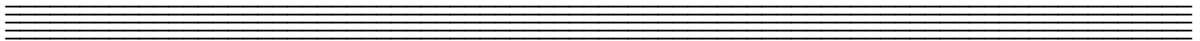


Assignment Kit for Coding Standard



Personal Software Process for Engineers: Part I

The Software Engineering Institute (SEI)
is a federally funded research and development center
sponsored by the U.S. Department of Defense and
operated by Carnegie Mellon University.

This material is approved for public release.
Distribution limited by the Software Engineering Institute to attendees.

Personal Software Process for Engineers: Part I

Assignment Kit for the Coding Standard

Overview

Overview This assignment kit covers the following topics.

Section	See Page
Prerequisites	2
Objectives	2
Coding standard requirements	3
Example coding standard	4
Evaluation criteria and suggestions	7
Coding standard template	8

Prerequisites Prerequisites

- Read Chapter 4
- Complete Size Counting Standard

Objectives The objectives of the coding standard are to

- establish a consistent set of coding practices
- provide criteria for judging the quality of the code that you produce
- facilitate size counting by ensuring your programs are written so they can be readily counted
- for LOC counting, require that there be a separate physical line for each logical line of code

Coding standard requirements

Coding standard requirements

Produce, document, and submit a completed coding standard that calls for quality coding practices.

For LOC counting, ensure that a separate physical source line is used for each logical line of code.

Submit the coding standard with your program 2 assignment package.

Example coding Standard

Coding standard example

Pages 5 and 6 of this workbook contain an example C++ coding standard.

Notes about the example

- Since it is an example, tailor it to meet your personal needs.
 - If you have an existing organizational standard, consider using it for the PSP exercises.
-

Continued on next page

Example C++ Coding Standard

Purpose	To guide implementation of C++ programs
Program Headers	Begin all programs with a descriptive header.
Header Format	<pre> /***** /* Program Assignment: the program number */ /* Name: your name */ /* Date: the date you started developing the program */ /* Description: a short description of the program and what it does */ *****/ </pre>
Listing Contents	Provide a summary of the listing contents
Contents Example	<pre> /***** /* Listing Contents: */ /* Reuse instructions */ /* Modification instructions */ /* Compilation instructions */ /* Includes */ /* Class declarations: */ /* CData */ /* ASet */ /* Source code in c:/classes/CData.cpp: */ /* CData */ /* CData() */ /* Empty() */ *****/ </pre>

(continued)

Example C++ Coding Standard (continued)

Reuse Instructions	<ul style="list-style-type: none"> - Describe how the program is used: declaration format, parameter values, types, and formats. - Provide warnings of illegal values, overflow conditions, or other conditions that could potentially result in improper operation.
Reuse Instruction Example	<pre> /***** /* Reuse instructions */ /* int PrintLine(char *line_of_character) */ /* Purpose: to print string, 'line_of_character', on one print line */ /* Limitations: the line length must not exceed LINE_LENGTH */ /* Return 0 if printer not ready to print, else 1 */ *****/ </pre>
Identifiers	Use descriptive names for all variable, function names, constants, and other identifiers. Avoid abbreviations or single-letter variables.
Identifier Example	<pre> Int number_of_students; /* This is GOOD */ Float: x4, j, ftave; /* This is BAD */ </pre>
Comments	<ul style="list-style-type: none"> - Document the code so the reader can understand its operation. - Comments should explain both the purpose and behavior of the code. - Comment variable declarations to indicate their purpose.
Good Comment	<pre>If(record_count > limit) /* have all records been processed? */</pre>
Bad Comment	<pre>If(record_count > limit) /* check if record count exceeds limit */</pre>
Major Sections	Precede major program sections by a block comment that describes the processing done in the next section.
Example	<pre> /***** /* The program section examines the contents of the array 'grades' and calcu- */ /* lates the average class grade. */ *****/ </pre>
Blank Spaces	<ul style="list-style-type: none"> - Write programs with sufficient spacing so they do not appear crowded. - Separate every program construct with at least one space.
Indenting	<ul style="list-style-type: none"> - Indent each brace level from the preceding level. - Open and close braces should be on lines by themselves and aligned.
Indenting Example	<pre> while (miss_distance > threshold) { success_code = move_robot (target_location); if (success_code == MOVE_FAILED) { printf("The robot move has failed.\n"); } } </pre>
Capitalization	<ul style="list-style-type: none"> - Capitalize all defines. - Lowercase all other identifiers and reserved words. - To make them readable, user messages may use mixed case.
Capitalization Examples	<pre> #define DEFAULT-NUMBER-OF-STUDENTS 15 int class-size = DEFAULT-NUMBER-OF-STUDENTS; </pre>

