Poo Laboratoare[1]

Contents

[1] http://www.google.ro/

Laborator5

Programare Orientata pe Obiecte: Laborator 5

1 Citire și scriere de fișiere în Java

1.1 Noțiuni de bază

- Pachetul care oferă suport pentru operațiile de intrare/ieșire este: **java.io**
- Atenție! Pentru o descriere completă a claselor puse la dispoziție de acest pachet, folosiți documentația JDK.

1.1.1 Fluxuri primitive

- Se ocupă cu citirea/scrierea efectivă a informațiilor într-un stream
- Exemple de clase:
  - FileReader, FileWriter
  - FileInputStream, FileOutputStream
- Crearea unui flux de intrare pe caractere :

```
FileReader in = new FileReader( ‘‘fisier.txt’’ );
```

1.1.2 Fluxuri de procesare

- Sunt responsabile cu preluarea datelor de la un flux primitiv și procesarea acestora pentru a le oferi într-o altă formă
- Exemple de clase:
  - BufferedReader, BufferedWriter
  - BufferedInputStream, BufferedOutputStream
- Crearea unui flux de intrare printr-un buffer :

```
BufferedReader = new BufferedReader( new FileReader( ‘‘fisier.txt’’ ) );
```

1.2 Exemplu

```
import java.io.*;
class CitireScriere
{
   String buf = ‘‘’’;
   public void citeste(String fisier)
   {
      int c;
      FileReader f = null;
      try {
         f = new FileReader(fisier);
         while ((c = f.read()) != -1) {
            buf = buf + (char)c;
         }
         f.close();
```

```
        }
        catch (FileNotFoundException e) {
           System.out.println(``Fisierul nu a fost gasit'');
        }
        catch (IOException e) {
           System.out.println(``Eroare la citire'');
        }
     }
     public void scrie(String fisier)
     {
        FileWriter f = null;
        try {
           f = new FileWriter(fisier);
           f.write(buf);
           f.close();
        }
        catch (IOException e) {
           System.out.println(``Eroare la scriere'');
        }
     }
     public static void main(String argv[])
     {
        CitireScriere c = new CitireScriere();
        c.citeste(``text.txt'');
        c.scrie(``out.txt'');}
      }
}
```

2 Probleme de laborator

2.1 Problema 1

(1 punct) Să se scrie un program pentru afișarea pe ecran a liniilor dintr-un fișier text, fiecare linie precedată de numărul liniei și de un spațiu. Se va folosi clasa *LineNumberReader*. Numele fișierului se dă în linia de comandă. Toate exceptiile de I/E sunt *aruncate* mai departe astfel:

```
public static void main (String argv[]) throws IOException { ...  }
```

2.2 Problema 2

(1 punct) Să se scrie un program pentru afișarea numărului de linii și numărului de cuvinte dintr-un fișier text. Cuvintele sunt separate prin spații albe. Exemplu de utilizare a clasei *StringTokenizer* (pachetul java.util) pentru extragere de cuvinte separate prin delimitatori din șirul *delim*, dintr-un șir *sir* :

```
StringTokenizer st = new StringTokenizer(sir, delim);
while (st.hasMoreTokens()) {
   String token = st.nextToken();
   ...  // token contine un cuvant
}
```

2.3 Problema 3

(1 punct) Să se scrie un program pentru citirea unui fișier text și crearea unui alt fișier în care toate literele din fișierul inițial sunt trecute în litere mari. Se va folosi metoda *read* din clasa *FileReader* pentru citirea unui caracter (metoda *read* are ca rezultat -1 la sfârșit de fișier).

Se poate folosi metoda statică: char Character.toUpperCase(char c)

2.4 Problema 4

(1 punct) Să se utilizeze clasa *PrintStream* pentru a scrie numere întregi într-un fișier text pe disc. Să se modifice apoi programul înlocuind clasa *PrintStream* cu clasa *PrintWriter* (cu aceleași metode *print* si *println*). Fișierul va fi verificat cu un editor de texte.

Să se scrie un program pentru citirea și afișarea fișierului de numere creat (cu *RandomAccessFile*).

```
Exemple de creare obiecte:
  PrintStream ps = new PrintStream( new FileOutputStream(filename));
  PrintWriter pw = new PrintWriter( new FileWriter(filename));
```

2.5 Problema 5

(1 punct) Să se rescrie programul anterior astfel ca numerele să fie scrise în format binar folosind clasa *DataOutputStream* (metoda *writeInt*). Program pentru citirea și afișarea pe ecran a fișierului de numere creat folosind metoda *readInt* din clasa *DataInputStream*.

2.6 Problema 6

(1 punct) Să se scrie un program pentru citirea de linii de la tastatura și afișarea lor pe ecran, folosind metoda *readLine* pentru un obiect de tip *DataInputStream* sau *BufferedReader*.

2.7 Problema 7

(1 punct) Să se rescrie programul anterior utilizând clasa Scanner.

2.8 Problema 8

(1 punct) Să se scrie un program pentru citirea integrală a unui fișier text în memorie, urmată de afișarea sa pe ecran prin două metode:

- Folosind metoda *readFully* din clasa *RandomAccessFile*
- Folosind operatorul $+$ pentru adăugarea unei linii la textul citit anterior.

2.9 Problema 9

(2 puncte) Program pentru cautarea unui șir dat în toate fișierele dintr-un director dat și afișarea numelor fișierelor care conțin acest șir.

2.10 Problema 10 (bonus)

(1 punct) Să se scrie un program pentru copierea unui fișier mare folosind succesiv clasele *FileInputStream*, *FileOutputStream*, apoi clasele *FileReader*, *FileWriter* și apoi clasele *BufferedReader* și *BufferedWriter*. Comparați timpul de copiere pentru cele trei cazuri.

2.11 Problema 11 (bonus)

(1 punct) Să se scrie un program pentru afișarea numerelor, cuvintelor și caracterelor speciale (operatori, delimitatori) dintr-un program sursă Java, folosind un obiect de tip *java.io.StreamTokenizer*. Numele fișierului se dă în linia de comandă sau se citește de la tastatură.