Lucrare de laborator nr. 12
Fişiere (II)

1. Scopul lucrării

În această lucrare se studiază clasele *FileWriter* şi *BufferedWriter*, foloste pentu a scrie în fişiere text. De asemenea se studiază câteva clase ce se folosesc pentru prelucrarea fişierelor binare: *FileInputStream, FileOutputStream, DataInputStream* şi *DataOutputStream*.

2. Breviar teroretic

2.1 Clase pentru scrierea în fişiere text.

Clasa FileWriter.

Este folosită pentru a scrie un fișier text caracter cu caracter.

Constructorul:

public FileWriter(String numeFisier)

Metode:

public void write(char ch)

Este folosita pentru a scrie în fișier, pe poziția curentă, caracterul dat ca parametru.

Clasa BufferedWriter.

Se foloseşte pentru a scrie un fişier text, linie cu linie .

Constructorul:

public BufferedWriter(FileWriter fw)

Metode:

a) public write(String s, int indexStart, int L)

Scrie in fișier, pe poziția curentă, porțiunea din șirul s (dat ca parametru), de lungime L caractere, incepând cu poziția indexStart din șir.

b) public write(String s)

Scrie în fișier, pe poziția curentă, şirul s (dat ca parametru)

c) public void newLine( )

Scrie un ENTER în fișier (trece pe linia următoare).

Clasa FileInputStream.

Se foloseşte pentru a citi un fişier binar, octet cu octet .

Constructorul:

public FileInputStream (String numeFisier)

Metode:

public int read()

Citeşte octetul curent din fişier. În cazul în care s-a atins sfârşitul de fişier, returnează -1.

Clasa FileOutputStream.

Se foloseşte pentru a scrie un fişier binar, octet cu octet .

Constructorul:

public FileOutputStream (String numeFisier)

Metode:

public void write(int b)

Scrie octetul dat ca parametru, în fişier, pe poziţia curentă.

Clasa DataInputStream.

Se foloseşte pentru a citi tipuri primitive de date (int, float, etc) dintr-un un fişier binar (în cazul general dintr-un stream de intrare) .

Constructorul:

public DataInputStream (InputStream is)

Metode:

public int readInt()

Citeşte o dată de tip int din fişier.

public char readChar()

Citeşte o dată de tip char din fişier.

public double readDouble()

Citeşte o dată de tip double din fişier.

Clasa DataOutputStream.
Se foloseşte pentru a scrie tipuri primitive de date (int, float, etc.) într-un un fişier binar (în cazul genereal într-un stream de ieşire) .
Un program poate să scrie date primitive într-un stream de ieşire, care ulterior vor fi citite folosind metode ale clasei **DataInputStream**.
Constructorul:

public DataOutputStream (OutputStream is)

Metode:

public void writeInt(int a)

Scrie o dată de tip int în fişier.

public void writeChar(int ch)

Scrie o dată de tip char în fişier.

public void writeDouble(double x)

Scrie o dată de tip double în fişier.

3. Probleme rezolvate

Problema 1

Se dă fişierul text test1.txt, aflat in directorul curent. Să se copieze acest fişier, în fişierul test2.txt, convertind literele mici în litere mari, restul caracterelor ramânând neschimbate.

Pentru a converti o literă mică la litera mare corespunzătoare se foloseşte metoda statică *toUpperCase*, din clasa *Character*.

Ea are următoarea semnătură:

public static char toUpperCase(char c)

```
import java.io.*;
class FisierTextLitereMari
{
 public static void main(String args[])
 {
  FileReader fr=null;
  FileWriter fw=null;
  try{
    fr=new FileReader("test1.txt");
    fw=new FileWriter("test2.txt");
    for(;;){
      int cod=fr.read();
```

```
    if(cod==-1)break;//s-a terminat fisier
    char c=(char)cod;
    char cMare=Character.toUpperCase(c);
    fw.write(cMare);
  }
  fr.close();
  fw.close();
 }catch(IOException e){
  System.out.println(e);
  System.exit(1);
 }
}//main
}
```

Problema 2

Se citeşte de la tastatură un număr natural N. Se citesc apoi N propoziţii de la tastatură. Sa se copieze acestea într-un fişier text, fiecare propoziţie pe câte o linie.

