I dokument Calculator.java

import java.util.List;

import java.util.ArrayList;

public class Calculator {

static float finalResult;

static class Operations {

static final char ADDITION\_SYMBOL = '+';

static final char SUBTRACTION\_SYMBOL = '-';

static final char MULTIPLICATION\_SYMBOL = '\*';

static final char DIVISION\_SYMBOL = '/';

private Operations() {

}

public static String ToString() {

return "" + ADDITION\_SYMBOL + MULTIPLICATION\_SYMBOL + DIVISION\_SYMBOL + SUBTRACTION\_SYMBOL;

}

}

public static String Run(String expression) {

return evaluateExpression(expression);

}

private static String evaluateExpression(String expression) {

// get all entered numbers in string format

// if expression starts with + or -, just add one zero at the beginning

if (expression.charAt(0) == Operations.ADDITION\_SYMBOL

|| expression.charAt(0) == Operations.SUBTRACTION\_SYMBOL) {

expression = 0 + expression;

}

// split expression by arithmetic operation to get all digits

String[] numbers = expression.split("[" + Operations.ToString() + "]");

// parse all operations

List<String> operationList = new ArrayList<>();

for (int i = 0; i < expression.length() - 1; i++) {

if (expression.charAt(i) == Operations.ADDITION\_SYMBOL

|| expression.charAt(i) == Operations.SUBTRACTION\_SYMBOL

|| expression.charAt(i) == Operations.MULTIPLICATION\_SYMBOL

|| expression.charAt(i) == Operations.DIVISION\_SYMBOL) {

operationList.add(String.valueOf(expression.charAt(i)));

}

}

// parse all string numbers to float ones

List<Float> numberList = new ArrayList<>();

for (int i = 0; i < numbers.length; i++) {

if (numbers[i].equals("-Infinity")) {

numberList.add(Float.NEGATIVE\_INFINITY);

} else if (numbers[i].equals("Infinity")) {

numberList.add(Float.POSITIVE\_INFINITY);

} else {

try {

numberList.add(Float.parseFloat(numbers[i]));

} catch (Exception exc) {

return "ERROR";

}

}

}

Calculate(numberList, operationList);

String textResult = Float.toString(finalResult);

return textResult;

}

private static void Calculate(List<Float> numbers, List<String> operations) {

if (numbers.size() == 1) {

finalResult = numbers.get(0);

return;

}

float result = 0;

int indexMultiply = operations.indexOf(String.valueOf(Operations.MULTIPLICATION\_SYMBOL));

int indexDivide = operations.indexOf(String.valueOf(Operations.DIVISION\_SYMBOL));

if (indexMultiply != -1 && indexDivide != -1) {

if (indexMultiply < indexDivide) {

result += numbers.get(indexMultiply) \* numbers.get(indexMultiply + 1);

numbers.set(indexMultiply, result);

numbers.remove(indexMultiply + 1);

operations.remove(indexMultiply);

Calculate(numbers, operations);

return;

} else {

result += numbers.get(indexDivide) / numbers.get(indexDivide + 1);

numbers.set(indexDivide, result);

numbers.remove(indexDivide + 1);

operations.remove(indexDivide);

Calculate(numbers, operations);

return;

}

}

if (indexMultiply != -1) {

result += numbers.get(indexMultiply) \* numbers.get(indexMultiply + 1);

numbers.set(indexMultiply, result);

numbers.remove(indexMultiply + 1);

operations.remove(indexMultiply);

Calculate(numbers, operations);

return;

}

if (indexDivide != -1) {

result += numbers.get(indexDivide) / numbers.get(indexDivide + 1);

numbers.set(indexDivide, result);

numbers.remove(indexDivide + 1);

operations.remove(indexDivide);

Calculate(numbers, operations);

return;

}

int indexPlus = operations.indexOf(String.valueOf(Operations.ADDITION\_SYMBOL));

int indexMinus = operations.indexOf(String.valueOf(Operations.SUBTRACTION\_SYMBOL));

if (indexMinus != -1 && indexPlus != -1) {

if (indexPlus < indexMinus) {

result += numbers.get(indexPlus) + numbers.get(indexPlus + 1);

numbers.set(indexPlus, result);

numbers.remove(indexPlus + 1);

operations.remove(indexPlus);

Calculate(numbers, operations);

return;

} else {

result += numbers.get(indexMinus) - numbers.get(indexMinus + 1);

numbers.set(indexMinus, result);

numbers.remove(indexMinus + 1);

operations.remove(indexMinus);

Calculate(numbers, operations);

return;

}

}

if (indexPlus != -1) {

result += numbers.get(indexPlus) + numbers.get(indexPlus + 1);

numbers.set(indexPlus, result);

numbers.remove(indexPlus + 1);

operations.remove(indexPlus);

Calculate(numbers, operations);

return;

}

if (indexMinus != -1) {

result += numbers.get(indexMinus) - numbers.get(indexMinus + 1);

numbers.set(indexMinus, result);

numbers.remove(indexMinus + 1);

operations.remove(indexMinus);

Calculate(numbers, operations);

return;

}

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

II dokument Start.java

import java.util.Scanner;

public class Start {

public static void main(String[] args) {

String Expression;

boolean active = true;

System.out.println("Enter expression here (type 'exit' to quit):");

Scanner scanIn;

while (active) {

scanIn = new Scanner(System.in);

Expression = scanIn.nextLine();

if (Expression.equals("exit")) {

scanIn.close();

active = false;

} else {

System.out.println(Calculator.Run(Expression));

}

}

}

}