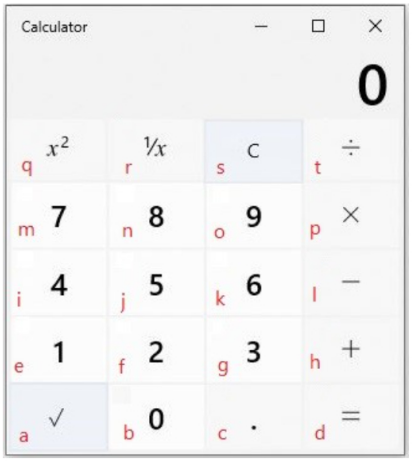
Java Week 12: Q1

Due on 2020-12-10, 23:59 IST

Complete the code to **develop an extended version of the ADVANCED CALCULATOR with added special functions that emulates all the functions of the GUI Calculator as shown in the image.**



Note the following points carefully:

1. Use only **double** datatype to store all numeric values.
2. Each button on the calculator should be operated by typing the characters from 'a' to 't'.
3. You may use the already defined function **gui_map(char)**.
4. Use predefined methods from **java.lang.Math** class wherever applicable.
5. Without '=' binary operations won't give output as shown in Input_3 and Output_3 example below.
5. The calculator should be able to perform required operations on one or two operands as shown in the below example:

Input_1:
okhid

Output_1:
100.0

Input_2:
ia

Output_2:
2.0

Input_3:
okhi

Output_3:

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	kq	36.0	36.0	Passed
Test Case 2	fr	0.5	0.5	Passed

The due date for submitting this assignment has passed.
2 out of 2 tests passed.
You scored 100.0/100.

Assignment submitted on 2020-12-09, 12:38 IST

Your last recorded submission was :

```
import java.util.Scanner;
public class Question92{
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        String input = sc.nextLine();

char seq[] = input.toCharArray();
int outflag=0;
for(int i=0; i<seq.length; i++)
        {
            seq[i]=gui_map(seq[i]);
            if (seq[i]=='R' || seq[i]=='S' || seq[i]=='F' ||seq[i]=='C')
                break;
        }

        //Print Mapped GUI (remove comment to see the mapped sequence input)
        /*
        for(int i=0; i<seq.length; i++){
            System.out.print(seq[i]);
        }
        */
```

```
        // Use double type of values for entire calculation
        double operand1=0.0;
        String o1="";
        double operand2=0.0;
        String o2="";
        double output=0.0;
        // Perform calculaton operations
        outerloop:
        for(int i=0; i<seq.length; i++){
            if(seq[i]=='C'){                //Clear
                operand1=0.0;
                operand2=0.0;
                output=0.0;
                outflag=0;
                break outerloop;
            }else if(seq[i]=='R'){          //Square Root
                for(int j=0; j<i; j++){
                    o1+=Character.toString(seq[j]);
                }
                operand1=Double.parseDouble(o1);
                output=Math.sqrt(operand1);
                outflag=1;
                break outerloop;
            }
            else if(seq[i]=='S'){           //Square
                for(int j=0; j<i; j++){
                    o1+=Character.toString(seq[j]);
                }
                operand1=Double.parseDouble(o1);
                output=Math.pow(operand1,2);
                outflag=1;
                break outerloop;
            }else if(seq[i]=='F'){          //Inverse
                for(int j=0; j<i; j++){
                    o1+=Character.toString(seq[j]);
                }
                operand1=Double.parseDouble(o1);
                output=Math.pow(operand1,-1);
                outflag=1;
                break outerloop;
            }else{
                int r=0;
                if(seq[i]=='+'||seq[i]=='-'||seq[i]=='/'||seq[i]=='*'||seq[i]=='='){
                    for(int j=0; j<i; j++){
                        o1+=Character.toString(seq[j]);
                    }
                    operand1=Double.parseDouble(o1);
                    for(int k=i+1; k<seq.length; k++){
                        if(seq[k]=='='){
                            outflag=1;
                            operand2=Double.parseDouble(o2);
                            if(seq[i]=='+'){
                                output=operand1+operand2;
                            }else if(seq[i]=='-'){
                                output=operand1-operand2;
                            }else if(seq[i]=='/'){
                                output=operand1/operand2;
                            }else if(seq[i]=='*'){
                                output=operand1*operand2;
                            }
                            break outerloop;
                        }else{
                            o2+=Character.toString(seq[k]);
                        }
                    }
                }
            }
        }
        // Check if output is available and print the output
        if(outflag==1)
            System.out.print(output);
    }// The main() method ends here.

// A method that takes a character as input and returns the corresponding GUI character
    static char gui_map(char in){
        char out = 'N';// N = Null/Empty
        char gm[][]={{'a','R'}
                    ,{'b','0'}
                    ,{'c','.'}
                    ,{'d','='}
                    ,{'e','1'}
                    ,{'f','2'}
                    ,{'g','3'}
                    ,{'h','+'}
                    ,{'i','4'}
                    ,{'j','5'}
                    ,{'k','6'}
                    ,{'l','-'}
                    ,{'m','7'}
                    ,{'n','8'}
                    ,{'o','9'}
                    ,{'p','*'}
                    ,{'q','S'}
                    ,{'r','F'}
                    ,{'s','C'}
                    ,{'t','/'}};
                     /* R = Square root
                    C = Clear/Restart
                    F = Fraction
                    S = Square
                    */

        // Checking for maps
        for(int i=0; i<gm.length; i++){
            if(gm[i][0]==in){
                out=gm[i][1];
                break;
            }
        }
        return out;
    }
}
```

