Universitatea “Spiru Haret”

Facultatea de Inginerie si Informatica

Examen TAP

Student: Teodorescu Vlad

Grupa: 202

**1.Cerinta**

**III.** Se consideră funcţia # ataşată unei instrucţiuni din limbajul S, aşa cum este definită la curs.

Să se scrie un program care citeşte de la intrare o instrucţiune I din limbajul S şi produce la ieşire valoarea #(I).

**1.1. Explicarea programului**

**1.1.1. Importarea claselor**

import javax.swing.\*; //Utilizat in iterfata grafica  
import java.awt.\*;//Utilizat in iterfata grafica  
import java.awt.event.ActionEvent;//Folosit pentru a seta actiunea unui event  
import java.awt.event.ActionListener;//Folosit pentru a captura un event  
import java.awt.event.KeyAdapter;  
import java.awt.event.KeyEvent;//Folosit pentru a seta actiunea unui event de tip keypress  
import java.util.Objects;

**1.1.2. Interfata grafica**

JPanel panel = new JPanel();  
JLabel rezLb = new JLabel("Rezultatul este:");  
JButton send = new JButton("Calculeaza");  
JLabel errLb = new JLabel("");  
  
panel.add(send);  
  
panel.add(rezLb);  
panel.add(errLb);

Implementarea butonului ce declansaza calculul, si a catorva label-uri ce vor fi folosite pentru a returna o eroare sau rezultatul.

GridLayout horizontal = new GridLayout(0,1);  
  
JPanel mainpan=new JPanel(horizontal);  
  
JPanel panelmain0 = new JPanel();

Utilizam GridLayout pentru pentru a alinia elementele grafice.

Initializam mainpan, un panou ce va tine toate celelalte elemente de selectie a input-ului.

JLabel lbl0 = new JLabel("Ce valoare are eticheta L?");  
lbl0.setAlignmentX(Component.*CENTER\_ALIGNMENT*);  
JTextField valA=new JTextField("",5);  
  
  
  
valA.addKeyListener(new KeyAdapter() {  
 public void keyPressed(KeyEvent ke) {  
 valA.setEditable(true);  
  
  
 if (ke.getKeyChar() >= '0' && ke.getKeyChar() <= '9'|| ke.getKeyCode() == KeyEvent.*VK\_BACK\_SPACE*) {  
 valA.setEditable(true);  
 errLb.setText("");  
 } else {  
 valA.setEditable(false);  
 errLb.setText("\* Enter only numeric digits(0-9)");  
 errLb.setForeground(Color.*RED*);  
 }  
  
 }  
});  
  
panelmain0.add(lbl0);  
panelmain0.add(valA);

Petru a grupa label si textfield declaram un nou pannel panelmain0.

Adaugam un keyListener pentru a impiedica introducerea altor caracter in afara de numere si backspace.

In cazul in care avem alte caractere introduse dezactivam campul si afisam o eroare.

La final adaugam elementele(label si textfield) in panemain0

JPanel panelmain1 = new JPanel();  
JPanel panelSec= new JPanel();  
JTextField valS=new JTextField(5);  
  
JLabel lbl1 = new JLabel("Ce tip de variabila avem?");  
lbl1.setAlignmentX(Component.*CENTER\_ALIGNMENT*);  
  
panelmain1.add(lbl1);  
  
String[] choices = { "Variabila de iesire","Variabila de intrare","Variabila de intermediara" };  
  
final JComboBox<String> cb = new JComboBox<String>(choices);  
  
cb.setMaximumSize(cb.getPreferredSize());  
cb.setAlignmentX(Component.*CENTER\_ALIGNMENT*);  
  
panelmain1.add(cb);

Declaram grupul 2 ca panelmain1. aplicam acelasi principiu ca si la panelmain0.

Pentru a introduce date folosim un ComboBox cu lista de valori dorita.

Setam latimea campului ca lungimea maxima a celui mai lung String.

La final adaugam elementele in panelmain1.

cb.addItemListener(arg0 -> {  
 if (Objects.*requireNonNull*(cb.getSelectedItem()).toString().equals("Variabila de iesire")){  
 panelSec.setVisible(false);  
 valS.setText("");  
  
 }else {  
 panelSec.setVisible(true);  
 }  
});  
  
  
JLabel lbls = new JLabel("Introduceti indicele variabilei!");  
lbls.setAlignmentX(Component.*CENTER\_ALIGNMENT*);  
  
  
valS.addKeyListener(new KeyAdapter() {  
 public void keyPressed(KeyEvent ke) {  
 valS.setEditable(true);  
 if (ke.getKeyChar() >= '0' && ke.getKeyChar() <= '9'|| ke.getKeyCode() == KeyEvent.*VK\_BACK\_SPACE*) {  
 valS.setEditable(true);  
 errLb.setText("");  
 } else {  
 valS.setEditable(false);  
 errLb.setText("\* Enter only numeric digits(0-9)");  
 errLb.setForeground(Color.*RED*);  
 }  
  
 }  
});  
panelSec.add(lbls);  
panelSec.add(valS);  
panelSec.setVisible(false);

Folosim un listener pentru a vedea cand se schimba valoarea si daca valoarea selectata necesita si indice afisam urmatorul set de campuri.

Folosim aceiasi regula ca la panelmain0 pentru a introduce numai numere.

la final adaugam fiecare element in panelul lui.

