*Національний технічний університет України*

*‘Київський політехнічний інститут’*

*Факультет інформатики та обчислювальної техніки*

*Кафедра обчислювальної техніки*

***Лабораторна робота №2***

*“ Організація мультипотокової обробки даних в Java ”*

*Виконав: Мроць Ю.Б.*

*Група: ІО-12*

*Номер зк: 1219*

*Перевірив: Абу Усбах О.Н.*

*Київ — 2013*

***Мета***

Отримання базових навичок з використання мультипотокового середовища в Java. Поглиблене вивчення механізму виключень. Ознайомлення з засобами Java для розбору текстових даних.

***Завдання***

1. Вивчити механізм виключень (Exceptions) в Java. Вміти створювати нові класи виключень, інстанціювати виключення, маскувати місце виникнення виключення та його тип. Знати призначення інтерфейсу Throwable.

2. Ознайомитись з засобами розбору даних. Вміти використовувати класи StreamTokenizer, StringTokenizer, Scanner.

3. Вивчити механізм організації множини легких процесів - потоків (threads). Вміти створювати потоки за допомогою класу Thread та інтерфейсу Runnable.

4. Вивчити механізми синхронізації у мультипотоковому середовищі.

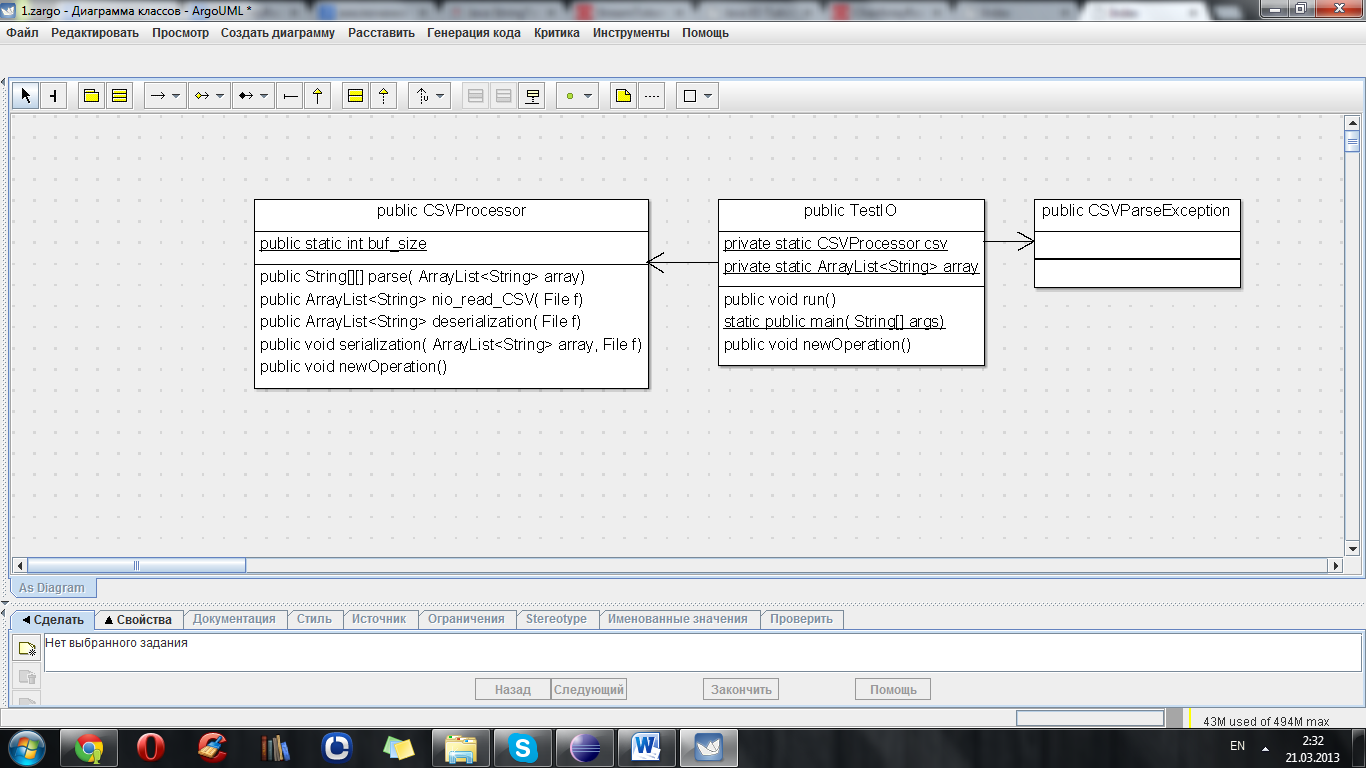
5. Вивчити стани потоку. Знати причини та засоби запобігання блокуванню потоків. Вміти керувати станами потоків, змінювати їх пріоритет.

