Др Душан Тошић

КОРИШЋЕЊЕ КОМУНИКАЦИОНИХ ПРОЗОРА УМЕСТО КОНЗОЛЕ У ПРОГРАМСКОМ ЈЕЗИКУ JAVA

У многим средњим школама у Србији још увек се уче процедурални програмски језици као што су C и Pascal. Постоје разна окружења за рад са овим језицима, међутим, најчешће се програми покрећу са командне линије („конзолни рад“). Највећи број таквих задатака може се урадити у програмском језику Java коришћењем конзолног рада и уношењем података коришћењем класе `Scanner`. Међутим, ученици више воле коришћење комуникационих прозора, тј. елементе графичког корисничког интерфејса. Увођењем неколико једноставних конструкција програмског језика Java, уместо конзоле, можемо користити комуникационе прозоре. Конкретно, уводи се класа `JOptionPane` из `swing`-а (стандардни пакет за подршку графичког корисничког интерфејса) и користе се два њена метода `showInputDialog()` и `showMessageDialog()`. Коришћењем ових метода практично се симулира конзолни рад употребом комуникационих прозора из графичког корисничког интерфејса. Пошто се у овим методима оперише само са стринговним подацима, потребно је знати и методе Jave за превођење стринговних података у бројчане (ако хоћемо да оперишемо са бројевима, што је чест случај).

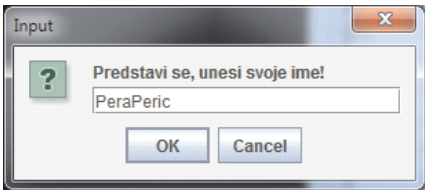
У следећим, релативно једноставним, примерима показаћемо како се „типични процедурални задаци“ могу решавати у програмском језику Java коришћењем неких елемената графичког корисничког интерфејса. У првом задатку демонстрира се коришћење поменутих метода класе `JoptionPane` коју треба увести из пакета `javax.swing`.

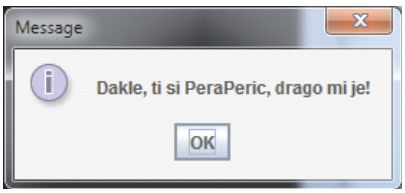
1. Написати Java програм за уношење имена ученика и штампање поздравног коментара у посебном прозору.

У следећем програму први аргумент (`null`) метода `showMessageDialog()` обезбеђује да се прозор са коментаром, тј. другим аргументом, прикаже у центру екрана.

```
import javax.swing.JOptionPane;
public class Dijalog{
  public static void main( String args[] )
{
      String ime;
    ime = JOptionPane.showInputDialog( "Predstavi se, unesi svoje ime!" );
    JOptionPane.showMessageDialog( null, "Dakle, ti si "+ime+", drago mi je!" );
  }
}
```

Тест пример:



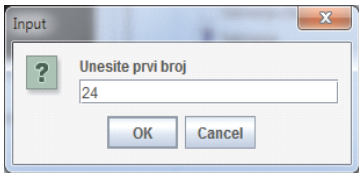


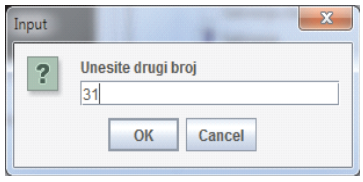
2. Написати програм за сабирање два цела броја и приказ резултата у прозору за приказ поруке. Сврха овог једноставног задатка је да покаже како се учитава цео број. У решењу задатка можемо видети да се цео број учитава као стринг, а потом преводи у целобројну вредност пошто је потребно да се над њим реализују аритметичке операције.

