Style Requirements for all CS1180 Projects and Labs

Naming your project:

- Name your project folder *yourLastName_project#* and name your labs *yourGroupNumber_lab#* (where # is the current project or lab number). Note that it can cause problems if you rename a project folder after the project has been created. To do so correctly, you will need to refactor. This will be demonstrated in lab.

Comments. You must follow steps 1-4 for all projects. Because lab time is limited, only step 1 is required for labs unless stated otherwise within the specific lab requirements document.

1. For all projects, put a comment block with the following information at the top of your code file:

```
/**
* @author [your first and last name]
* CS1180, Summer 2021
* the name of the assignment (e.g. Lab 1, Project 3)
*/
```

2. Use a javadoc-style comment before the class name to describe the purpose of the program:

```
/**
 * This program demonstrates the use of the NetBeans IDE.
 * The program will prompt for input, do numerical manipulation,
 * and output results.
*/
```

3. Use a javadoc-style comment before every method to describe the purpose of the method, method parameters, and the return value. Note: If the parameters have constraints (such as must be greater than 0 or the condition shown for @param high in the example below), be sure to state those in the comment for that parameter.

   Example:

```
/**
 * Prompts the user to enter a value in a given range until the user
 * provides valid input
 * @param low - the low end of the input range
 * @param high - the high end of the input range; this value is greater
 *               than the value in @param low
 * @return the value entered by the user.
 */
 public static int getInputWithinRange(int low, int high)
 {
      //method body
  }
```

4. Use single-line comments before each main step within a method (including main).

Naming Conventions:

- The first word of a variable or method name should be lowercase; if a variable or method name consists of several words use camelCase.
- All letters in the name of a constant should be capitalized; use the underscore to separate words (example, STATE_TAX_RATE).
- Capitalize the first letter of each word in a class name ~~(CamelCase)~~. camelCase
- Identifier names must be meaningful; class names and variable names should reflect the purpose of object or data; a method name should be camelCase and should contain a verb.

Indenting code:

- Be sure all code within a block (usually this means within a matched set of braces) is indented one tab stop (tab stops should be 3-4 spaces; the default tabs in NetBeans are fine)
- Comments should be indented the same amount as the code they describe
- Use a blank line between sections of code (usually this means above a comment)
- Do NOT double-space code within a section

Block Styles

A block is a group of statements surrounded by braces. The two allowed styles for the braces are next-line (shown in the first example below) and end-of-line (shown in the second example). You may use either style as long as you use it consistently within a program.

- In the next-line style both the beginning and ending braces are on separate lines, and are aligned with the "heading" for that group of statements. All code within the braces is indented one more tab stop than the heading and braces.

```
public class Test
{
    public static void main(String[] args)
    {
        System.out.println("This block uses next-line style.");
    }
}
```

- For end-of-line style, the opening brace follows the heading rather than being on a separate line. The closing brace is on a separate line, and indenting is the same as in next-line style.

```
public class Test {
    public static void main(String[] args) {
       System.out.println("This block uses end-of-line style.");
    }
}
```

Next-line style is the preferred style in this class for if statements and for loops.

Additional examples follow on the next page.

Next-line style:

```
if (quantity > 10)
{
    System.out.println("The discount is 20%");
    finalPrice = price * .80;
}
else if (quantity > 5)
{
    System.out.println("The discount is 10%");
    finalPrice = price * .90;
}
else
{
    System.out.println("No discount.");
    finalPrice = price;
}
```

```
int count = 1;
while (count < 10)
{
   System.out.println("Hello!");
   count++;
 }
```

End-of-line style:

```
if (quantity > 10){
    System.out.println("The discount is 20%");
    finalPrice = price * .80;
}
else if (quantity > 5){
    System.out.println("The discount is 10%");
    finalPrice = price * .90;
}
else {
    System.out.println("No discount.");
    finalPrice = price;
}
```

```
int count = 1;
while (count < 10)  {
   System.out.println("Hello!");
   count++;
 }
```