Programming Style and Documentation

It is not unusual for employers to establish mandatory guidelines for programming style and documentation by their software development staff. The purpose of such rules is to make software easier to maintain and reduce debugging time. The following (rather relaxed) guidelines apply for all work you produce in this class.

**Style**

1. Variables are only accessible by modules that need them. If a variable is only needed by a single module, declare it within that module.

2. Methods have only one entry point and one exit. This applies to the bodies of for and while loops as well. **Break** and **continue** are only used in a switch structure.

3. The body of any block is indented one tab stop from the header. This applies to class, method, for, while, and if blocks. (A block is the collection of statements between braces).

4. Opening and closing braces for blocks should be used consistently. Many programmers prefer to always place both braces in line with the header of the block so that it is clear they are both there.

5. Write one statement per line.

6. If a statement is too long to fit on one line when the listing is printed, break it and indent the continuations. Do not allow lines to violate the indentation for the block they are in.

7. Identifiers clearly describe the method, class, or variable they represent. Indices may be named i, j, or k if no reasonable name such as **day** or **month** applies. Identifiers for methods should be verb or verb-object in form, and describe the function of the method. Variables should usually be noun or adjective-noun in form, although boolean variables should describe a condition.

   It is wise to avoid identifiers that are too similar (weight and weight1, for instance). It is easy to use the wrong name when writing code, and hard to find the source of the error when debugging.

   Sample method identifiers: processData, getStudentName, computeSalary.

   Sample variable identifiers: studentName, foundIndex, validInput.

8. Major blocks of code are set off by blank lines before and after.

9. Any block of code (other than a switch) has only one exit. This applies to programs, methods, and loops. No method should have more than one return statement. The exit for a program should be in the main method.

**Documentation**

A block of documentation precedes the class header. This block describes the general purpose of the program and what inputs and outputs it has. The author’s name and the date written appear at the end, followed by a revision history.

Each method is preceded by a block comment that identifies the purpose of the method, what arguments it receives, and what information (if any) it returns. If the operation of the method is subtle, a brief description of the algorithm is included. Some programmers also describe what variables will be used, though this is not required.

Cryptic or complicated statements are explicated by line comments. This includes variables that defy clear naming. These comments should not crowd java code; they need to be clearly visible.
This is not required, but using a line comment to identify what a closing brace ends can be a great time-saver.

Examples
Use of indentation and braces:

```java
while ( x < 30 )
{
    if ( y == z )
    {
        System.out.println ("Hurrah!");
    }
    x = x + 2;
}
```

Proper continuation of a line:

```java
inputVal = JOptionPane.showInputDialog ("Please enter the full name of the next contestant");
```

Improper ways to handle long lines. The following chunks of code are BAD.

1) just letting the text run off the page:

```java
// this is just a brief little comment to explain the purpose
```

2) breaking the line but letting the continuation break the form of the code by violating the indentation of the current block:

```java
while ( x < 30 )
{
    y = Integer.parseInt ( JOptionPane.showInputDialog ("Enter the next number");

    if ( y == z )
    {
        System.out.println ("Hurrah!");
    }
    x = x + 2;
}
```

Hint—to reduce the possibility of having lines that wrap or run off the page without your being aware of it, use gEdit’s preferences to display a right margin on the screen. Your margin should be between column 74 and 80. You can also disable text wrapping. (Both of these settings are under the “View” tab.)