Javadoc – Java Documentation Standards

I. General Structure of Documentation:

Formatting your java files

Comments within your source code:

All files should follow the javadoc documentation formatting standard. The formatting standard is as follows:

- The first line is indented to line up with the code below the comment, and starts with the begin-comment symbol (/**) followed by a return.
- Subsequent lines start with an asterisk * . They are indented an additional space so the asterisks line up. A space separates the asterisk from the descriptive text or tag that follows it.
- Insert a blank comment line between the description and the list of tags, as shown
- Insert additional blank lines to create "blocks" of related tags (discussed in greater detail below).
- The last line begins with the end-comment symbol (*/) followed by a return. Note that the end-comment symbol contains only a single asterisk (*).

Example:

```
/**
 * This is the description part of a doc comment
 *
 * @tag    Comment for the tag
 */
```

Break any doc-comment lines exceeding 80 characters in length with
 If you have more than one paragraph in the doc comment, separate the paragraphs with a <p> paragraph tag.

- ***Main / driver file (file that contains class with main method) - (placed directly above the class declaration):***

```
[declaration part: where you import other classes]
/**
* This is a detailed description of the class.
*  Note that ordinary HTML tags can be embedded in your comments
*
* <p><b>
* Extra Credit:
```

```
* </b><pre>
* Whether or not you attempted the extra credit (if applicable)
* </pre><b>
*
* History:
* </b><pre>
* List of changes, updates, fixes revisions which includes date
* </pre>
*
* @author       (Your Name)
* @see          (Any class whose documentation aids in understanding the
*               implementation of your class)
*
*               Proper citation of code borrowed from anther source
*                       - for web sites put the URL, the authors name,
*                       and description of what was borrowed
*                       - for a book, paper, etc put the Title, the author,
*                       publisher, pages, and description of what
*                       was borrowed
*
*               NOTE: - Each citation must be contained within quotation marks
*                       - Break up each @see with <br> after the ending "
*                       - If you did not borrow code then your @see should
*                               simply be
*                       @see "No Borrowed Code"
*/
```

[declaration part: here declare your class] - ***NOTE: the declaration of the class must come immediately after the comments.***

Now include the main method (and any other methods that the class may contain)

- ***All Methods (except the main method) will contain the following comments placed right above the method declaration***

```
/**
* This first sentence will appear in the method summary.  This method
description should be a
* detailed description of what the method does.  You should have more than
one sentence, just
* know the first sentence appears in the method summary but all the sentences
appear in the
* method detail
*
* @param        List the name (not the type) and then a description of the
*                       data
*               - if the method takes more than one parameter you will need
*                       additional @param on the next line
*               - if the method takes no parameters, do not use a @param tag
*
* @return       List the type and then a description of the data
*               - if the method does not return a value: DO NOT USE THE TAG
*               - if the method is a constructor, DO NOT use the tag at all
*               since constructors never return a value

* @exception    List the exception class name and then a description of what
*               would cause an exception to be thrown.
*               - if no exception is thrown list none - No exception thrown
```

```
* @see          "Any class whose documentation aids in understanding the
*               implementation of your class"
*
*               Proper citation of code borrowed from anther source
*                       - for web sites put the URL, the authors name,
*                       and description of what was borrowed
*                       - for a book, paper, etc put the Title, the author,
*                       publisher, pages, and description of what
*                       was borrowed
*
*               NOTE: - Each citation must be contained within quotation marks
*                       - Break up each @see with <br> after the ending "
*                       - If you did not borrow code then your @see should
*                               simply be
*                       @see "No Borrowed Code"
*/
```

The template for methods is:

```
/**
 * This first sentence will appear in the method summary.  This method description
should be a
 * detailed description of what the method does.  You should have more than one
sentence, just
 * know the first sentence appears in the method summary but all the sentences appear
in the
 * method detail
 *
 * @param       name - name of variable in parameter list
 * @return       void - Does not return a value
 * @exception    none - No exception thrown
 * @see          give credit to anyone/anything that helped you
 */
```

Samples of commented Java source code:

```
/**
 * This file has been stripped down for this example, but it is a very good file
 * to use for this example.  Notice how this comment must appear right above the
 * class definition.  If you place the import after this comment, right before
 * the class definition, it will not work.
 *
 * <p><b>
 * Extra Credit:
 * </b><pre>
 * No Extra Credit was Attempted
 * </pre><b>
 *
 * History:
 * </b><pre>
 * 1/3/2002 Started writing the Documentation Standards
 * 1/6/2002 Posted current standards
 * 1/9/2002 Copied this file from old final exam and wrote the documentation
 * </pre>
 *
 * @author  Ada Lovelace
 * @see     "java.lang.Math"<br>
 *               "StudentDriver.java - written by Ada Lovelace for CSCD210"
 */
public class StudentDriver
{
        public static void main(String[] args)
        {
                //  Your cool code goes in here...

        }//end main method

}//end class
```

II. SPECIFIC DOCUMENTATION RULES

1. **Variables/constants**
   You are only required to comment the purpose and use of variables when their usage is not obvious from the program context. This is a gray area, so just use your best judgment. You will never, however, be scored down for including reasonable variable comments.
2. **Method Descriptions - (See - Your Java Files)**.
   All methods other than the main method will have a description The description should be set apart from the code using the javadoc standards outlined above. If any well-known algorithm is used in the method, this should be mentioned, i.e., Bubble-sort, Picard's method, etc.

3. **Whitespace**
   You should use blank lines to separate blocks of code from each other to enhance code readability, for instance between the program header and the declaration section, between the declaration section and the body of the program, between methods, or even between code structures such as loops, to distinguish them from the rest of your code.
4. **Citations**
   Any code that you use that was not created by you needs to be cited as a professional courtesy. Properly citing is a ***MUST*** .
   - If the code is obtained from the internet site the complete link, the author (if none, state so) and give a description of what was borrowed.
   - If the code came from another person (via help), state the person's name and what they do (i.e. tutor, fellow classmate, etc.), and how they helped you.
   - If the code was given in class, cite the day it was given in class, and a description of what was borrowed.
   - If the code was obtained from a book, include the book title, author, publisher, and pages where the code is found as well as a exact description of what is being used.
5. **Capitalization/Indentation**
   Java is case sensitive! All Java keywords are lower case. You may use any reasonable names for methods and variables as long as you follow the following requirements:
   - Class names will begin with a capital letter. Multi-word class names should be separated with capital letters.
   - Method names will begin with a lower case letter. Multi-word method names should be separated with capital letters.
   - Constants will be completely capitalized – use an underscore to separate multi-word constants
   - All variables will begin with a lowercase letter. Multi-word variable names should be separated with capital letters.

     Examples:

```
// total income - private instance data
private double totalIncome;
public void myMethod()

{
   String  temp, anotherTemp; // local instance within a method

} // end of method
```

- Indentation should be used to enhance code readability. If code belongs to a larger construct, indent it to make that clear. Most syntax highlighting editors will take care of this for you, but use your own discretion.