Citirea de la tastatură a datelor (numărul N şi propoziţiile) se va face în modul text, folosind metoda *readLine()* din clasa *BufferedReader* . Obiectul *BufferedReader* folosit pentru a citi şiruri de la tastatură se instanţiază astfel:

```
BufferedReader bfr=
            new BufferedReader(new InputStreamReader(System.in));
```

```
import java.io.*;
class CitireSiruriSiSalvare
{
 public static void main(String args[])
 {
  BufferedReader bfr=null;
  FileWriter fw=null;
  BufferedWriter bfw=null;
  try{
   fw=new FileWriter("propozitii.txt") ;
   bfw=new BufferedWriter(fw);
   bfr=new BufferedReader(new InputStreamReader(System.in));
   System.out.print("N=");
   int N=Integer.parseInt(bfr.readLine());
   for(int i=0;i<N;i++){
```

```
   System.out.print("sir = ");
   String s=bfr.readLine();
   bfw.write(s);
   bfw.newLine();
  }
  bfr.close();
  bfw.close();
  fw.close();
 }catch(IOException e){
   System.out.println(e);
   System.exit(1);
 }
}
}
```

Problema 3

Să sc creeze prin program un fişier text, ce conţine 26 de linii , fiecare linie conţine toate cele 26 de litere ale alfabetului, dar prima linie începe cu A, a doua cu B, a treia cu C, etc.

ABCDEFGHIJKLMNOPQRSTUVWXYZ
BCDEFGHIJKLMNOPQRSTUVWXYZA
CDEFGHIJKLMNOPQRSTUVWXYZAB
…….
YZABCDEFGHIJKLMNOPQRSTUVWX
ZABCDEFGHIJKLMNOPQRSTUVWXY

```
import java.io.*;
class ScrieInFisierAlfabetul
{
 public static void main(String args[])
 {
  final int N=26; //numarul de linii din fisier
                  // (sunt 26 de litere in alfabet englez)
  int i,j;
  FileWriter fw=null;
  BufferedWriter bfw=null;
  try{
   fw=new FileWriter("alfabet.txt");
   bfw=new BufferedWriter(fw);
   String s="ABCDEFGHIJKLMNOPQRSTUVWXYZ";
```

```
   bfw.write(s);
   bfw.newLine();
   //scriem urmatoarele N-1 linii in fisier, obtinute prin
   // permutarea corespunzatoare a lui s:
   for(i=1;i<N;i++){
    //Construim linia curenta:
    String crt="";
    for(j=i;j<N;j++)
        crt=crt+s.charAt(j);
    //in continuare copiem de la inceputul lui s:
    for(j=0;j<i;j++)
        crt=crt+s.charAt(j);
    //scriem linia crt in fisier:
    bfw.write(crt);
    bfw.newLine();// scrie ENTER in fisier
    }
   bfw.close();
   fw.close();
  }catch(IOException e){
      System.out.println(e);
      System.exit(1);}
 }
}
```