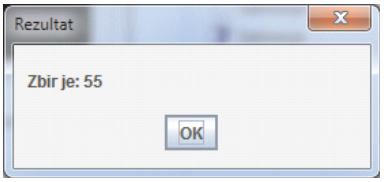
За превођење стринговне репрезентације целог броја у целобројну вредност, користи се метод `parseInt()` из класе `Integer`. Слични методи постоје у класам `Float` и `Double` за превођење разломљених бројева.

```
import javax.swing.JOptionPane;
  public class Sabiranje {
public static void main( String args[] )
  {
    String prviBroj;
      String drugiBroj;
    int broj1;
    int broj2;
    int suma;
    prviBroj =JOptionPane.showInputDialog( "Unesite prvi broj" );
      drugiBroj =JOptionPane.showInputDialog( "Unesite drugi broj" );
    broj1 = Integer.parseInt( prviBroj );
    broj2 = Integer.parseInt( drugiBroj );
    suma = broj1 + broj2;
    JOptionPane.showMessageDialog(
        null, "Zbir je:  " + suma, "Rezultat",JOptionPane.PLAIN_MESSAGE );
}
  }
```

Тест пример:







У овом случају коришћен је метод `showMessageDialog()` који има четири аргумента. Прва два имају исту улогу, трећи представља назив прозора са поруком, а четврти обезбеђује да се у прозору за резултат не појављује икона са обрнутим ускличником.

3. Написати програм за симулирање следеће игре са картама између корисника и рачунара. Корисник и рачунар наизменично извлаче по једну карту три пута. Побеђује онај ко има већи збир вредности на картама после трећег извлачења.

У овом примеру користимо конструкције као и у претходним примерима и уводимо Java-метод `Math.random()` за генерисање случајних бројева из интервала $(0, 1)$.

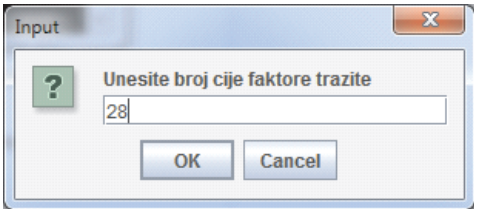
```
import javax.swing.JOptionPane;
public class Karte{

  public static void main( String args[] )
    {
      int n, racunar=0, igrac=0;
JOptionPane.showMessageDialog(
      null, "Vucemo 3 puta po jednu kartu.  Pobedjuje onaj ko ima veci zbir na
      kartama!" );
      for(int i=1; i<=3; i++){
        n=1+(int)(14*Math.random()); racunar +=n;
        JOptionPane.showMessageDialog(null, "U "+i+". izvlacenju dobio sam "+n);
        n=1+(int)(14*Math.random()); igrac +=n;
        JOptionPane.showMessageDialog(null, "U " +i +". izvlacenju ti si dobio "+n );
      }
  if(igrac>racunar) JOptionPane.showMessageDialog(null, "Bravo, ti si pobedio!
    Imas:  "+igrac+", a ja imam:  "+racunar );
    else JOptionPane.showMessageDialog(null, "Zao mi je, ja sam pobedio!  Imam:
      "+racunar+", a ti imas:  "+igrac);
  }
}
```

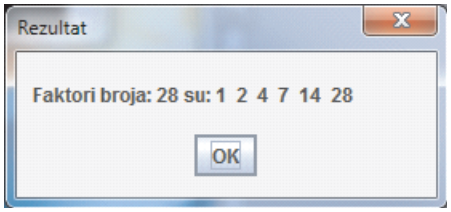
Због великог броја прозора са порукама, овде нећемо наводити тест примере, већ тестирање препуштамо читаоцу.