6. Створити клас виключень CSVParseException. Обґрунтувати його розміщення в ієрархії виключень Java.

7. В класі CSVProcessor створити метод parse(), який виконує розбір завантажених в попередній роботі даних (з агрегованого об’єкту класу ArrayList) в двовимірний масив. Під час розбору перехоплювати всі виключення. В разі виникнення виключення, або невідповідності даних формату (не є CSV, не вистачає значень або є зайві, тощо) генерувати виключення класу CSVParseException.

8. Модифікувати тестовий клас TestIO. В методі main() додати створення нового потоку розбору даних (за допомогою метода parse() ), при зупинці основного потоку до отримання результатів розбору.

***Діаграма класів***



***Лістинг коду***

**package** lab1;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.FileNotFoundException;

**import** java.io.FileOutputStream;

**import** java.io.IOException;

**import** java.io.ObjectInputStream;

**import** java.io.ObjectOutputStream;

**import** java.nio.ByteBuffer;

**import** java.nio.channels.FileChannel;

**import** java.nio.charset.Charset;

**import** java.util.ArrayList;

/\*\*

\* У класі CSVProcessor реалізувані методи завантаження/вивантаження CSV-файлу в/з агрегований об’єкт класу ArrayList,

\* в якому знаходяться рядки CSV-файлу. Реалізувати методи серіалізації/десеріалізації агрегованого об’єкту класу ArrayList

\* в файлі serialized.dat.

\* **@author** Aruy

\*/

**public** **class** **CSVProcessor** {

/\*\*

\* Розмір буфера.

\*/

**private** **static** **int** *buf\_size*;

/\*\*

\* Конструктор

\* **@param** buf\_size розмір буфера.

\*/

**public** CSVProcessor(**int** buf\_size) {

*setBuf\_size*(buf\_size);

}

/\*\*

\* Метод зчитує з об'єкту типу ArrayList<String> дані і записує їх в об'єкт типу String[][].

\* **@param** array об'єкт звідки треба зчитати дані

\* **@return** об'єкт, типу String[][]

\* **@throws** CSVParseException

\*/

**public** String[][] parse(ArrayList<String> array) **throws** CSVParseException{

**int** l = 0;

**int** h = 0;

**int** max = 0;

**for** (**int** i = 0; i < array.size(); i++) {

String k = array.get(i);

**if** (k.equals("\n")) {

h++;

**if** (max < l)

max = l;

l = 0;

} **else** {

l++;

}

}

String[][] a = **new** String[h][max];

**int** i = 0;

**int** b = 0;

**for** (**int** j = 0; j < array.size(); j++) {

String k = array.get(j);

**if** (!k.equals("\n")) {

a[i][j - b] = k;

} **else** {

i++;

b = j + 1;

}

}

**return** a;

}

/\*\*

\* Метод серіалізує дані об'єкту array у файл

\* **@param** array об'єкт, що серіалізуватиметься

\* **@param** f об'єкт типу File, що містить шлях до файлу, в який серіалізується об'єкт

\*/

**public** **void** serialization (ArrayList<String> array, File f) {

ObjectOutputStream oos = **null**;

**try** {

oos = **new** ObjectOutputStream(**new** FileOutputStream(f));

oos.writeObject(array);

} **catch** (IOException e) {

e.printStackTrace();

} **finally** {

**try** {

**if** (oos != **null**)

oos.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

/\*\*

\* Метод десеріалізує дані з файлу і повертає у вигляді об'єкту типу ArrayList<String>

\* **@param** f об'єкт типу File, що містить шлях до файлу, в якому міститься серіалізований об'єкт

\* **@return** десеріалізований об'єкт типу ArrayList<String>

\*/

**public** ArrayList<String> deserialization (File f) {

ArrayList<String> array = **null**;

ObjectInputStream ois = **null**;

**try** {

ois = **new** ObjectInputStream(**new** FileInputStream(f));

array = (ArrayList<String>) ois.readObject();

} **catch** (FileNotFoundException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

} **finally** {

**try** {

**if** (ois != **null**)

ois.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

**return** array;

}

/\*\*

\* Метод зчитує інформацію з CSV-файлу за допомогою потоків вводу пакету java.nio

