Assignment 3: File I/O and Array Problems

Weight: 15% of your final grade

You will have the background to start this assignment after completing Unit 6.  
You will have the background to submit this assignment after completing Unit 8.

**Instructions**

Refer to the [*Study Guide*](https://comp.athabascau.ca/268/r12/unit01.html) as needed to ensure you have the understanding of the course material required to complete this assignment.

Refer to the [*Instructor’s Notebook*](https://landing.athabascau.ca/pages/view/4424610/the-instructors-notebook) for the most up-to-date software (code editors and compilers) for Java programming.

Submit your assignment using the drop box on the course home page. Follow exactly the format for submission outlined in [Assignment Requirements](https://scis.lms.athabascau.ca/mod/page/view.php?id=34783).

All assignment submissions must include all of the following deliverables:

* Java source code for a complete, working program that fulfills the requirements of the following five (5) problems;
* additional files as described in the requirements for each problem;
* full documentation and a test plan for your primary Java source code file as described in the Assignment Requirements; and
* reflections on the assignment (Personal Notebook). Be sure to document all your code changes and observations in your Personal Notebook, excerpts of which will accompany your assignment source code submission.

A significant number of marks is attributed to your documentation and test plans. Be sure to include all deliverables with your assignment.

**The Problems**

**Problem 1**

Write a program that opens a file and counts the whitespace-separated words in that file. You may obtain the name of the input file either from the command line or via prompt and user input. Be sure to clearly document the method chosen.

You may test your program using any number of text files; however, you **must** be sure to test your program using the Sample Text File provided at the end of this assignment (excerpt.txt).

**Problem 2**

Write a program that opens a (text) file and displays the contents of that file one line at a time. You must obtain the name of the input file for this program via prompt and user input, not via command line arguments.

After a line has been displayed, the program should wait for the user to press the <Enter> key before displaying the next line in the file.

As with Problem 1, you may test your program on any number of text files;

however, you **must** be sure to test your program using the Sample Text File provided at the end of this assignment (excerpt.txt).

**Problem 3**

Write a program in which you create a class called *TextFileReader*. The TextFileReader class must contain an instance variable consisting of an array of  Strings 100 elements long.

TextFileReader should have two constructors: a default constructor and a constructor that takes a String argument, which represents the name of a text file to be opened and read into the array of Strings. The first (default) constructor does not really do anything. If the second constructor is used, it will open the file and read the contents of the file into the array of Strings as follows: fill the array by having your program read one line of the file into each string until you have filled the array. Once the array is full, you may stop reading the input file.

Add a member function contents(), which will convert the array of Strings into a single StringBuffer, which is then returned to the calling program so that it may be displayed.

Add a second member function display(), which prints the array to standard output using the format *line #: <string>*, where *#* is the actual line number and *<string>* is the stored string. [You can use the array counter for the *#* value.] Do not forget that line numbers for the output should start with 1, not 0.

Now, create a second class TextFileReaderDemo, which contains a main() method used to test TextFileReader. Create an instance of TextFileReader using the second constructor, and then call contents() to obtain the file’s text. Display the file’s contents to standard output. Finally, call display() to output the file’s text together with line numbers, as stated in the display() requirements.

TextFileReaderDemo should use command-line arguments to obtain the text file name supplied to the TextFileReader constructor.

As with Problems 1 and 2, you may test your program on any number of text files; however, you **must** be sure to test your program using the Sample Text File provided at the end of this assignment (excerpt.txt).

**Problem 4**

Write a program that creates three floating point **arrays**. Fill the first two arrays with 25 floating point numbers using loop structures, as follows: fill the first array with the loop counter value and fill the second array with the loop counter value squared. Finally, write a loop that adds the corresponding elements in the first two arrays and puts the result in the corresponding element of the third array. Display all three arrays using the format *for counter; element + element = element*.

**Problem 5**

Define a class called *Book*. The Book class should store attributes such as the title, ISBN, author, edition, publisher, and year of publication. These attributes must be private.

Provide public get/set methods in this class to access these attributes. Test your Book class by creating several books and displaying the attributes. You may start with the Book program you wrote for Assignment 2, Problem 2.

Define a class called *Bookshelf*, which contains only a main() method. The Bookshelf class must create a dozen (12) Book objects with distinct attributes, and it must store them in an ArrayList of Books.

The Bookshelf class must then list all the attributes of all books in the ArrayList in the order they were entered into the ArrayList. Create a sort function for Bookshelf, which will sort books in the ArrayList in ascending order by name, and then by year of publication.

**Hint**: You will need to define a comparator class that takes two Book objects as parameters of the compareTo method. This method should make a two-step comparison and return a Boolean value. The first comparison should compare the book names. If the names are the same, the second comparison should compare the years of publication.

Add code to main() to display the sorted Book list after the first output list—that is, Books in the order they were entered into the ArrayList.

**Sample Text File for Problems 1, 2 and 3 (excerpt.txt**[**[1]**](https://scis.lms.athabascau.ca/mod/assign/view.php?id=34781#foot1)**):**

There was nothing so VERY remarkable in that; nor did Alice

think it so VERY much out of the way to hear the Rabbit say to

itself, `Oh dear!  Oh dear!  I shall be late!'  (when she thought

it over afterwards, it occurred to her that she ought to have

wondered at this, but at the time it all seemed quite natural);

but when the Rabbit actually TOOK A WATCH OUT OF ITS WAISTCOAT-

POCKET, and looked at it, and then hurried on, Alice started to

her feet, for it flashed across her mind that she had never

before seen a rabbit with either a waistcoat-pocket, or a watch to

take out of it, and burning with curiosity, she ran across the

field after it, and fortunately was just in time to see it pop

down a large rabbit-hole under the hedge.

In another moment down went Alice after it, never once

considering how in the world she was to get out again.

The rabbit-hole went straight on like a tunnel for some way,

and then dipped suddenly down, so suddenly that Alice had not a

moment to think about stopping herself before she found herself

falling down a very deep well.

Either the well was very deep, or she fell very slowly, for she

had plenty of time as she went down to look about her and to

wonder what was going to happen next.  First, she tried to look

down and make out what she was coming to, but it was too dark to

see anything; then she looked at the sides of the well, and

noticed that they were filled with cupboards and book-shelves;

here and there she saw maps and pictures hung upon pegs.  She

took down a jar from one of the shelves as she passed; it was

labelled `ORANGE MARMALADE', but to her great disappointment it

was empty:  she did not like to drop the jar for fear of killing

somebody, so managed to put it into one of the cupboards as she

fell past it.

**Note**: After you have completed this assignment, do not discuss its contents on any forum or post your solutions anywhere on the Internet. Doing so will be considered cheating and will be dealt with in accordance with the [Student Academic Misconduct Policy](http://calendar.athabascau.ca/undergrad/current/student-code/index.php#acad_misconduct).

[[1]](https://scis.lms.athabascau.ca/mod/assign/view.php?id=34781#fn1) This excerpt is adapted from *Alice’s Adventures in Wonderland* by Lewis Carroll, available from Project Gutenberg. <http://www.gutenberg.org/files/11/11-h/11-h.htm>