**CS 139 Program Style Evaluation Chart**

1 point off each occurrence up to maximum of 2 per line. Maximum 5 points off for each section.

|  |
| --- |
| A. Names |
|  | 1. Names should be descriptive and readable |  |
|  | 2. Multi word names should use underscore or capital letter (camel case) |  |
|  | 3. Variable and method names should begin with a lowercase letter. |  |
|  | 4. Class names should begin with a capital letter and use title case. |  |
|  | 5. Constant names should be all caps with an underscore separator. |  |
| B. Declarations |
|  | 1. Constants should be named and initialized at the top of the method or class |  |
|  | 2. All variables should be declared directly after the constants (white space sep ok) |  |
|  | 3. There must be a line of white space directly after the variable declarations. |  |
|  | 4. It is NOT permissible to initialize a variable as you are declaring it. |  |
|  | 5. Each declaration should be on its own line. Comment to the right if not clear |  |
| C. Indentation |
|  | 1. Subsections of code should be indented a consistent 3 spaces |  |
|  | 2. Statements that are too long should be indented 3 spaces for 2nd + lines |  |
|  | 3. Blocks of code should be surrounded by curly braces lined up directly under the first character of the header |  |
| D. Literals and constants |
|  | 1. Numeric literals must be of the correct type for the context in which they are used. |  |
|  | 2. Constants should be used for meaningful values. |  |
| E. Operators |
|  | 1. Binary operators should be separated from their operands by a single space |  |
|  | 2. Unary operators should NOT be separated by a space. |  |
| F. Structure |
|  | 1. Constants in a method or class should come first |  |
|  | 2. One line of white space should follow the declaration of variables. |  |
|  | 3. Use white space to separate segments of code.  |  |
|  | 4. Lines should be kept to a short length (<-80 characters) |  |
|  | 5. Methods should be no longer than 25 lines in length. |  |
|  | 6. If a method returns a value it should have a single return statement. (one entry one exit) |  |
|  | 7. Break statements are not permitted except for their use in a switch. |  |
|  | 8. You must NOT have any unused variables, constants, or do nothing lines of code (a = a;) or empty if/else blocks |  |
|  | 9. You must NOT have any empty if/else blocks. |  |
| G. Class structure |  |
|  | 1. In a class declaration, constructors should precede all other methods. (Later Pas) |  |
|  | 2. Methods should be defined in order by visibility modifier, then alphabetically (Later Pas) |  |
|  | 3. All class variables should be private unless there is a documented reason for public (Later Pas) |  |
|  | 4. All class constants should be public unless there is a documented reason for private (Later Pas)  |  |
|  | 5. All visibility modifiers should be explicitly stated (Later Pas) |  |
| H. Comments |
|  | 1. Should use normal English spelling and grammar. Phrases okay. |  |
|  | 2. They must come **before** the code that they are describing. |  |
|  | 3. Inline comments should only describe major structures or steps in a method |  |
|  | 4. A line of white space should precede every block or full line inline comment |  |
| I. Class Comment |
|  | 1. Every class must contain a javadoc style comment in the correct format |  |
|  | 2. It must include an @author tag  |  |
|  | 3. It must include an @version tag |  |
|  | 4. It must describe the purpose of the class |  |
| J. Acknowledgements |
|  | 1. All PAs must have an Acknowledgement section referencing help (-5 if missing) |  |
|  | 2. This section must come at the top of each file just beneath the class comment. |  |
| K. Method Comment |
|  | 1. Every method must have a javadoc style comment in the correct format (-2 each if missing) |  |
|  | 2. It must include a description of the purpose of the class |  |
|  | 3. It must have an @param tag for EACH parameter and must describe each |  |
|  | 4. It must have an @return tag describing the return value (if the method returns) |  |
|  | 5. It must NOT include an @param if no parameters or @return if no return value |  |
| L. Miscellaneous |
|  | This category will be different for each PA.  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**CS 139 Program Style Evaluation Chart**