\* **@param** f CSV-файл, звідки зчитуються дані

\* **@return** масив елементів типу String, що містяться в класі ArrayList

\* **@throws** IOException

\*/

**public** ArrayList<String> nio\_read\_CSV (File f) **throws** CSVParseException, IOException{

ArrayList<String> array = **new** ArrayList<String>();

FileChannel in\_channel = **null**;

ByteBuffer buf;

**try** {

in\_channel = **new** FileInputStream(f).getChannel();

buf = ByteBuffer.*allocate*(*getBuf\_size*());

**while** ((in\_channel.read(buf)) != -1 ) {

buf.flip();

**for** (**int** i = 0; i < buf.limit(); i++)

array.add((**char**)buf.get() + "");

buf.clear();

}

} **catch**(Exception e) {

**throw** **new** CSVParseException();

} **finally** {

**if** (in\_channel != **null**)

in\_channel.close();

}

**return** array;

}

/\*\*

\* Метод записує дані з заданого масиву типу ArrayList<String> в файл за допомогою потоків виводу пакету java.nio

\* **@param** array масив даних, що записуватимуться в файл

\* **@param** f об'єкт типу File, що містить шлях до файлу, куди записуватимуться дані

\*/

**public** **void** nio\_write\_CSV (ArrayList<String> array, File f) {

FileChannel out\_channel = **null**;

ByteBuffer buf;

**try** {

out\_channel = **new** FileOutputStream(f).getChannel();

buf = ByteBuffer.*allocate*(*getBuf\_size*());

**for** (**int** i = 0; i < array.size(); i++) {

buf = ByteBuffer.*wrap*(array.get(i).getBytes(Charset.*forName*("utf-8")));

i++;

buf.flip();

out\_channel.write(buf);

buf.clear();

}

} **catch** (FileNotFoundException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

} **finally** {

**try** {

**if** (out\_channel != **null**)

out\_channel.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

/\*\*

\* **@return** the buf\_size

\*/

**public** **static** **int** getBuf\_size() {

**return** *buf\_size*;

}

/\*\*

\* **@param** buf\_size the buf\_size to set

\*/

**public** **static** **void** setBuf\_size(**int** buf\_size) {

CSVProcessor.*buf\_size* = buf\_size;

}

}  
  
**package** lab1;

**import** java.io.File;

**import** java.util.ArrayList;

**import** java.util.Scanner;

/\*\*

\* Клієнт.

\* **@author** Aruy

\*/

**public** **class** **TestIO** **implements** Runnable {

**private** **static** CSVProcessor *csv*;

**private** **static** ArrayList<String> *array*;

/\*\*

\* Виводить на екран дані об'єкту типу ArrayList<String>

\* **@param** array об'єкт типу ArrayList<String>

\*/

**public** **static** **void** show(ArrayList<String> array, String s, String path) {

System.*out*.println (" " + s + " \"" + path.toString() + "\" :");

**for** (**int** i = 0; i < array.size(); i++)

System.*out*.print(array.get(i));

System.*out*.println ();

}

/\*\*

\* Виводить на екран дані об'єкту типу String[][]

\* **@param** array об'єкт типу String[][]

\*/

**public** **static** **void** show\_mass(String[][] array) {

System.*out*.println (" Дані, що були перенесені з об'єкту типу ArrayList в String[][].");

**for** (**int** i = 0; i < array.length; i++) {

**for** (**int** j = 0; j < array[i].length; j++) {

System.*out*.print(array[i][j]);

}

}

}

**public** **static** **void** main(String[] args) **throws** Exception {

String ser\_path = "D:/serialized.dat";

File file = **null**;

File file\_ser = **new** File(ser\_path);

*csv* = **new** CSVProcessor(1024);

Scanner sc = **new** Scanner(System.*in*);

**byte** attempt = 3;

*array* = **null**;

Runnable r = **new** TestIO();

Thread t = **new** Thread(r);

t.join();

**if** (file\_ser.exists()) {

ArrayList<String> ser = *csv*.deserialization(file\_ser);

*show*(ser, "Дані десеріалізовані з файлу", ser\_path);

} **else** {

**do** {

**try** {

System.*out*.println (" Введіть шлях до CSV-файлу, звідки будуть зчитуватись дані.");

file = **new** File(sc.nextLine()/\*"D:/in.csv"\*/); // D:/in.csv

*array* = *csv*.nio\_read\_CSV(file);

