ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ
*Одсек за РТИ и Одсек за СИ* Београд 15.06.2015.

## Испит из Објектно оријентисаног програмирања II

**1)** *(30 поена)* Одговорити концизно (по једна или две реченице) и прецизно на следећа питања:

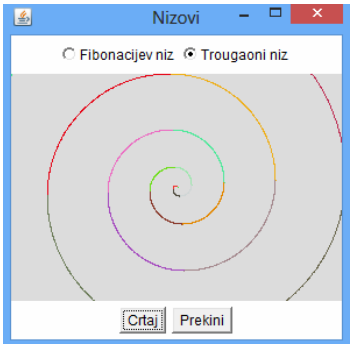
**а)** (*Java*) Објаснити однос класа `Component` и `Container` из пакета `java.awt`.

**б)** (*C#*) Какво је подразумевано право приступа члановима (1) класа (2) структура?

**в)** (*C#*) Шта су делегати и догађаји – категорије корисничких типова или чланови типова? Какав је однос између делегата и догађаја?

**2)** *(укупно 70 поена)* Написати на језику *Java* следећи скуп типова логично организованих по пакетима (грешке пријављивати изузецима опремљеним текстовима порука):

- *(20 поена)* ***Произвођач*** целобројне вредности може да врати целобројну вредност.
- ***Генератор*** низа целих бројева је произвођач који садржи низ целобројних клица који се задаје при стварању.
- ***Фибоначијев генератор*** је генератор низа целих бројева, који генерише бројеве Фибоначијевог низа (почев од вредности 1). Почетне клице су {0, 1}. Нова вредност Фибоначијевог низа се добија сабирањем клица (*klice[0] + klice[1]*). Приликом сваког генерисања потребно је ажурирати низ клица тако да вредности у новом низу клица буду буду {*klice[1]*, *nova_generisana_vrednost*}.
- ***Троугаони генератор*** је генератор низа целих бројева, који генерише бројеве Троугаоног низа. Почетне клице су {0, 1, 1}. Нова вредност Троугаоног низа се добија сабирањем *klice[0] + klice[1]*. Приликом сваког генерисања потребно је ажурирати низ клица тако да вредности у новом низу клица буду {*nova_generisana_vrednost*, *klice[1] + klice[2]*, *klice[2]*}.
- *(25 поена)* ***Тачка*** у равни садржи реалне координате *x* и *y* које се задају при стварању (подразумевано (0,0)) и које могу да се дохвате.
- Кружни ***лук*** садржи тачку која представља центар лука, дужину полупречника, почетни угао, крајњи угао и графичку компоненту за цртање. Угао се мери у односу на смер позитивне X осе, насупрот кретања казаљке часовника. За задати угао, може да се дохвати тачка на луку чије се координате рачунају по формулама `x = xC + r*Math.cos(u)` и `y = yC - r*Math.sin(u)` (`r` – полупречник лука, `u` - угао, `xC` и `yC` – координате центра лука). Грешка је ако је угао изван опсега од почетног до крајњег угла лука. Лук се црта надовезивањем (кратких) правих линија на придружену компоненти, са кораком `2*Math.PI/1000` и паузом од 3ms после сваког корака. Боја лука треба да буде случајна комбинација RGB компоненти.
- *(25 поена)* ***Графички приказивач***, на обухваћеном активном ***платну***, црта фигуру са слике на основу генерисаног низа целобројних вредности, према приложеној слици. За сваки број низа се црта кружни лук полупречника једнаког генерисаном броју, ширине угла (разлике почетног и крајњег угла) `Math.PI/2`. Центар лука који представља први број низа налази се на средини површине за цртање, а његов почетни угао је 0. Центре и углове лукова који представљају остале бројеве у низу прилагодити, тако да фигура изгледа као на приложеној слици. Цртање може да се покрене и прекине.




---

**НАПОМЕНЕ:** **а)** Испит траје **180** минута.

**б)** Рад се предаје искључиво у вежбанци за испите (-5 поена за неадекватну вежбанку). Није дозвољено имати поред себе друге листове папира, нити уз себе имати мобилни телефон, без обзира да ли је укључен или искључен.

**в)** Дозвољена је употреба ***Подсетника за AWT***.

**г)** Водити рачуна о уредности. Нечитки делови текста ће бити третирани као непостојећи. Решења задатака навести по горњем редоследу (-1 поен за лош редослед). Препоручује се рад обичном графитном оловком.

