Practical 4 Name: Date:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Files | | **Supervisor Signature** | | **Completed** |
| Due date: 5 April 2017 | |
| **Assessment task** | **Comment** | | **Completed** | |
| *Practical preparation* |  | |  | |
| *Demonstrate code* |  | |  | |

# Aim

The aim of this practical is to design, write, test and debug programs in Java which use files and streams.

Reference material:

* **Lecture Slides and Notes** *4\_CS2\_Files.pdf*
* **Deitel, chapter 17** Files, Streams and Object Serialisation ( **chapter 17**)
* The source code for all the examples in the lecture notes can be found on the student drive, in the subject folder under *code examples*.

# Practical preparation

1. Study the lecture slides and look up the Java API <https://docs.oracle.com/javase/7/docs/api/> to answer the following questions:
   1. what type are the objects *System.in*, *System.out* and *System.err* respectively
   2. why do we need to write *System.out* and not just simply *out*
   3. what type of class is InputStream
   4. what type of method is the method *read( )*
   5. does the method *read( )* throw any exceptions, and if yes, what type (checked or unchecked)
   6. what does method *read()* return
   7. what is the recommended use for FileInputStream type of streams, as indicated in the API
2. Study the code in FileLength.java, compile and run and answer the following questions:
   1. what is *inputStream*
   2. what is *args*
   3. what is the value of *args.length* for the following :



* 1. what is the value of *args[2]*

* 1. which constructor of FileInputStream is used in the line inStream = new FileInputStream(args[0]);
  2. delete the following public static void main(String[] args) **~~throws IOException~~** {

and compile. What error do you get and why?

* 1. when does the line inStream.close(); get to be executed?
  2. if the line

InputStream inStream = null; changes to

FileInputStream inStream = null;  
will this still be correct: inStream = System.in;   
Why?

* 1. add a line of code to display the 5th byte read

1. Study the lecture slides and look up the Java API <https://docs.oracle.com/javase/7/docs/api/> to answer the following questions:
   1. what type of class is class *Reader*
   2. is it compulsory for subclasses of *Reader* to implement the method *read( )* ?
   3. Show the code to create an object of type *InputStreamReader* that uses the keyboard as input

Hint: use the constructor [InputStreamReader](https://docs.oracle.com/javase/7/docs/api/java/io/InputStreamReader.html#InputStreamReader%28java.io.InputStream%29)([InputStream](https://docs.oracle.com/javase/7/docs/api/java/io/InputStream.html) in)

* 1. Show the code to create an object of type *FileReader* that uses the file boo.txt as input?

1. Fill in the gaps in the file *CharStreams.java*, to use *InputStreamReader* and *FileReader* to read text from keyboard or a text file.
2. Study the lecture slides and look up the Java API <https://docs.oracle.com/javase/7/docs/api/> to answer the following questions:
   1. what type of class is class *Writer*
   2. what are the parent classes of the class *FileWriter*
   3. show with an example how you can use the method write(int c) and the   
      method write (String str, int off, int len) for an object of type *FileWriter*
   4. Show the code to create an object of type *FileWriter* that uses the file *boo.txt* as output?
3. Look up the Java API <https://docs.oracle.com/javase/7/docs/api/> , study the code in *DataStreams.java*, compile and run to observe behaviour. Answer the following questions:
   1. what are the parent classes of DataOutputStream and DataInputStream
   2. which DataOutputStream method would you use to write the items of the array below in a file:

float[ ] weights = { 79.5, 120, 57.8, 100.5, 48 };

* 1. which constructors are used in the line below:

out = new DataOutputStream(new FileOutputStream(dataFile));

# Practical procedure

1. Write a short application to test some of the methods of class File. Your code should take as an input argument on the command line a file or a directory name and it should display on the screen the name of the file, if it is a directory or a file, its size and its absolute path.
2. Write a program which encrypts a file. For encryption use the simple Cesar Cipher (substitution cipher) – this is a simple method that uses an encryption key (number) to shift each character (character + key = encrypted character). The user is supposed to input the encryption key and the name of the file to be encrypted. The program creates the encrypted file with the name *originalFileName*Encrypted
3. Write a program which writes in a file the values of the powers of 10 from 0 to 10 separated by a comma. Open the file in Excel.
4. Create an Excel spreadsheet with a number of floating point values. Save it as a CSV (Comma Separated Value) type file. Write a program which reads the numbers from the comma-separated file, finds their average and saves it at the end of the original file.
5. Modify the VehicleTest class so that it also has an array of vehicles. Save the Vehicle-type objects created in VehicleTest in the array. Save all the elements of the array into a file. For this, you need to create an ObjectOutputStream. You will also need to modify your source files, so that they allow serialization.
6. Open the file with an editor – what do you notice?
7. Add code to VehicleTest so that it reads back the objects saved in the file and places them into a new array of vehicles. Display them using a for loop.

# Topic Review

1. Fill in the gaps:
   1. Files that are created using byte-based streams are \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

* 1. Files that are created using \_\_\_\_\_\_\_\_\_\_\_\_\_-based streams are text files.
  2. When Java program begins executing there are 3 stream objects associated with devices. They are: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ and \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
  3. Class \_\_\_\_\_\_\_\_\_\_ is used to obtain information about files and directories.
  4. \_\_\_\_\_\_\_\_\_\_\_ and \_\_\_\_\_\_\_\_\_\_\_\_\_\_ are abstract classes for performing byte-based input and output.
  5. Any input or output of data needs to be done via a \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.
  6. To close a text input you need to use \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

1. What does the following line of code do:

Scanner input = new Scanner(System.in);

Is Scanner an IO class? Why do we need to use it?

1. Look up the interface DataInput. What is it used for? Which classes implement it?
2. Find the error in this block of code and show how to correct it:

ObjectOutputStream outputStream;

outputStream.writeInt(account);

outputStream.writeChars(company);

outputStream.writeDouble(amount);