ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ
Одсек за СИ Београд 06.05.2015.

Други колоквијум из
Објектно оријентисаног програмирања II

1) *(30 поена)* Одговорити концизно (по једна или две реченице, или "да"/"не" где се не тражи образложење) и прецизно на следећа питања:

а) Да ли је и зашто боље да интерфејс садржи симболичке константе (коначна статичка поља) него тип набрајања чије се вредности користе као вредности аргумената метода интерфејса?

б) Да ли је могућ приступ јавном методу спољашње класе директним именовањем из метода анонимне класе и зашто?

в) Која је разлика између метода `notify()` и `notifyAll()` класе `Object`?

2) *(укупно 70 поена)* Саставити на језику Java следећи пакет класа:

- *(25 поена)* ***Пошиљка*** има једнозначан аутоматски генерисан целобројан идентификатор и адресу (ниска) на коју се шаље. Може да се дохвати адреса слања, да се направи копија пошиљке и да се састави текстуални опис у облику **`p_`***идентификатор***`[`***адреса***`]`**.
- Поштанско ***сандуче*** има адресу и садржи низ пошиљки задатог капацитета. Ствара се празно, после чега се пошиљке могу додавати појединачно. Може да се из сандучета извади прва пошиљка и да се извади пошиљка са задатом адресом примаоца (у случају да постоји више пошиљки које се шаљу на исту адресу, из сандучета се вади прва која се пронађе; при вађењу неке пошиљке, наредне се померају). При покушају узимања пошиљке из празног сандучета или покушаја додавања пошиљке у пуно сандуче, нит извршиоца радње се привремено блокира. Може да се дохвати адреса сандучета. Може да се састави текстуални опис сандучета наводећи пошиљке које се у њему налазе (сваку пошиљку у засебном реду).
- *(35 поена)* Активна ***особа*** има име, границе опсега времена (у *ms*) и придружено сандуче. У случајним временским интервалима, између најкраћег и најдужег времена (задатих при стварању особе), особа извршава одређену радњу. Грешка је ако су најкраће и најдуже време извршавања неисправно задати. Може да се састави текстуални опис особе наводећи име особе, а у посебном реду придружено сандуче.
- ***Поштар*** је особа која има листу сандучића у које доставља пошиљке. Ствара се са празном листом сандучића, после чега се сандучићи могу додавати појединачно. У сандучету придруженом поштару налазе се пошиљке које поштар разноси. Радња поштара је достава пошиљки на неку адресу. Адреса доставе је адреса прве пошиљке из придруженог сандучета, а достављају се (у оквиру исте радње) све пошиљке са том адресом слања.
- ***Прималац*** је особа. У сандучету придруженом примаоцу налазе се пошиљке које прималац узима, по редоследу којем се у сандучету налазе. Радња примаоца је узимање пошиљке из придруженог сандучета.

(10 поена) Написати на језику Java програм који направи једног поштара са придруженим сандучетом са неколико пошиљки, испише поштара, потом направи примаоца у чије придружено сандуче направљени поштар доставља пошиљке, после неког времена заустави поштара и примаоца и испише их. При писању програма користити константне параметре (не треба ништа учитавати).

НАПОМЕНЕ: **а)** Колоквијум траје **120** 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) 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.

