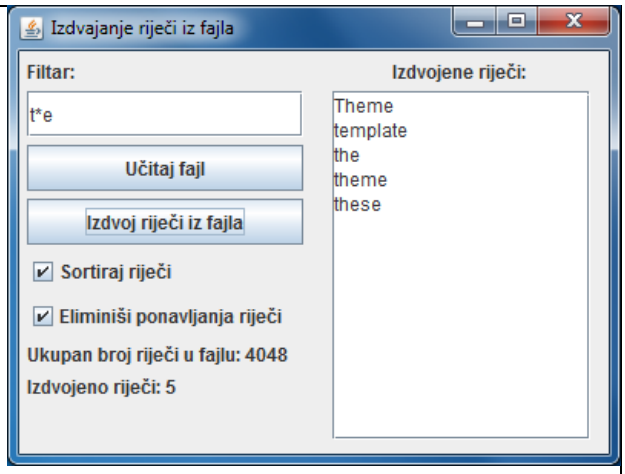
OBJEKTNO-ORIJENTISANI DIZAJN SOFTVERA

-VI čas računskih vježbi-

1. Napisati JAVA program koji će omogućiti pretraživanje riječi u tekstualnom fajlu. Aplikacija treba da, na osnovu specificiranog filtra, prikaže sve riječi iz odabranog fajla koje zadovoljavaju odgovarajući kriterijum. U filtru se, pored standardnih karaktera, mogu naći i džoker znaci '*' i '?'. Znak '*' mijenja proizvoljan broj karaktera u riječi, dok '?' mijenja tačno jedan karakter.




```
import java.awt.Dimension;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.File;
import java.util.Collections;
import java.util.HashSet;
import java.util.LinkedList;
import java.util.Scanner;
import java.util.Set;
import java.util.TreeSet;
import javax.swing.*;
import javax.swing.filechooser.FileNameExtensionFilter;
import java.util.Iterator;
public class FilterRijeci extends JFrame implements ActionListener
{
  private JTextArea tekstProstor;
  private JTextField textPolje;
  private JButton b1,b2;
  private JLabel labela1, labela2,labela3, labela4;
  private JPanel panel1, panel2;
  private LinkedList<String> sveRijeci;
  private LinkedList<String> izdvojeneRijeci;
  private JCheckBox chB1, chB2;
  public FilterRijeci()
  {
    super("Izdvajanje riječi iz fajla");
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setSize(400,300);
    this.setLayout(new GridLayout(1,2));
    // Kreiranje dva panela za organizaciju komponenti na formi
    panel1=new JPanel();
    panel1.setLayout(new FlowLayout(FlowLayout.LEFT));
    this.add(panel1);
    panel2=new JPanel();
    this.add(panel2);
    // Kreiranje odgovarajućih grafičkih komponenti
    labela1=new JLabel("Filtar: ");
    labela2=new JLabel("Izdvojene riječi: ");
    // Kreiranje JTextArea
    tekstProstor=new JTextArea(14,15);
    // Cilj je da korisnik ne može mijenjati zapise u JTextArea
    tekstProstor.setEditable(false);
    panel2.add(labela2);
    panel2.add(tekstProstor);
    panel2.add(new JScrollPane(tekstProstor));
    // Kreiranje text polja za unos filtra
    textPolje=new JTextField(16);
    textPolje.setPreferredSize(new Dimension(180,30));
    b1=new JButton("Učitaj fajl");
    b1.setPreferredSize(new Dimension(180,30));
    b2=new JButton("Izdvoj riječi iz fajla");
    b2.setPreferredSize(new Dimension(180,30));
    panel1.add(labela1);
    panel1.add(textPolje);
    panel1.add(b1);
    panel1.add(b2);
    // Dodavanje osluškivača na akciona dugmad
```

```
  // primjetimo da sama klasa FilterRijeci implementira interfejs ActionListener
  // pa ista predstavlja osluškivač
  b1.addActionListener(this);
  b2.addActionListener(this);
  // Na početku, lista sa svim riječima je prazna
  sveRijeci= new LinkedList <String> ();
  // U listi izdvojeneRijeci će se nalaziti sve riječi koje zadovoljavaju kriterijum propisan
  //filtrom
  izdvojeneRijeci= new LinkedList <String> ();
  // Check dugmad kojima se vrši izbor prikaza u JTextArea
  chB1=new JCheckBox("Sortiraj riječi");
  chB2=new JCheckBox("Eliminiši ponavljanja riječi");
  panel1.add(chB1);
  panel1.add(chB2);
  // Labele koje prikazuju ukupan broj riječi u fajlu i izdvojeni broj riječi
  labela3=new JLabel();
  labela4=new JLabel();
  panel1.add(labela3);
  panel1.add(labela4);
}
private void ucitajRijeci()
{
