Laboratory Sheet 8

This Lab Sheet contains material based on Lectures 1 – 16 (up to *13 November 2013*), and contains the submission information for Laboratory 8 (week 9, *18 – 22 November 2013*).

You are expected to begin work on laboratory sheets before your scheduled session. This will be necessary if you are to make best use of the presence of your tutor during the lab, and to make a satisfactory attempt at the submission exercise(s).

The deadline for submission of the lab exercise is 24 hours after the end of your scheduled laboratory session in week 9 *(18 –22 November 2013)*.

Of course you may submit work that is incorrect or incomplete. In order to stretch the stronger members of the class, some of the laboratory exercises are quite challenging, and you should not be too discouraged if you cannot complete all of them.

Aims and objectives

- to gain experience of file I/O in Java
- to practice String and character manipulation
- to reinforce array concepts

Set up

When you download Laboratory8.zip from moodle, please unzip this file. You will obtain a folder Laboratory8, containing a subfolder entitled Submission8_1. Remember that for this Laboratory you will have to switch your Eclipse workspace to the Laboratory8 folder.

In the folder Submission8_1 will be the following files:

- LongestWord.java which defines a `main` method
- words.txt which is a plain text file containing a list of words

In Eclipse, you should create a new project entitled Submission8_1; the given files will automatically become part of this project.

You will need to add code to the `main` method of LongestWord.java for this exercise.

Submission material

Preparatory work for this programming exercise, prior to your scheduled lab session, is expected and essential to enable you to submit a satisfactory attempt.

Submission exercise 8

You are given a text file words.txt that contains one word per line. Your task is to find, for each letter of the alphabet, one of the longest words that begins with that letter. Your main method should take a single `String` argument that specifies the name of the input text file. See
http://help.eclipse.org/kepler/index.jsp?topic=%2Forg.eclipse.jdt.doc.user%2Ftasks%2Ftasks-java-local-configuration.htm for how to specify arguments to `main`.

Hints

1.
Use a `String` array, `longestWords`, with 26 elements.
`longestWords[0]` stores the longest word beginning with 'a'
`longestWords[1]` stores the longest word beginning with 'b'
...
`longestWords[25]` stores the longest word beginning with 'z'

2.
Use a `FileReader` wrapped in a `BufferedReader` to read the input text file line-by-line into memory.

3.
Use appropriate `String` library methods to access the first character in each word, and to calculate the length of each word.

4.
Your final output should be printed to the console, and should look like this:

```
a: anthropomorphologically
b: blepharosphincterectomy
c: cholecystenterorrhaphy
d: dacryocystoblennorrhea
e: epididymodeferentectomy
f: formaldehydesulphoxylate
g: gastroenteroanastomosis
h: hematospectrophotometer
i: indistinguishableness
j: jurisprudentialist
k: keratoconjunctivitis
l: laparocolpohysterotomy
m: macracanthrorhynchiasis
n: naphthylaminesulphonic
o: omnirepresentativeness
p: pathologicopsychological
q: quadratomandibular
r: reticulatocoalescent
s: scientificophilosophical
t: tetraiodophenolphthalein
u: ureterocystanastomosis
v: vagoglossopharyngeal
w: weatherproofness
x: xanthocreatinine
y: yohimbinization
z: zoologicoarchaeologist
```

5. What are the tutors looking for?

- `try`-with-resources statement, or try/finally to close input streams.
- `catch` blocks for specific `Exception` subclasses.
- sensible use of Java standard library methods.
- elegant code with appropriate control-flow constructs.

Extra Challenge (not worth any extra credit)

If you have enough time, try to setup a second `String` array called `mostCharacters`, which stores, for each letter of the alphabet, a word that has the most occurrences of that letter. My console output for this second part looks like:

```
a: astragalocalcaneal
b: beblubber
c: chlorococcaceae
d: disdodecahedroid
e: electrotelethermometer
f: giffgaff
g: cuggermugger
h: choledochorrhaphy
i: impossibilification
j: ajaja
k: akiskemikinik
l: allochlorophyll
m: dynamometamorphism
n: nonannouncement
o: choledochoduodenostomy
p: aplopappus
q: equivoque
r: archcorrupter
s: possessionlessness
t: anticonstitutionalist
u: untumultuous
v: overconservative
w: bowwow
x: adnexopexy
y: dacryocystosyringotomy
z: zizz
```

Submission

You should submit your work before the deadline no matter whether the programs are fully working or not.

When you are ready to submit, go to the JP2 moodle site. Click on Laboratory 8 Submission. Click 'Add Submission'. Open Windows Explorer and browse to the folder that contains your Java source code ...\Laboratory8\Submission8_1\ and drag *only* the **single** Java file LongestWord.java into the drag-n-drop area on the moodle submission page. **Your markers only want to read your java file, not your class file.** Then click the blue save changes button. Check the.java file is uploaded to the system. Then click submit assignment and fill in the non-plagiarism declaration. Your tutor will inspect your file and return feedback to you via moodle.

Gaining the credits for JP2

Recall that the credit criteria for JP2 include obtaining at least 7 ticks for lab assignments. To obtain a tick you must attend the lab and submit the assignment.