Setam sa nu fie vizibil campul de index ca sa fie aliniat cu valoarea default a comboBox-ului.

JPanel panelmain2 = new JPanel();  
  
JLabel lbl2 = new JLabel("Ce instructiune folosim?");  
lbl2.setAlignmentX(Component.*CENTER\_ALIGNMENT*);  
  
panelmain2.add(lbl2);  
  
String[] choices2 = { "v←v","v←v+1","v←v-1","IF v≠0 goto L" };  
  
final JComboBox<String> cb2 = new JComboBox<String>(choices2);  
  
cb2.setMaximumSize(cb2.getPreferredSize());  
cb2.setAlignmentX(Component.*CENTER\_ALIGNMENT*);  
  
panelmain2.add(cb2);  
  
  
//  
  
  
JPanel panelmain3 = new JPanel();  
JLabel lbl3 = new JLabel("Introduceti L!");  
lbl3.setAlignmentX(Component.*CENTER\_ALIGNMENT*);  
JTextField valI=new JTextField(5);  
  
valI.addKeyListener(new KeyAdapter() {  
 public void keyPressed(KeyEvent ke) {  
 valI.setEditable(true);  
 if (ke.getKeyChar() >= '0' && ke.getKeyChar() <= '9'|| ke.getKeyCode() == KeyEvent.*VK\_BACK\_SPACE*) {  
 valI.setEditable(true);  
 errLb.setText("");  
 } else {  
 valI.setEditable(false);  
 errLb.setText("\* Enter only numeric digits(0-9)");  
 errLb.setForeground(Color.*RED*);  
 }  
  
 }  
});  
panelmain3.add(lbl3);  
panelmain3.add(valI);  
panelmain3.setVisible(false);  
  
cb2.addItemListener(arg0 -> {  
 if (Objects.*requireNonNull*(cb2.getSelectedItem()).toString().equals("IF v≠0 goto L")){  
 panelmain3.setVisible(true);  
  
 }else {  
 panelmain3.setVisible(false);  
 valI.setText("");  
  
 }  
});

Folosim aceiasi logica ca si pentru panelmain1 si pentru panelmain3 impreuna cu texfield-ul asociat valorii lui L.

send.addActionListener(new ActionListener() {  
  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 int a=0, b = 0, c = 0, i, l;  
  
 if (!valA.getText().isEmpty()) {  
 a=Integer.*parseInt*(valA.getText());  
 }  
  
 switch (cb.getSelectedItem().toString()) {  
 case "Variabila de iesire" -> {  
 c = 0;  
 errLb.setText("");  
  
 }  
 case "Variabila de intrare" -> {  
 if (valS.getText().isEmpty()) {  
 errLb.setText("\* Completati Indicele!");  
 errLb.setForeground(Color.*RED*);  
 rezLb.setText("Rezultatul este:");  
 } else {  
 i = Integer.*parseInt*(valS.getText());  
 c = (2 \* i) - 1;  
 errLb.setText("");  
 }  
 }  
 case "Variabila de intermediara" -> {  
 if (valS.getText().isEmpty()) {  
 errLb.setText("\* Completati Indicele!");  
 errLb.setForeground(Color.*RED*);  
 rezLb.setText("Rezultatul este:");  
 } else {  
 i = Integer.*parseInt*(valS.getText());  
 c = 2 \* i;  
 errLb.setText("");  
 }  
 }  
 }  
  
 switch (cb2.getSelectedItem().toString()) {  
 case "IF v≠0 goto L" -> {  
 if (valI.getText().isEmpty() || Integer.*parseInt*(valI.getText()) == 0) {  
 errLb.setText("\* L trebuie sa fie un numar mai mare ca 0");  
 errLb.setForeground(Color.*RED*);  
 rezLb.setText("Rezultatul este:");  
 } else if (errLb.getText().equals("")) {  
 l = Integer.*parseInt*(valI.getText());  
 b = 2 + l;  
 errLb.setText("");  
 } else {  
 rezLb.setText("Rezultatul este:");  
  
 }  
 }  
 case "v←v-1" -> {  
 b = 2;  
 }  
 case "v←v+1" -> {  
 b = 1;  
 }  
 case "v←v" -> {  
 b = 0;  
 }  
 }

Setam o actiune pentru butonul de calcul.

Colectam datele intorduse de utilizator pana in acel moment, validam combinatiile de input utilizand switch.

Returnam eroare in cazul in care nu se poate calcula un raspuns valid.

if (errLb.getText().equals("")){  
 rezLb.setText("Rezultatul este:"+*fBiject*(a,*fBiject*(b,c)));  
}

Daca nu avem nici o eroare calculam si afisam rezultatul.

//Adding Components to the frame.  
frame.getContentPane().add(BorderLayout.*SOUTH*, panel);  
//frame.getContentPane().add(BorderLayout.NORTH,panelmain1 );  
mainpan.add(panelmain0);  
mainpan.add(panelmain1);  
mainpan.add(panelSec);  
mainpan.add(panelmain2);  
mainpan.add(panelmain3);  
frame.getContentPane().add(BorderLayout.*CENTER*, mainpan);  
  
frame.setVisible(true);

La final adaugam toate grupurile in panel-ul mainpan si si setam frame.setVisible(true) ca sa apara elementele pe ecran.

Mai jos avem cateva cazuri:







