南开大学

JAVA语言与应用

图形化计算器实验报告

姓 名：冯朝芃

学 号：2012039

年 级： 2020 级

学 院： 计算机学院

专 业 ：计算机科学与技术

授课教师：刘嘉欣

完成日期：2021年 11月 21日

一、概述：

本作业为图形化计算器。本作业实现的功能有：加减乘除、括号运算、小数运算、乘方运算、负数运算、百分数、退格、清除上一次输入数字等

二、运行展示：

运行效果截图：

电脑屏幕的照片

低可信度描述已自动生成

附录：完整代码

package Interface;

import cacu.Caculor;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class Interface extends JFrame{

public Interface(String title) {

super(title);

GridLayout textLayout = new GridLayout(2,1);

JPanel textPanel = new JPanel(textLayout);

JTextField preTextField = new JTextField("0");

preTextField.setEditable(false);

preTextField.setHorizontalAlignment(JTextField.RIGHT);

preTextField.setFont(new Font("宋体", Font.BOLD, 14));

preTextField.setBounds(0, 0, 500, 20);

JTextField numlineField=new JTextField("0");

numlineField.setBounds(22,22,500,50);

numlineField.setHorizontalAlignment(JTextField.RIGHT);

numlineField.setFont(new Font("Times New Roman",Font.BOLD,40));

textPanel.add(preTextField);

textPanel.add(numlineField);

GridLayout butGridLayout=new GridLayout(6,4);

JPanel butPanel=new JPanel(butGridLayout);

butPanel.setPreferredSize(new Dimension(600,600));

{

butPanel.add(new JButton("+/-"));

butPanel.add(new JButton("C"));

butPanel.add(new JButton("CE"));

butPanel.add(new JButton("backspace"));

butPanel.add(new JButton("("));

butPanel.add(new JButton(")"));

butPanel.add(new JButton("^"));

butPanel.add(new JButton("/"));

butPanel.add(new JButton("7"));

butPanel.add(new JButton("8"));

butPanel.add(new JButton("9"));

butPanel.add(new JButton("\*"));

butPanel.add(new JButton("4"));

butPanel.add(new JButton("5"));

butPanel.add(new JButton("6"));

butPanel.add(new JButton("-"));

butPanel.add(new JButton("1"));

butPanel.add(new JButton("2"));

butPanel.add(new JButton("3"));

butPanel.add(new JButton("+"));

butPanel.add(new JButton("%"));

butPanel.add(new JButton("0"));

butPanel.add(new JButton("."));

butPanel.add(new JButton("="));

}

class buttonListener implements ActionListener{

@Override

public void actionPerformed(ActionEvent e){

String what=((JButton) e.getSource()).getText();

preTextField.setText(numlineField.getText()+"=");

switch(what){

case "CE":

int tmp=numlineField.getText().length()-1;

while(tmp>0){

if(numlineField.getText().charAt(tmp)<'0'||numlineField.getText().charAt(tmp)>'9'){

break;

}

tmp--;

}

numlineField.setText(numlineField.getText().substring(0,tmp+1));

break;

case "C":

numlineField.setText("0");

break;

case"+/-":

numlineField.setText(numlineField.getText().equals("0") ? "0" : "-"+numlineField.getText());

break;

case "=":

numlineField.setText(new Caculor().caculate(numlineField.getText()));

break;

case "backspace":

if(numlineField.getText().length()==1) {

numlineField.setText("0");

break;

}

numlineField.setText(numlineField.getText().equals("0") ? "0" : numlineField.getText().substring(0,numlineField.getText().length()-1));

break;

default:

numlineField.setText(numlineField.getText().equals("0") ? what : numlineField.getText() + what);

}

}

}

this.add(BorderLayout.NORTH,textPanel);

this.add(BorderLayout.SOUTH,butPanel);

for (int i=0;i<24;i++) {

((JButton) butPanel.getComponent(i)).addActionListener(new buttonListener());

}

}

public static void main(String[] args) {

Interface startInterface=new Interface("Calculator");

startInterface.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

startInterface.setSize(600, 800);

startInterface.setUndecorated(true); // 去掉窗口的装饰

startInterface.getRootPane().setWindowDecorationStyle(JRootPane.PLAIN\_DIALOG);//采用指定的窗口装饰风格

startInterface.setVisible(true);

}

}

package cacu;

import java.util.Stack;

public class Caculor {

//stack of numbers

private Stack<Double> numbers = new Stack<Double>();

//stack of operands

private Stack<Character> operands = new Stack<Character>();

public String caculate(String exp) {

String tmp=cacuExpressions(exp).toString();

numbers.clear();

operands.clear();

return tmp;

}

//get the priority of the operator

private int getPriority(char op) {

switch (op) {

case '+':

case '-':

return 1;

case '\*':

case '/':

return 2;

case '^':

return 3;

default:

return -1;

}

}

//calculate the number of operators in a string expression

private int getOperatorNum(String exp) {

int num = 0;

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

if (exp.charAt(i) == '+' || exp.charAt(i) == '-' || exp.charAt(i) == '\*' || exp.charAt(i) == '/' || exp.charAt(i) == '^') {

num++;

}

}

return num;

}

//calculate the top two numbers and the top operator, no input validation

private void cacuTwoNumbers() {

double num1 = numbers.pop();

double num2 = numbers.pop();

switch (operands.pop()) {

case '+':

numbers.push(num2 + num1);

break;

case '-':

numbers.push(num2 - num1);

break;

case '\*':

numbers.push(num2 \* num1);

break;

case '/':

numbers.push(num2 / num1);

break;

case '^':

numbers.push(Math.pow(num2, num1));

break;

}

}

private void numfixer(String exp,int i,int j){

Double tmp;

if(i==0&&exp.charAt(i-1) == '-'){

numbers.push(0.0);

}

tmp = exp.charAt(j-1)=='%'?Double.parseDouble(exp.substring(i, j-1))/100:Double.parseDouble(exp.substring(i, j));

numbers.push(tmp);

}

//caculate the result of the infix expression

private Double cacuExpressions(String exp) {

int num = getOperatorNum(exp);

if (num == 0) {

return Double.parseDouble(exp);

}

if(num == 1) {

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

if (exp.charAt(i) >= '0' && exp.charAt(i) <= '9') {

int j = i;

while (j < exp.length() && ((exp.charAt(j) >= '0' && exp.charAt(j) <= '9')

||exp.charAt(j) =='.'||exp.charAt(j) == '%')) {

j++;

}

numfixer(exp,i,j);

i = j - 1;

}else{

operands.push(exp.charAt(i));

}

}

cacuTwoNumbers();

return numbers.pop();

}

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

if (exp.charAt(i) >= '0' && exp.charAt(i) <= '9') {

int j = i;

while (j < exp.length() && ((exp.charAt(j) >= '0' && exp.charAt(j) <= '9')

||exp.charAt(j) =='.'||exp.charAt(j) == '%')) {

j++;

}

numfixer(exp,i,j);

i = j - 1;

} else {

if (exp.charAt(i) == '(') {

operands.push(exp.charAt(i));

} else if (exp.charAt(i) == ')') {

while (operands.peek() != '(') {

cacuTwoNumbers();

}

operands.pop();

} else {

while (!operands.empty() && (getPriority(exp.charAt(i)) <= getPriority(operands.peek()))) {

cacuTwoNumbers();

}

operands.push(exp.charAt(i));

}

}

}

while (!operands.empty()) {

cacuTwoNumbers();

}

return numbers.pop();

}

}