Sample solutions (Provided by instructor)

```
import java.util.Scanner;
public class Question92{
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        String input = sc.nextLine();

         char seq[] = input.toCharArray();
        int outflag=0;

        for(int i=0; i<seq.length; i++){
            seq[i]=gui_map(seq[i]);
           if (seq[i]=='R' || seq[i]=='S' || seq[i]=='F' ||seq[i]=='C')
                break;
        }
        //Print Mapped GUI (remove comment to see the mapped sequence input)
        /*
```

```
        for(int i=0; i<seq.length; i++){
            System.out.print(seq[i]);
        }
        */
        // Use double type of values for entire calculation
        double operand1=0.0;
        String o1="";
        double operand2=0.0;
        String o2="";
        double output=0.0;
// Perform calculaton operations
        outerloop:
        for(int i=0; i<seq.length; i++){
            if(seq[i]=='C'){                //Clear
                operand1=0.0;
                operand2=0.0;
                output=0.0;
                outflag=0;
                break outerloop;
            }else if(seq[i]=='R'){          //Square Root
                for(int j=0; j<i; j++){
                    o1+=Character.toString(seq[j]);
                }
                operand1=Double.parseDouble(o1);
                output=Math.sqrt(operand1);
                outflag=1;
                break outerloop;
            }
            else if(seq[i]=='S'){           //Square
                for(int j=0; j<i; j++){
                    o1+=Character.toString(seq[j]);
                }
                operand1=Double.parseDouble(o1);
                output=Math.pow(operand1,2);
                outflag=1;
                break outerloop;
            }else if(seq[i]=='F'){          //Inverse
                for(int j=0; j<i; j++){
                    o1+=Character.toString(seq[j]);
                }
                operand1=Double.parseDouble(o1);
                output=Math.pow(operand1,-1);
                outflag=1;
                break outerloop;
            }else{
                int r=0;
                        if(seq[i]=='+'||seq[i]=='-'||seq[i]=='/'||seq[i]=='*'||seq[i]=='='){
                    for(int j=0; j<i; j++){
                        o1+=Character.toString(seq[j]);
                    }
                operand1=Double.parseDouble(o1);
                for(int k=i+1; k<seq.length; k++){
                 if(seq[k]=='='){
                      outflag=1;

                     operand2=Double.parseDouble(o2);
                 if(seq[i]=='+'){

                            output=operand1+operand2;
                    }else if(seq[i]=='-'){
                 output=operand1-operand2;
                    }else if(seq[i]=='/'){

                           output=operand1/operand2;
                    }else if(seq[i]=='*'){

                          output=operand1*operand2;
                     }
                    break outerloop;
                    }else{
                o2+=Character.toString(seq[k]);
                            }
                        }
                    }
                }
            }
            // Check if output is available and print the output
            if(outflag==1)
                System.out.print(output);

      }// The main() method ends here.

// A method that takes a character as input and returns the corresponding GUI character
        static char gui_map(char in){
            char out = 'N';// N = Null/Empty
            char gm[][]={{'a','R'}
                        ,{'b','0'}
                        ,{'c','.'}
                        ,{'d','='}
                        ,{'e','1'}
                        ,{'f','2'}
                        ,{'g','3'}
                        ,{'h','+'}
                        ,{'i','4'}
                        ,{'j','5'}
                        ,{'k','6'}
                        ,{'l','-'}
                        ,{'m','7'}
                        ,{'n','8'}
                        ,{'o','9'}
                        ,{'p','*'}
                        ,{'q','S'}
                        ,{'r','F'}
                        ,{'s','C'}
                        ,{'t','/'}};
                         /* R = Square root
                        C = Clear/Restart
                        F = Fraction
                        S = Square
                        */

            // Checking for maps
            for(int i=0; i<gm.length; i++){
                if(gm[i][0]==in){
                    out=gm[i][1];
                    break;
                }
            }
            return out;
        }
}
```