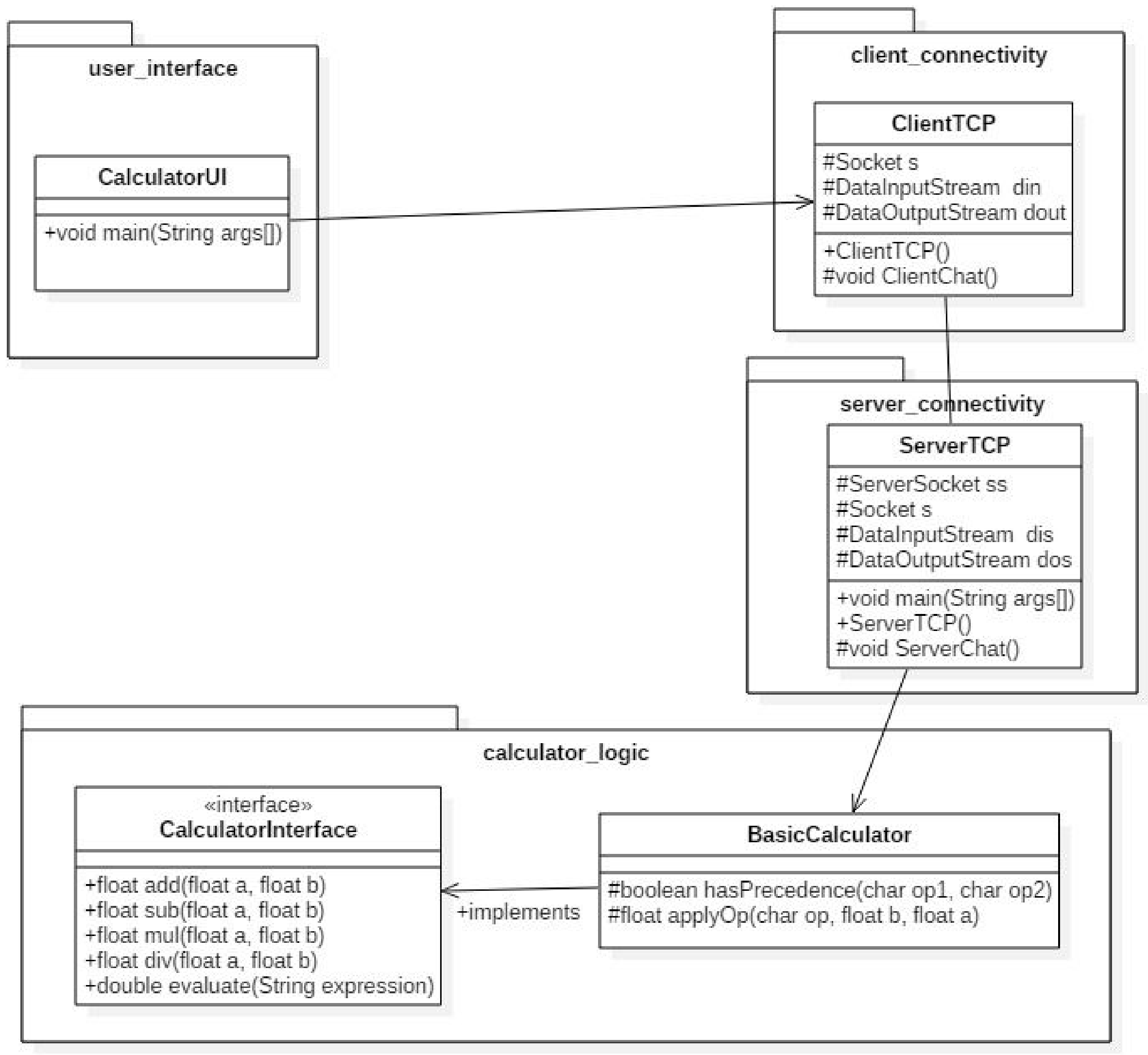


Problems @ Javadoc Declaration Console

```
<terminated> Userinterface [Java Application] C:\Program Files\Java\jre-9.0.1\bin\javaw.exe (Jan 14, 2018, 8:34:03 PM)
Enter the expression :
1 + 3 + 4 * 10 - 2
Answer is 42
Do You not continue(Y/N):Y
Enter the expression :
12 * 8 - 9 / 10
Answer is 95
Do You not continue(Y/N):N
```