Problema 4

Se dă fişierul text *noteClasa.txt* care conţine pe fiecare linie numele complet al unui elev (numele de familie şi unul sau mai multe prenume) şi nota obţinută de acesta la o lucrare de control (nota fiind un număr întreg). Considerăm că notele pot fi: 0, 1, 2,…,9 , 10 (nota 0 poate să semnifice de exemplu că elevul respectiv a fost absent la lucrarea de control). Să se creeze fişierul *noteSortateClasa.txt*, care se obţine din fişierul *noteClasa.txt*, prin gruparea elevilor după note: la început toţi elevii cu nota 0, apoi toţi cu nota 1, etc. O linie din fişierul *noteSortateClasa.txt* are aceeaşi structură ca şi o linie din fişierul *noteClasa.txt* (numele complet al unui elev şi nota).

```
import java.io.*;
import java.util.*;
class SortareClasa
{
```

```
public static void main(String args[])
{
 FileReader fr=null;
 BufferedReader bfr=null;
 FileWriter fw=null;
 BufferedWriter bfw=null;
 final int N_VALORI_NOTE=11; //11 note diferite: 0, 1,...,10
 String nume[][]=new String[N_VALORI_NOTE][100];//100 este o
                    // dimensiune acoperitoare, pt nr elevi din clasa
 int contorNota[]=new int[N_VALORI_NOTE];
 int i;
 for(i=0;i<N_VALORI_NOTE;i++)
  contorNota[i]=0;
 try{
  fr=new FileReader("noteClasa.txt");
  bfr=new BufferedReader(fr);
  fw=new FileWriter("noteSortateClasa.txt");
  bfw=new BufferedWriter(fw);
  for(;;){
   String s=bfr.readLine();
   if(s==null)break;
   //extragem numele complet al elevului si nota:
   StringTokenizer tk=new StringTokenizer(s);
   int n=tk.countTokens();
   //Primii n-1 atomi reprezinta numele, ultimul atom este nota:
   String numeCrt="";
   for(i=1;i<=n-1;i++)
       numeCrt=numeCrt+tk.nextToken()+" ";
   int notaCrt=Integer.parseInt(tk.nextToken());
   //introducem corespunzator aceste date:
   nume[notaCrt][contorNota[notaCrt]]=numeCrt;
   contorNota[notaCrt]++;
  }
  //pacurgem matricea nume le copiem in fisierul noteSortateClasa:
  int j;
  for(i=0;i<N_VALORI_NOTE;i++){
    //vom scrie elevii de la nota i:
    for(j=0;j<contorNota[i];j++){
        bfw.write(nume[i][j]+" "+i);
        bfw.newLine();
```

```
      }
    }
    bfw.close();
    fw.close();
    bfr.close();
    fr.close();
  }catch(IOException e){
        System.out.println(e);
        System.exit(1);
  }
 }
}
```

Problema 5

Se dă un fişier text în care este memorată o imagine binară (alb- negru) ce conţine N1xN1 pixeli (valori de 0 sau 1, separate printr-un spaţiu în cadrul aceleiaşi linii). Valoarea N1 este cunoscută. Să se construiască un alt fişier text ce conţine imaginea anterioară redusă la N2xN2 pixeli (N2 cunoscut şi este un divizor al lui N1). Algoritmul de reducere este următorul:

Se calculează dimensiunea unui bloc (notata cu dimBloc) dimBloc= N1/N2, şi apoi imaginea iniţială de N1xN1 pixeli se împarte în blocuri de dimBloc linii şi dimBloc coloane. Fiecare bloc va fi redus la un pixel, în matricea redusă (ce va avea dimensiunea N2xN2). Valoarea pixelului se calculează astfel: este 1 dacă numărul de pixeli de 1 din bloc este >= decât numărul de pixeli de 0 din bloc, şi este 0 în caz contrar.

```
import java.io.*;
import java.util.*;
class ReduceFisier{
 public static void main (String args[])
{
  final int N1=100;
  final int N2=10;
  int dimBloc=N1/N2;
  FileReader fr = null;
  BufferedReader bfr=null;
  int a[][] = new int[N1][N1];
  int i,j;
```

```
try{
 //copiem fisierul mare, ce are N1 linii, intr-o matrice:
 fr = new FileReader("unu.txt");
 bfr = new BufferedReader(fr);
 for (i=0;i<N1;i++)
 {
  String s = bfr.readLine();
  StringTokenizer tk = new StringTokenizer(s);
  for(j=0;j<N1;j++)
 {
     String atomCrt = tk.nextToken();
     a[i][j]=Integer.parseInt(atomCrt);
  }
 }
 bfr.close();
 fr.close();
}catch (IOException e){
  System.out.println(e);
  System.exit(1);}
//Reducem matricea si o copiem in al doilea fisier:
try {
   int b[][] = new int [N2][N2];
  //construim matricea redusa, b[][]:
  for (i=0;i<N2;i++)
     for (j=0;j<N2;j++){
      int contor1 = 0;
      for (int k = i*dimBloc;k<(i+1)*dimBloc;k++)
       for (int l = j*dimBloc;l<(j+1)*dimBloc;l++)
          if (a[k][l] == 1)contor1++;
       if (contor1>=dimBloc*dimBloc/2)b[i][j] = 1;
       else b[i][j] = 0;
      }
      //copiem matricea redusa, in fisier:
     FileWriter fw = new FileWriter ("doi.txt");
     BufferedWriter bfw = new BufferedWriter (fw);
     for (i=0;i<N2;i++){
       String st="";
       for (j=0;j<N2;j++)
        st = st + b[i][j] + " ";
       bfw.write(st,0,st.length() - 1);
```

```
        bfw.newLine();
       }
      bfw.close();
      fw.close();
   }catch (IOException e){
   System.out.println(e);
System.exit(1);
  }
 }
 }
```