4. Написати програм за штампање свих фактора датог целог броја n. Овде практично нема нових елемената програмског језика Java. За сваки број од 2 до $n/2$ проверавамо да ли је фактор броја n. Скупу фактора прикључујемо 1 и сам број n.

```
import javax.swing.JOptionPane;
  public class Faktori {
public static void main( String args[] )
  {
  String ns, rez ="1";
  int n,i;
ns =JOptionPane.showInputDialog( "Unesite broj cije faktore trazite" );
n = Integer.parseInt( ns );
  for(i=2; i<=n/2; i++)
    if (n%i==0) rez+=" "+i;
JOptionPane.showMessageDialog(null, "Faktori broja: "+n+" su: "+rez+" "+n, "Rezultat",
JOptionPane.PLAIN_MESSAGE );
  }
}
```

Тест пример:





5. Написати метод за одређивање n-тог Фибоначијевог броја коришћењем

(а) итеративног

(б) рекурзивног

поступка. Тестирати написане методе учитавајући n коришћењем метода `showInputDialog()`.

За одређивање n-тог Фибоначијевог броја f_n природно се користи рекурзивна формула

$$f_n = f_{n-1} + f_{n-2} \quad (n = 3, 4, \dots) \quad \text{и} \quad f_1 = f_2 = 1.$$

Стога је и решење засновано на рекурзији једноставније, али брже се извршава програм у којем се користи итеративни поступак. Постоји могућност да се убрза рекурзивно израчунавање (коришћењем репне рекурзије у рекурзивном методу), али се тиме нећемо овде бавити.

(а)

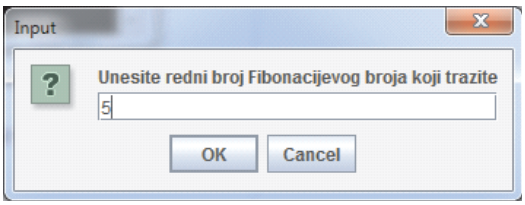
```
import javax.swing.JOptionPane;
  public class FibIter {
    static long fib(int k){
      long f=0,f1,f2;
      int i;
      if ((k == 1) || (k==2)) return 1;
      else{
        f1=1; f2=1;
        i=3;
        while(i<=k){
          f=f1+f2;
          f1=f2;
          f2=f;
            i++;
  }
            return f;
        }
      }
    public static void main( String args[] )
    {
String ns, rez;
int n;
ns = JOptionPane.showInputDialog( "Unesite redni broj Fibonacijevog broja koji
  trazite" );
n = Integer.parseInt( ns );
JOptionPane.showMessageDialog(
null,"Fibonacijev broj "+n+". po redu je:  "+fib(n), "Rezultat",
```

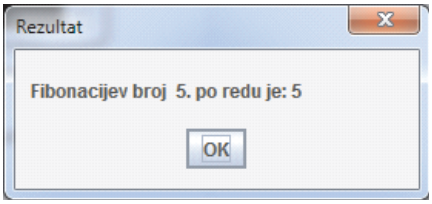
```
JOptionPane.PLAIN_MESSAGE );
    }
  }
```

(б)

```
import javax.swing.JOptionPane;
public class FibRec {
  static long fib(int k){
    if ((k == 1) || (k==2)) return 1;
    else return fib(k-1)+fib(k-2);
    }
  public static void main( String args[] )
  {
String ns, rez;
  int n;
    ns = JOptionPane.showInputDialog( "Unesite redni broj Fibonacijevog broja koji
      trazite" );
    n = Integer.parseInt( ns );
    JOptionPane.showMessageDialog(
      null,"Fibonacijev broj "+n+".  po redu je:  "+fib(n), "Rezultat",
      JOptionPane.PLAIN_MESSAGE );
  }
}
```

Тест пример:





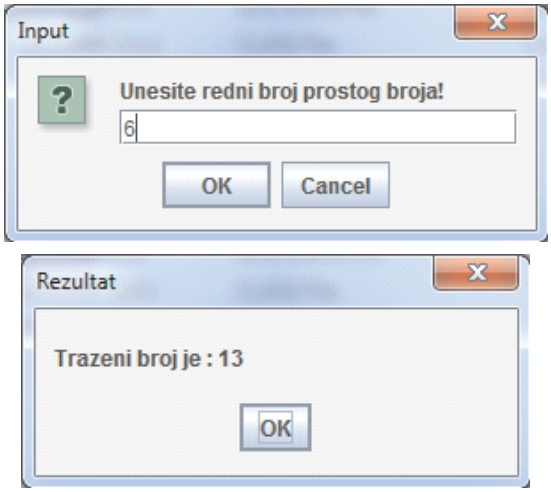
6. Написати програм који за дати природан број n одређује n-ти прост број.