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

```
// Posiljka.java
package posiljke;
public class Posiljka implements Cloneable {
   private static int posID = 0;
   private int id = ++posID;
   private String adr;
   public Posiljka(String adr){ this.adr = adr; }
   public String adr(){ return adr; }
   public String toString()
   { return "p_" + id + "[" + adr + "]"; }
   public Posiljka clone(){
      try {
         Posiljka p = (Posiljka)super.clone();
         p.id = ++posID; return p;
      } catch (CloneNotSupportedException e)
      { return null; }
   }
}
```

```
// Sanduce.java
package posiljke;
public class Sanduce {
   private String adr;
   private Posiljka niz[]; private int br;
   public Sanduce(String a,int k)
   { adr = a; niz = new Posiljka[k];}
   public synchronized Sanduce dodaj(Posiljka p)
                     throws InterruptedException {
      while(br == niz.length) wait();
      niz[br++] = p; notifyAll();
      return this;
   }
   public synchronized Posiljka izvadi(String adr)
                     throws InterruptedException {
      while(br == 0) wait();
      for(int i=0;i<br;i++)
         if(niz[i].adr().equals(adr))
            return izvadi(i);
      return null;
   }
   public synchronized Posiljka izvadi()
                    throws InterruptedException {
      while(br == 0) wait();
      return izvadi(0);
   }
   private Posiljka izvadi(int ind){
      Posiljka p = niz[ind];
      for(int i=ind+1;i<br;i++) niz[i-1] = niz[i];
      niz[br-1] = null; br--;
      notifyAll(); return p;
   }
   public String adr(){ return adr; }
   public synchronized String toString(){
      StringBuilder s = new StringBuilder();
      for(int i=0;i<br;i++)
      { s.append(niz[i]); s.append("\n"); }
      return s.toString();
   }
}
```

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

```
//Osoba.java
package posiljke;
public abstract class Osoba extends Thread {
   private String ime; protected Sanduce s;
   private int minT, maxT;
   public Osoba(String i, Sanduce ss,
                int min, int max) throws GOpseg {
      if(min > max) throw new GOpseg();
      ime = i; s = ss; minT = min; maxT = max;
      start();
   }
   public void run(){
      try {
         while(!interrupted()){
            radi(); sleep((long)(minT +
                     Math.random()*(maxT-minT)));
         }
      } catch (InterruptedException e) {}
   }
   public abstract void radi()
                     throws InterruptedException;
   public String toString(){ return ime+"\n"+s; }
}
```

```
// Postar.java
package posiljke;
public class Postar extends Osoba{
   private static class Elem{
      Sanduce sanduce; Elem sled;
      Elem(Sanduce s){ sanduce = s; }
   }
   private Elem prvi, posl;
   public Postar(String i, Sanduce s, int min,
   int max) throws Gopseg{ super(i, s, min, max);}
   public synchronized Postar dodaj(Sanduce s){
      Elem novi = new Elem(s);
      if(prvi == null) prvi = novi;
      else posl.sled = novi;
      posl = novi; return this;
   }
   public void radi()throws InterruptedException {
      Posiljka p = s.izvadi(); String a = p.adr();
      Elem tek = prvi;
      synchronized(this)
      {  for(;tek!=null;tek=tek.sled)
            if(tek.sanduce.adr().equals(a)) break;}
      if (tek == null) s.dodaj(p);
      else while (p!=null)
      { tek.sanduce.dodaj(p); p = s.izvadi(a); }
   }
}
```

```
// Primalac.java
package posiljke;
public class Primalac extends Osoba {
   public Primalac(String i, Sanduce ss, int min,
   int max) throws GOpseg { super(i,ss,min,max); }
   public void radi()throws InterruptedException
   {  s.izvadi(); }
}
```

```
// PosiljkeT.java
import posiljke.*;
public class PosiljkeT {
   public static void main(String[] args) {
      try {
         Posiljka p1 = new Posiljka("Vasina 17"),
            p2 = new Posiljka("Milesevska 25"),
            p3 = new Posiljka("Vasina 17");
         Postar p = new Postar("Petar Petrovic",
            new Sanduce("Terazije 1", 5).dodaj(p1)
            .dodaj(p2).dodaj(p3),1000, 5000);
         System.out.print(p);
         Sanduce s;
         Primalac r = new Primalac("Vasa Vasic",
            s  = new Sanduce("Vasina 17", 10),
            2000,7000);
         p.dodaj(s);
         Thread.sleep(10000);
         p.interrupt(); r.interrupt();
         System.out.print(r); System.out.print(p);
      } catch (Exception e){System.out.print(e);}
   }
}
```

```
Petar Petrovic
p_2[Milesevska 25]
p_3[Vasina 17]
p_1[Vasina 17]
Vasa Vasic
p_1[Vasina 17]
Petar Petrovic
p_2[Milesevska 25]
```