  // Ova funkcija omogućava izbor fajla iz koga se vrši čitanje
  // kao i učitavanje riječi iz fajla
  JFileChooser izborFajla = new JFileChooser();
  // Na ovaj način definišemo filtar u JFileChooser-u
  FileNameExtensionFilter filter = new FileNameExtensionFilter("Tekstualni fajlovi (*.txt)",
  "txt");
  izborFajla.addChoosableFileFilter(filter);
  int ret = izborFajla.showDialog(null, "Izaberi fajl");
  if (ret == JFileChooser.APPROVE_OPTION)
  {
    try
    {
      // Selektovani fajl
      File fajl=izborFajla.getSelectedFile();
      Scanner sc=new Scanner(fajl);
      sc.useDelimiter("[ ,.;!?]");
      sveRijeci.clear();
      while (sc.hasNext())
      {
        String p=sc.next();
        if (p.length()!=0) sveRijeci.add(p);
      }
    }
    catch (Exception e)
    {
      System.out.println("Greška " + e);
    }
  }
}
private void izdvojRijeci(String filtar)
{
  // ovom funkcijom vršimo izdvajanje riječi prema odgovarajućem filtru
  izdvojeneRijeci.clear();
  // Iterator omogućava prolaz kroz sve elemente liste
  Iterator <String> iter=sveRijeci.iterator();
  if (filtar.equals("")) filtar="*";
  while (iter.hasNext())
  {
    String pom=iter.next();
    // Ako riječ zadovoljava kriterijum propisan filtrom, dodaj je u listu izdvojeneRijeci
    if (zadovoljavaKriterijum(pom, filtar))
      izdvojeneRijeci.add(pom);
  }
}
public boolean zadovoljavaKriterijum(String rijec, String stringFiltra)
{
```

```
  // Ova funkcija provjerava da li data riječ zadovoljava kriterijum propisan filtrom
  // karakteri '*' i '?' predstavljaju džoker znake
  String pom=stringFiltra.replace("*", "(.*)");
  pom=pom.replace("?", "(.?)");
  rijec=rijec.toLowerCase();
  pom=pom.toLowerCase();
  return rijec.matches(pom);
}
private void upisiRijeciUTekstProstor()
{
  // Ova funkcija vrši upis podataka u JTextArea
  Iterator <String> iter=null;

  if (chB2.isSelected())
  {
    if (chB1.isSelected())
    {
      // Ako je korisnik izabrao sortiranje i eliminaciju ponavljanja u JTextArea
      Set< String > setRijeci = new TreeSet< String >(izdvojeneRijeci);
      iter=setRijeci.iterator();
    }
    else
    {
      // Ako je korisnik izabrao samo eliminaciju ponavljanja u JTextArea
      Set< String > setRijeci = new HashSet< String >(izdvojeneRijeci);
      iter=setRijeci.iterator();
    }
  }
  else
  {
    // Ako se korisnik odlučio da ne izvrši eliminaciju ponavljanja riječi
    LinkedList <String> setRijeci=izdvojeneRijeci;
    if (chB1.isSelected())
    {
      // Ako je odabrao sortiranje
      Collections.sort(setRijeci);
    }
    iter=setRijeci.iterator();
  }
  // Upis podataka u JTextArea
  int br=0;
  String Pom="";
  while (iter.hasNext())
  {
    if (br==0)
      Pom=iter.next();
    else
      Pom+= "\n" + iter.next();
    br++;
  }
  tekstProstor.setText(Pom);
  // Ažuriranje labela o broju riječi
  labela3.setText("Ukupan broj riječi u fajlu: "+sveRijeci.size());
  labela4.setText("Izdvojeno riječi: " + br);
}
public static void main(String [] args)
{
  FilterRijeci ff=new FilterRijeci();
  ff.setVisible(true);
}
@Override
public void actionPerformed(ActionEvent e)
{
  if (e.getSource()==b1)
  {
    ucitajRijeci();
    izdvojRijeci("");
  }
```

```
    if (e.getSource()==b2)
    {
      izdvojRijeci(textPolje.getText().trim());
    }
    upisiRijeciUTekstProstor();
  }
}
```