Увод

в обектно ориентираното програмиране

Работа с файлове - Работа с текстови файлове

Съдържание

- Потоци
- Четене от файл
- Писане във файл
- Try / Catch
- Задачи

Потоци

Потоци

- Последователност от байтове, които се изпращат от едно приложение към друго, или от един компютър към друг, или от приложение към файл и обратно

- Имат подредба

Потоците

- Позволяват четене и писане на данни
- Са подредени
- Достъпът при тях е последователен
- Трябва да бъдат затваряни, за да не се получи повреждане на файла

Основни класове

Основните класове са InputStream & OutputStream. Те са абстрактни. Има конкретни реализации, които ги наследяват:

- BufferedInputStream BufferedOutputStream
- DataInputStream, DataOutputStream,
- Reader, Writer,
- BufferedReader, BufferedWriter,
- PrintWriter и PrintStream

Четене от текстов файл

Четене от текстов файл

Ще използваме Scanner:

```
// Link the File variable to a file on the computer

File file = new File("test.txt");

// Create a Scanner connected to a file and specify encoding

Scanner fileReader = new Scanner(file, "windows-1251");

// Read file here...

// Close the resource after you've finished using it

fileReader.close();
```

Четене от текстов файл

```
int lineNumber = 0;

// Read file
while (fileReader.hasNextLine()) {
    lineNumber++;

    System.out.printf("Line %d: %s%n", lineNumber, fileReader.nextLine());
}
```

Писане в текстов файл

Писане в текстов файл

PrintStream има същите методи като при писане на конзолата:

```
// Create a PrintStream instance
PrintStream fileWriter = new PrintStream("numbers.txt");
// Loop through the numbers from 1 to 20 and write them
for (int num = 1; num <= 20; num++) {
    fileWriter.println(num);
}
// Close the stream when you are done using it
fileWriter.close();
```

Важно

Не пропускайте да затворите потока след като приключите с използването му!

За затваряне използвайте метода close().

Try / Catch

Try / Catch

Когато един код има потенциала да хвърли грешка, трябва да се обгърне в try { } блок.

Той пречи на грешката да прекъсне програмата. Try трябва да бъде последван от catch {} блок. Той се изпълнява, АКО в try е хвърлена грешка. По желание след тях може да се сложи finally {} блок, той се изпълнява винаги.

Пример

```
try {
PrintStream fileWriter = new PrintStream("file.txt");
} catch (Exception e) {
System.println(“We had an error”);
} finally {
fileWriter.close();
}
```

Задачи

Задача

Напишете проста програма, която брои колко пъти се среща дума в даден текстов файл (за дума считаме всеки подниз от текста). В примера, нека текстът изглежда така:

This is our "Intro to Programming in Java" book.

In it you will learn the basics of Java programming. You

will find out how nice Java is.

Решение:

https://github.com/LillyMihaylova/AcademyDemos/blob/master/FilesDemo/src/WordCounter.java

Задача

Напишете програма, която чете списък от имена от един текстов файл, сортира ги по азбучен ред и ги запазва в друг файл. Имената да са с латински букви. На всеки ред от файла, където са записани имената, има точно по едно име. На всеки ред от файла с резултата също трябва да има само по едно име.

Решение:
https://github.com/LillyMihaylova/AcademyDemos/blob/master/FilesDemo/src/SortNames.java

Домашно

Задача

Напишете програма, която заменя най-често срещаната дума в текстов файл с *******, броя звездички отговаря на дължината на думата.

Задача

Прочетете това:

•http://www.introprogramming.info/intro-java-book/read-online/glava15-tekstovi-failove/