**д)** Резултати испита биће објављени на *Web*-у на адреси: `http://rti.etf.bg.ac.rs/rti/ir2oo2/index.html/`.

```
// Proizvodjac.java
package proizvodjac;

public interface Proizvodjac {
  int generisi();
}
```

```
// Generator.java
package proizvodjac;

public abstract class Generator implements
                                Proizvodjac {
  protected int[] klice;

  public Generator(int[] k){
    klice = new int[k.length];
    for (int i=0; i<k.length; i++)
      klice[i] = k[i];
  }
}
```

```
// FibonacijevGenerator.java
package proizvodjac;

public class FibonacijevGenerator extends
                                  Generator {
  public FibonacijevGenerator() {
    super(new int[]{0, 1});
  }

  public int generisi() {
    int broj = klice[0] + klice[1];
    klice[0] = klice[1];
    klice[1] = broj;
    return broj;
  }
}
```

```
// TrougaoniGenerator.java
package proizvodjac;

public class TrougaoniGenerator extends
                                  Generator {
  public TrougaoniGenerator() {
    super(new int[]{0, 1, 1});
  }

  public int generisi() {
    int broj = klice[0] + klice[1];
    klice[0] = broj;
    klice[1] = klice[1] + klice[2];
    return broj;
  }
}
```

```
// Tacka.java
package grafika;

public class Tacka {
  private double x, y;

  public Tacka(double xx, double yy) {
    x = xx; y = yy;
  }

  public Tacka(){ this(0,0); }

  public double x() { return x; }
  public double y() { return y; }
}
```

```
// Luk.java
package package grafika;

import java.awt.*;

public class Luk {
  private Tacka c;
  private Graphics g;
  private double r;
  private double min, max;

  public Luk(Tacka cc, Graphics gg, int rr,
                  double mmin, double mmax){
    c = cc; g = gg; r = rr;
    min = mmin; max = mmax;
  }

  public Tacka t(double u) throws GOpseg{
    if(u<min || u>max) throw new GOpseg();
    return new Tacka( c.x() + r*Math.cos(u),
                      c.y() - r*Math.sin(u)); }

  public void crtaj() throws
                        InterruptedException{
    if(g!=null){
      double du = (2*Math.PI)/1000;
      try {
        Tacka t = t(min);
        int x0 = (int)(t.x()),
                            y0 = (int)(t.y());
        g.setColor (
          new Color((int)(Math.random()*256),
          (int)(Math.random()*256),
          (int)(Math.random()*256)));

        for (double ugao=min+du;  ugao<=max
                    && !Thread.interrupted();
                                    ugao+=du) {
          t = t(ugao);
          int x = (int)(t.x()),
                             y = (int)(t.y());
          g.drawLine (x0, y0, x, y);
          x0 = x; y0 = y;
          Thread.sleep(3);
        }
      } catch (GOpseg e) {}
    }
  }
}
```

```
// GOpseg.java
package grafika;
public class GOpseg extends Exception {
  public String toString() {
    return "Neispravno zadat ugao!";
  }
}
```

```
// GrafickiPrikazivac.java
package grafika;

import java.awt.*;
import java.awt.event.*;
import proizvodjac.*;

public class GrafickiPrikazivac extends
                                      Frame{
  private Generator gen = null;
  private Thread nit = null;
  private Luk luk = null;
  private Platno platno = null;
  private Checkbox rdgFibonacijev,
                                rdgTrougaoni;

  private class Platno extends Canvas
                        implements Runnable{
    private int x, y;

    public void paint(Graphics g) {
      if (nit != null){ nit.interrupt(); }
      (nit = new Thread(this)).start();
    }

    public void run() {
      if (rdgFibonacijev.getState()) {
          gen = new FibonacijevGenerator();
      } else if (rdgTrougaoni.getState()) {
            gen = new TrougaoniGenerator();
      }

      x = getWidth()/2; y = getHeight()/2;
      Graphics g = getGraphics();
      double uMin = 0, uMax = Math.PI/2;
      int kvadrant = 0;
      int broj = gen.generisi();

      try {
        while(!Thread.interrupted()) {
          luk = new Luk(new Tacka(x,y), g,
                            broj, uMin, uMax);
          luk.crtaj();

          int noviBroj = gen.generisi();
          kvadrant = (kvadrant+1)%4;

          x += ((kvadrant%2==0) ?
             ((kvadrant-1)*(noviBroj - broj))
                                          : 0);
          y += ((kvadrant%2==1) ?
             ((2-kvadrant)*(noviBroj - broj))
                                          : 0);

          broj = noviBroj;
          uMin = kvadrant*Math.PI/2;
          uMax = uMin+Math.PI/2;
        }
      } catch (InterruptedException e) {}
    }
  }
```

```
  public GrafickiPrikazivac(){
    super ("Nizovi");
    setBounds(100, 100, 300,300);
    setLayout(new BorderLayout());
    popuniProzor();
    setVisible(true);
    addWindowListener(new WindowAdapter() {
      public void windowClosing
                          (WindowEvent d) {
        if (nit != null){ nit.interrupt();}
        dispose();
      }
    });
  }

  private void popuniProzor() {
    Panel ploca = new Panel();
    add (ploca, "North");
    CheckboxGroup grp = new CheckboxGroup();
    ploca.add(rdgFibonacijev  =
          new Checkbox("Fibonacijev niz",
                                grp, true));
    ploca.add(rdgTrougaoni =
          new Checkbox("Trougaoni niz",
                          grp, false));

    platno = new Platno();
    add(platno, "Center");
    platno.setBackground(
                    new Color(220,220,220));

    ploca = new Panel();
    add (ploca, "South");
    Button dgm = new Button("Crtaj");
    ploca.add(dgm);
    dgm.addActionListener(
                    new ActionListener () {
      public void actionPerformed(
                          ActionEvent d){
        platno.repaint();
      }
    });
    ploca.add(dgm = new Button("Prekini"));
    dgm.addActionListener(
                    new ActionListener() {
      public void actionPerformed(
                          ActionEvent d){
        if (nit != null){ nit.interrupt();}
      }
      });
  }

  public static void main(String[] args) {
    new GrafickiPrikazivac();
  }
}
```