1 point off each occurrence up to maximum of 2 per line. Maximum 5 points off for each section.

|  |
| --- |
| A. Names |
|  | 1. Names should be descriptive and readable |  |
|  | 2. Multi word names should use underscore or capital letter (camel case) |  |
|  | 3. Variable and method names should begin with a lowercase letter. |  |
|  | 4. Class names should begin with a capital letter and use title case. |  |
|  | 5. Constant names should be all caps with an underscore separator. |  |
| B. Declarations |
|  | 1. Constants should be named and initialized at the top of the method or class |  |
|  | 2. All variables should be declared directly after the constants (white space sep ok) |  |
|  | 3. There must be a line of white space directly after the variable declarations. |  |
|  | 4. It is NOT permissible to initialize a variable as you are declaring it. |  |
|  | 5. Each declaration should be on its own line. Comment to the right if not clear |  |
| C. Indentation |
|  | 1. Subsections of code should be indented a consistent 3 spaces |  |
|  | 2. Statements that are too long should be indented 3 spaces for 2nd + lines |  |
|  | 3. Blocks of code should be surrounded by curly braces lined up directly under the first character of the header |  |
| D. Literals and constants |
|  | 1. Numeric literals must be of the correct type for the context in which they are used. |  |
|  | 2. Constants should be used for meaningful values. |  |
| E. Operators |
|  | 1. Binary operators should be separated from their operands by a single space |  |
|  | 2. Unary operators should NOT be separated by a space. |  |
| F. Structure |
|  | 1. Constants in a method or class should come first |  |
|  | 2. One line of white space should follow the declaration of variables. |  |
|  | 3. Use white space to separate segments of code.  |  |
|  | 4. Lines should be kept to a short length (<-80 characters) |  |
|  | 5. Methods should be no longer than 25 lines in length. |  |
|  | 6. If a method returns a value it should have a single return statement. (one entry one exit) |  |
|  | 7. Break statements are not permitted except for their use in a switch. |  |
|  | 8. You must NOT have any unused variables, constants, or do nothing lines of code (a = a;) or empty if/else blocks |  |
|  | 9. You must NOT have any empty if/else blocks. |  |
| G. Class structure |  |
|  | 1. In a class declaration, constructors should precede all other methods. (Later Pas) |  |
|  | 2. Methods should be defined in order by visibility modifier, then alphabetically (Later Pas) |  |
|  | 3. All class variables should be private unless there is a documented reason for public (Later Pas) |  |
|  | 4. All class constants should be public unless there is a documented reason for private (Later Pas)  |  |
|  | 5. All visibility modifiers should be explicitly stated (Later Pas) |  |
| H. Comments |
|  | 1. Should use normal English spelling and grammar. Phrases okay. |  |
|  | 2. They must come **before** the code that they are describing. |  |
|  | 3. Inline comments should only describe major structures or steps in a method |  |
|  | 4. A line of white space should precede every block or full line inline comment |  |
| I. Class Comment |
|  | 1. Every class must contain a javadoc style comment in the correct format |  |
|  | 2. It must include an @author tag  |  |
|  | 3. It must include an @version tag |  |
|  | 4. It must describe the purpose of the class |  |
| J. Acknowledgements |
|  | 1. All PAs must have an Acknowledgement section referencing help (-5 if missing) |  |
|  | 2. This section must come at the top of each file just beneath the class comment. |  |
| K. Method Comment |
|  | 1. Every method must have a javadoc style comment in the correct format (-2 each if missing) |  |
|  | 2. It must include a description of the purpose of the class |  |
|  | 3. It must have an @param tag for EACH parameter and must describe each |  |
|  | 4. It must have an @return tag describing the return value (if the method returns) |  |
|  | 5. It must NOT include an @param if no parameters or @return if no return value |  |
| L. Miscellaneous |
|  | This category will be different for each PA.  |  |
|  |  |  |
|  |  |  |
|  |  |  |