Class Diagram:
<u>Client-server:</u>

fig.class Diagram: Calculator_clientserver_basic



<u>Implementation:</u>

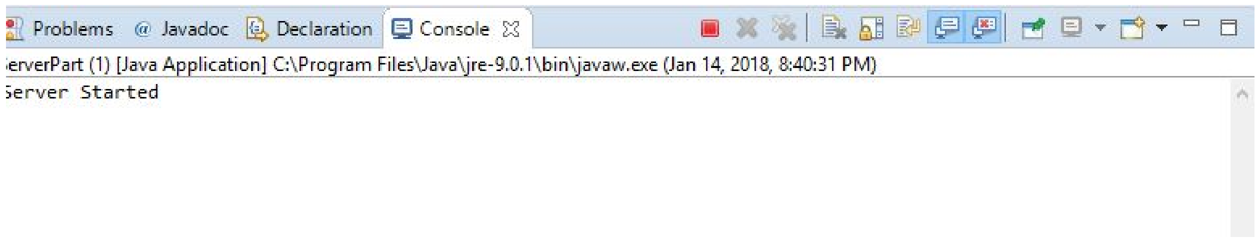
```
//user interface client
package ClientPackage;
import java.io.*;
import java.net.*;
import java.util.*;

public class ClientFile {

      Socket s;
   DataInputStream din;
   DataOutputStream dout;

   public static void main(String as[])
   {
      new ClientFile();
   }


   public ClientFile()
   {
       try
       {

          s=new Socket("localhost",10);
          //System.out.println(s);
          din= new DataInputStream(s.getInputStream());
          dout= new DataOutputStream(s.getOutputStream());
          ClientChat();
       }
       catch(Exception e)
       {
          System.out.println(e);
       }
    }
    public void ClientChat() throws IOException
    {
        String choice;
        do{

                Scanner scan=new Scanner(System.in);
                System.out.println("Enter the Expression : ");
                String exp=scan.nextLine();
                dout.writeUTF(exp);
                System.out.println(din.readUTF());
                String servermsg=din.readUTF();
                System.out.println(servermsg);
                choice=scan.next();
                dout.writeUTF(choice);
                dout.flush();
        }while(choice.equals("Y"));
   }
}

//server control file
package ServerPackage;
import java.io.*;
import basic_calculator.*;
import java.net.*;
import java.util.Stack;
import java.util.*;

public class ServerPart {
        ServerSocket ss;
        Socket s;
        DataInputStream dis;
        DataOutputStream dos;

        public static void main(String[] args) {
                new ServerPart();
```

```
	}
	public ServerPart()
	{
		try
		{
			System.out.println("Server Started");
			ss=new ServerSocket(10);
			s=ss.accept();
			System.out.println(s);
			System.out.println("Client Connected");
			dis= new DataInputStream(s.getInputStream());
			dos= new DataOutputStream(s.getOutputStream());
			ServerChat();
		}
		catch(Exception e)
		{
			System.out.println(e);
		}
	}

	public void ServerChat() throws IOException
	{
		do{
			int c=0;
			String resmsg;
			CalculatorLogic cal=new CalculatorLogic();
			String expression=dis.readUTF();
			int answer=cal.evaluate(expression);
			String res=Integer.toString(answer);
			dos.writeUTF(res);
			String conmsg="Do you want to continue:Y/N";
			dos.writeUTF(conmsg);
```

```
                        dos.flush();
                }while(dis.readUTF().equals("Y"));
        }
        }

//Business logic
  //interfaces
package calculator_logic;

public interface CalculatorInterface {
        public float add(float a,float b);
        public float sub(float a,float b);
        public float mul(float a,float b);
        public float div(float a,float b);
        public double evaluate(String expression);

}



//concrete implementation
package calculator_logic;
import java.util.Stack;
 public class BasicCalculator implements CalculatorInterface {
        @Override
        public float add(float a, float b) {
                // TODO Auto-generated method stub
                return a+b;
        }

        @Override
        public float sub(float a, float b) {
                // TODO Auto-generated method stub
                return a-b;
	}

	@Override
	public float mul(float a, float b) {
		// TODO Auto-generated method stub
		return a*b;
	}

	@Override
	public float div(float a, float b) {
		// TODO Auto-generated method stub
		return b/a;
	}

	public double evaluate(String expression)
	{
		Stack<Float> values;
		Stack<Character> ops;
		char[] tokens;
		tokens = expression.toCharArray();
		values = new Stack<>();
		ops = new Stack<Character>();
		for (int i = 0; i < tokens.length; i++)
		{
			if (tokens[i] == ' ')
				continue;
			if (tokens[i] >= '0' && tokens[i] <= '9')
			{
				StringBuffer sbuf = new StringBuffer();
				while (i < tokens.length && tokens[i] >= '0' &&
tokens[i] <= '9')
					sbuf.append(tokens[i++]);
				values.push(Float.parseFloat(sbuf.toString()));
				}
				else if (tokens[i] == '(')
					ops.push(tokens[i]);
				else if (tokens[i] == ')')
				{
					while (ops.peek() != '(')
					values.push(applyOp(ops.pop(), values.pop(),
values.pop()));
					ops.pop();
				}
				else if (tokens[i] == '+' || tokens[i] == '-' ||
						tokens[i] == '*' || tokens[i] == '/')
				{
					while (!ops.empty() && hasPrecedence(tokens[i],
ops.peek()))
					values.push(applyOp(ops.pop(), values.pop(),
values.pop()));
					ops.push(tokens[i]);
				}
			}
			while (!ops.empty())
				values.push(applyOp(ops.pop(), values.pop(),
values.pop()));
			return values.pop();
		}
		protected static boolean hasPrecedence(char op1, char op2)
		{
			if (op2 == '(' || op2 == ')')
				return false;
			if ((op1 == '*' || op1 == '/') && (op2 == '+' || op2 == '-'))
				return false;
			else
				return true;
```