Problema 6

Să se calculeze maximul dintr-un fişier binar de octeţi. Numele acestui fișier se citește de la tastatură.

```
import java.io.*;
class MaximOctet
{
  public static void main(String args[])
  {
   FileInputStream fi=null;
   String numeF=JOptionPane.showInputDialog("nume fisier=");
   byte max=0; //initializare maxim
   try{
    fi=new FileInputStream(numeF);
    for(;;){
      int cod=fi.read();
      if(cod==-1)break; //s-a terminat fisierul
      byte b=(byte)cod;//octetul curent citit
      if(b>max)max=b;
    }
      fi.close();
   }catch(IOException e){
System.out.println(e);
System.exit(1);
  }
  System.out.println("maxim="+max);
 }
}
```

Problema 7

Se dă un fişier binar de octeţi, date.bin. Să se copieze acest fişier în NFIS fişiere binare (NFIS – contantă ce se cunoaşte): date1.bin, date2.bin, date3.bin, etc., aceste fişiere conţin părţi egale din fişierul iniţial date.bin (cu eventuala excepţie a ultimului fişier din cele NFIS).

Exemplu:

Dacă NFIS=2, atunci prima jumătate din date.bin se copiază în date1.bin, cealaltă jumătate se copiază în date2.bin.

```
import java.io.*;
class SplitBinaryFile
{
  public static void main(String args[])
  {
   final int NFIS=3;//numar de fisiere
   final int DIM=10000;//dim. acoperitoare
   final String numeFisier="date.bin";
   FileInputStream fi=null;
   byte b[]=new byte[DIM];
   int N=0;//numar de octeti din fisier (initializare)
  //copiere fisier in vector de octeti:
   try{
     fi=new FileInputStream(numeFisier);
    for(;;){
      int cod=fi.read();
      if(cod==-1)break; //s-a terminat fisierul
      b[N]=(byte)cod;
      N++;
     }
    fi.close();
  }catch(IOException e){
        System.out.println(e);
        System.exit(1);
  }
  //Generare vector de nume de fisiere de iesire:
 //(in exemplul folosit: date1.txt, date2.txt, date3.txt)
 String nume[]=new String[N];
//Pentru a extrage din numeFisier, numele fara extensie:
StringTokenizer tk=new StringTokenizer(numeFisier,".");
```

```
 String numeBaza=tk.nextToken();
 String numeExtensie=tk.nextToken();
int i,j;
for(i=0;i<NFIS;i++)
  nume[i]=numeBaza+(i+1)+"."+numeExtensie;
  for(i=0;i<NFIS;i++)
   System.out.println(nume[i]);
 //Construirea celor NFIS fisiere:
FileOutputStream fo=null;
//nr. de octeti ce se copiaza intr-un fisier din cele NFIS:
int dim=N/NFIS;
try{
  for(i=0;i<NFIS;i++){
    fo=new FileOutputStream(nume[i]);
   //se va copia din vectorul b[]:
    int indexStart=i*dim;
    int indexStop=indexStart+dim;
    if(i==NFIS-1)indexStop=N; //daca este ultimul fisier
    for(j=indexStart;j<indexStop;j++)
      fo.write(b[j]);
    fo.close();
   }
 }catch(IOException e){
System.out.println(e);
System.exit(1);
 }
 }
}
```