Проверу да ли је број прост вршићемо проверавајући да ли је број дељив целим бројевима од 2 до целобројног дела квадратног корена датог броја увећаног за 1. За рачунање квадратног корена користи се метод `Math.sqrt()`.

```
import javax.swing.JOptionPane;
public class NtiProstBroj {
public static void main( String args[] )
  {
  String prviBroj;
    int n;
    prviBroj = JOptionPane.showInputDialog( "Unesite redni broj prostog broja!" );
      n = Integer.parseInt( prviBroj );
    int j=0;
    int k=2;
    boolean p=false;
    while(j<n){ p=true;
      for(int i=2; i<((int)( Math.sqrt(k)+1));i++ )
        if(k%i==0) p=false; break;;
      if(p)j++;
      k++;
    };
```

```
  JOptionPane.showMessageDialog(
      null, "Trazeni broj je :  " + (k-1),"Rezultat", JOptionPane.PLAIN_MESSAGE );
  }
}
```

Тест пример:

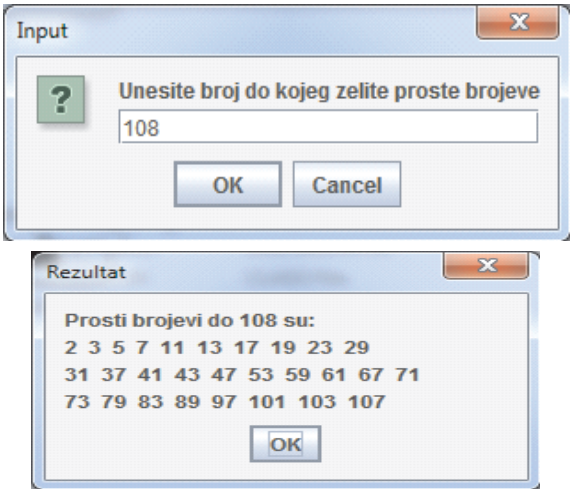


7. Написати програм за одређивање свих простих бројева мањих од n, при чему се n учитава помоћу метода `showInputDialog()`.

Овде се користи сличан алгоритам за налажење простих бројева и формирање тражене таблице. Излазни стринг `rez` у једном реду садржи максимално 10 простих бројева. За контролу колико у једном реду има простих бројева користи се бројач m.

```
import javax.swing.JOptionPane;
public class Prosti {
public static void main( String args[] )
{
  String ns, rez ="";
  int n,k,m,i;
  boolean ind;
  ns = JOptionPane.showInputDialog( "Unesite broj do kojeg zelite proste brojeve" );
  n = Integer.parseInt( ns );
    m=1;
  for(k=2; k<=n; k++){
    { ind = true;
      for(i=2; i<((int)( Math.sqrt(k)+1));i++)
        if (k%i==0) ind=false;
      if(ind) {rez+=k+" "; m++;}
    }
  if (m>10) {rez+="\n"; m=1;}
  }
  JOptionPane.showMessageDialog(
  null, "Prosti brojevi do "+n+" su:\n"+rez, "Rezultat", JOptionPane.PLAIN_MESSAGE );
  }
}
```

Тест пример:



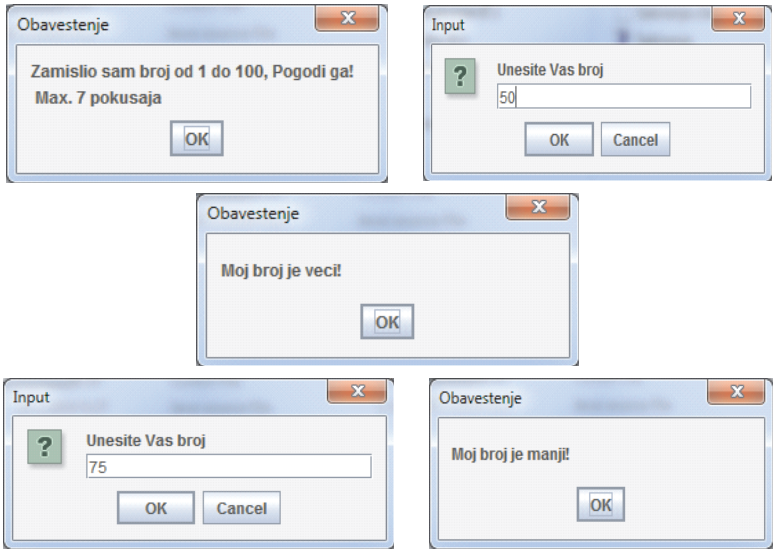
8. Написати програм за реализацију следеће игре: рачунар генерише случајан цео број из интервала $[1, 100]$. Задатак корисника је да у што мање покушаја одреди генерисани случајани број.

Изабрани број се сигурно може одредити у, максимално, 7 покушаја. Уколико корисник из 7 покушаја не одреди број, сматраћемо да је изгубио игру и штампати одговарајућу поруку. (Уколико желимо да погађамо број из интервала $[1, m]$, под условом да правилно играмо, број можемо сигурно погодити из $[\log_2 m] + 1$ покушаја, где је $[r]$ ознака за целобројни део броја r.)

```
import javax.swing.JOptionPane;
public class Pogadjanje{
  public static void main( String args[] )
    {
      int n, k;
      String p;
      boolean ind=true;
      n=(int)(100*Math.random()+1);

      JOptionPane.showMessageDialog(
    null, "Zamislio sam broj od 1 do 100, Pogodi ga!\n Max.  7 pokusaja",
    "Obavestenje",JOptionPane.PLAIN_MESSAGE );
        for(int i=1; i<=7; i++){
  p = JOptionPane.showInputDialog( "Unesite Vas broj" );
  k = Integer.parseInt( p );
  if(k==n){JOptionPane.showMessageDialog(
    null, "Bravo pogodili ste broj iz "+i+". pokusaja!","Obavestenje",
  JOptionPane.PLAIN_MESSAGE );ind =false; break;}
    else if(n>k)JOptionPane.showMessageDialog(null, "Moj broj je veci!",
      "Obavestenje",JOptionPane.PLAIN_MESSAGE); else if(n<k)
JOptionPane.showMessageDialog(null, "Moj broj je manji!",
  "Obavestenje",JOptionPane.PLAIN_MESSAGE);
      }
    if(ind) JOptionPane.showMessageDialog(null, "Niste pogodili broj iz 7
      pokusaja!","Obavestenje",JOptionPane.PLAIN_MESSAGE );
```

```
  }
}
```

Тест пример (због великог броја комуникационих прозора, који се појављују при тестирању, овде наведимо само неколико):



ЛИТЕРАТУРА

[1] C. Horstman, G. Cornell: *Java 2, Tom I-Osnove*, CET i RAF, 2007.

[2] I. Horton, *Java2 – JDK 5*, CET, Beograd, 2006.

[3] J. Zukowski, *Java 2 J2SE 1.4*, Kompjuter biblioteka, Čačak, 2004.

[4] K. Arnold, J.Gosling, D. Holmes, *Programski jezik Java*, CET, Beograd, 2001.

[5] H. Schildt, *Java J2SE 5: kompletan priručnik*, Mikro knjiga, Beograd, 2006.

Катедра за рачунарство и информатику, Математички факултет, Универзитет у Београду
E-mail: `tosic@matf.bg.ac.rs`