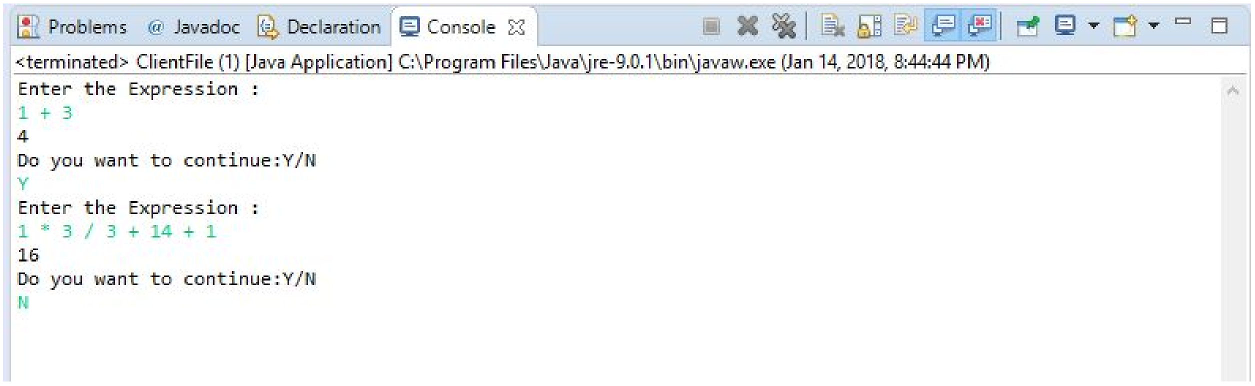
```
	}
	protected float applyOp(char op, float b, float a)
	{
		switch (op)
		{
		case '+':
			return add(a,b);
		case '-':
			return sub(a,b);
		case '*':
			return mul(a,b);
		case '/':
			if (b != 0)
				return div(a,b);
		}
		return 0;
	}

}
```

<u>Output:</u>



```
ServerPart (1) [Java Application] C:\Program Files\Java\jre-9.0.1\bin\javaw.exe (Jan 14, 2018, 8:40:31 PM)
Server Started
```

161080974_prashant 161080975_lalbahadur 161081981_freya



```
<terminated> ClientFile (1) [Java Application] C:\Program Files\Java\jre-9.0.1\bin\javaw.exe (Jan 14, 2018, 8:44:44 PM)
Enter the Expression :
1 + 3
4
Do you want to continue:Y/N
Y
Enter the Expression :
1 * 3 / 3 + 14 + 1
16
Do you want to continue:Y/N
N
```