Problema 8

Se citeşte un număr natural N. Se citesc N numere întregi, care se vor copia într-un fişier binar de numere întregi.

```
import java.io.*;
import javax.swing.*;
class CopiereNumereInFisier
{
public static void main(String args[])
{
  int N=Integer.parseInt( JOptionPane.showInputDialog("N="));
```

```
  FileOutputStream fos=null;
  DataOutputStream f=null;
  try{
   fos=new FileOutputStream("numere.dat");
   f=new DataOutputStream(fos);
   //Citim cele N numere de la tastatura:
   for(int i=0;i<N;i++){
    int nr=Integer.parseInt(JOptionPane.
                                    showInputDialog("nr="));
     //Scriem numarul in fisier:
     f.writeInt(nr);
   }
  f.close();
  fos.close();
 }catch(IOException e){
System.out.println(e);
System.exit(1);
 }
 }
}
```

Problema 9

Să se calculeze media aritmetică a numerelor dintr-un fişier binar de numere întregi.

```
import java.io.*;
import javax.swing.*;
class MediaDinFisier
{
 public static void main(String args[])
 {
  File f=null;
  FileInputStream fis=null;
  DataInputStream dis=null;
  String numeF=JOptionPane.showInputDialog("nume fisier=");
  try{
    f=new File(numeF);
    long L=f.length();//lungimea in octeti
    int N=(int)L/4;//Numarul de intregi memorati in fisier
                    //(un int ocupa 4 octeti )
```

```
   fis=new FileInputStream(f);
   dis=new DataInputStream(fis);
   int suma=0;
   //Citim cele N numere din fisier:
   for(int i=0;i<N;i++){
     int nr=dis.readInt();
     suma=suma+nr;
   }
  double medie=suma/(double)N;
 System.out.println("medie="+medie);
 dis.close();
 fis.close();
 //f.close(); Nu exista close() pentru File
 }catch(IOException e){
   System.out.println(e);
   System.exit(1);
 } catch(SecurityException se){
   //pentru metodele din clasa File
   System.out.println(se);
   System.exit(1);
  }
 }
}
```

3. Probleme propuse

Problema 1

Citim de la tastatură o linie de text. Să copiem caracterele din această linie în fişierul *linie.txt*, câte un caracter pe fiecare linie a fişierului text.

Problema 2

Se da fişierul text test1.txt, aflat in directorul curent. Să se copieze acest fişier, în fişierul test2.txt, convertind literele mici în litere mari, restul caracterelor ramânând neschimbate. Spre deosebire de problema rezolvată 1, vom citi fişierul test1.txt linie cu linie.

Pentru a converti un şir s1 la litere mari şi a-l copia în şirul s2, se foloseşte metoda *toUpperCase()* astfel:

```
String s2=s1.toUpperCase();
```

Problema 3

Se citeşte de la tastatură un număr natural N. Să se creeze prin program, un fişier text ce are 10 linii, astfel: în prima linie N valori de 0, separate între ele prin câte un spaţiu; în a doua linie N valori de 1 separate între ele prin câte un spaţiu, etc.

Astfel pentru N=3 fişierul are următorul conţinut:

0 0 0
1 1 1
2 2 2
3 3 3
4 4 4
5 5 5
6 6 6
7 7 7
8 8 8
9 9 9

Problema 4

Citim de la tastatură o propoziţie. Să copiem fiecare cuvânt din această propoziţie. în fişierul *cuvinte.txt*, câte un cuvânt pe o linie separată din fişierul text.

Problema 5

Să se calculeze câte valori nule sunt într-un fişier binar de octeţi, al cărui nume se citeşte de la tastatură.

Problema 6

Să se afişeze dacă toţi octeţii dintr-un fişier binar, sunt diferiţi între ei sau nu.

Algoritm:

Se vor copia toți octeții din fișier într-un vector (se va aloca o dimensiune acoperitoare acestui vector). Apoi se va aplica algoritmul de la vectori pentru toate numerele diferite.

Problema 7

Să se calculeze și afișeze maximul dintr-un fişier binar de numere întregi. Numele acestui fișier se va citi de la tastatură.

Problema 8

Să se afişeze dacă toate numerele dintr-un fişier binar de numere întregi sunt în progresie aritmetică sau nu. Numele fișierului se va citi de la tastatură.