} **catch** (CSVParseException e) {

attempt--;

**if** (attempt == 0) {

System.*out*.print ("У вас закінчились спроби!\n");

**throw** **new** Exception();

}

System.*out*.println (e);

System.*out*.println ("Помилка!!! Залишилось спроб: " + attempt + "\n");

}

} **while** (!file.exists()) ;

*csv*.serialization(*array*, file\_ser);

*show*(*array*, "Дані зчитані з файлу", file.getAbsolutePath());

t.start();

}

}

**public** **void** run() {

**try** {

*show\_mass*(*csv*.parse(*array*));

} **catch**(CSVParseException e) {

System.*out*.println (e);

}

}

}  
  
**package** lab1;

/\*\*

\* Клас виключення.

\* **@author** Aruy

\*/

**public** **class** **CSVParseException** **extends** Exception {

**public** CSVParseException() {

System.*err*.println ("Помилка CSVParseException");

}

}

***JavaDoc***

## C

[**CSVParseException**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVParseException.html) - Exception in [lab1](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\package-summary.html)

Клас виключення.

[**CSVParseException()**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVParseException.html#CSVParseException()) - Constructor for exception lab1.[CSVParseException](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVParseException.html)

[**CSVProcessor**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html) - Class in [lab1](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\package-summary.html)

У класі CSVProcessor реалізувані методи завантаження/вивантаження CSV-файлу в/з агрегований об’єкт класу ArrayList, в якому знаходяться рядки CSV-файлу.

[**CSVProcessor(int)**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html#CSVProcessor(int)) - Constructor for class lab1.[CSVProcessor](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html)

Конструктор

## D

[**deserialization(File)**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html#deserialization(java.io.File)) - Method in class lab1.[CSVProcessor](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html)

Метод десеріалізує дані з файлу і повертає у вигляді об'єкту типу ArrayList

## G

[**getBuf\_size()**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html#getBuf_size()) - Static method in class lab1.[CSVProcessor](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html)

## L

[lab1](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\package-summary.html) - package lab1

## M

[**main(String[])**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\TestIO.html#main(java.lang.String[])) - Static method in class lab1.[TestIO](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\TestIO.html)

## N

[**nio\_read\_CSV(File)**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html#nio_read_CSV(java.io.File)) - Method in class lab1.[CSVProcessor](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html)

Метод зчитує інформацію з CSV-файлу за допомогою потоків вводу пакету java.nio

[**nio\_write\_CSV(ArrayList<String>, File)**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html#nio_write_CSV(java.util.ArrayList, java.io.File)) - Method in class lab1.[CSVProcessor](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html)

Метод записує дані з заданого масиву типу ArrayList в файл за допомогою потоків виводу пакету java.nio

## P

[**parse(ArrayList<String>)**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html#parse(java.util.ArrayList)) - Method in class lab1.[CSVProcessor](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html)

Метод зчитує з об'єкту типу ArrayList дані і записує їх в об'єкт типу String[][].

## R

[**run()**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\TestIO.html#run()) - Method in class lab1.[TestIO](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\TestIO.html)

## S

[**serialization(ArrayList<String>, File)**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html#serialization(java.util.ArrayList, java.io.File)) - Method in class lab1.[CSVProcessor](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html)

Метод серіалізує дані об'єкту array у файл

[**setBuf\_size(int)**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html#setBuf_size(int)) - Static method in class lab1.[CSVProcessor](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\CSVProcessor.html)

[**show(ArrayList<String>, String, String)**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\TestIO.html#show(java.util.ArrayList, java.lang.String, java.lang.String)) - Static method in class lab1.[TestIO](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\TestIO.html)

Виводить на екран дані об'єкту типу ArrayList

[**show\_mass(String[][])**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\TestIO.html#show_mass(java.lang.String[][])) - Static method in class lab1.[TestIO](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\TestIO.html)

Виводить на екран дані об'єкту типу String[][]

## T

[**TestIO**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\TestIO.html) - Class in [lab1](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\package-summary.html)

Клієнт.

[**TestIO()**](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\TestIO.html#TestIO()) - Constructor for class lab1.[TestIO](file:///D:\--==--\Java\%D0%9B%D0%B0%D0%B1%D0%B8\2\JavaDoc\lab1\TestIO.html)

***Висновок***

*Виконавши цю лабораторну роботу, я отримав базові навички з використання мультипотокового середовища в Java. Поглиблено вивчив механізм виключень. Ознайомився з засобами Java для розбору текстових даних.*

*.*