Evaluation criteria and suggestions

Evaluation criteria

Your standard must be

- complete
 - legible
-

Suggestions

Keep your standards simple and short.

Do not hesitate to copy or build on the PSP materials.

Coding Standard Template

Purpose	Establecer una práctica de codificación consistente y de alta calidad para programas en Java, siguiendo las convenciones oficiales de Oracle, garantizando legibilidad, mantenibilidad y confiabilidad del código.
Program Headers	Todos los archivos fuente de Java deben comenzar con un encabezado descriptivo estándar que incluya:
Header Format	<code>/* * Nombre de la clase * Versión * Derechos de autor * Descripción: breve explicación del propósito del programa */</code>
Listing Contents	Provide a summary of the listing contents.
Contents Example	<pre> /* * Listing Contents: * * Package: com.ejemplo.universidad * * Classes included: * Estudiante.java – Define la clase Estudiante y sus atributos. * Curso.java – Contiene la información de los cursos y métodos de inscripción. * GestorInscripciones.java – Controla la lógica principal del programa. * * Main methods: * calcularPromedio() – Calcula el promedio final del estudiante. * registrarCurso() – Agrega un curso a la lista del estudiante. * mostrarDatos() – Muestra la información del estudiante en consola. * * External files: * datos/estudiantes.txt – Archivo de entrada con información de estudiantes. * resultados.txt – Archivo de salida con los promedios finales. * * Compilation: * javac com/ejemplo/universidad/*.java * * Execution: * java com.ejemplo.universidad.GestorInscripciones */ </pre>
Reuse Instructions	<ul style="list-style-type: none"> Describe how the program is used. Provide the declaration format, parameter values and types, and parameter limits. Provide warnings of illegal values, overflow conditions, or other conditions that could potentially result in improper operation.
Reuse Example	<code>/** * Calcula el área de un círculo. * @param radio El radio del círculo * @return El área del círculo */</code>
Identifiers	Use descriptive names for all variables, function names, constants, and other identifiers. Avoid abbreviations or single letter variables.

Identifier Example	<u>Tipo</u>	<u>Convención</u>	<u>Ejemplo</u>
	<u>Clases / Interfaces</u>	Sustantivos con <u>mayúscula</u> en cada <u>palabra interna</u>	<u>RegistroAlumno,</u> <u>GestorDatos</u>
	<u>Métodos</u>	Verbos con <u>minúscula</u> <u>inicial</u> y <u>mayúsculas</u> en <u>palabras internas</u>	<u>calcularPromedio(),</u> <u>obtenerNombre()</u>
	Variables	<u>Minúscula inicial</u> y <u>mayúsculas</u> en <u>palabras internas</u>	<u>numeroEstudiantes,</u> <u>totalPuntos</u>
	<u>Constantes</u>	<u>Mayúsculas</u> con <u>guiones</u> <u>bajos</u>	MAX_TAMANO, PUERTO_POR_DEFECTO

(continued)

Coding Standard Template (continued)

Comments	<ul style="list-style-type: none"> • Document the code so that the reader can understand its operation. • Comments should explain both the purpose and behavior of the code. • Comment variable declarations to indicate their purpose.
Good Comment	<pre>// Verifica si el usuario tiene edad suficiente para registrarse if (edad >= 18) { registrarUsuario(); } /** * Calcula el área de un rectángulo. * @param base la longitud de la base * @param altura la altura del rectángulo * @return el área calculada */ public double calcularArea(double base, double altura) { return base * altura; } // Reinicia el contador cuando se alcanza el límite máximo if (contador > LIMITE_MAXIMO) { contador = 0; }</pre>
Bad Comment	<pre>// compara si edad es mayor o igual a 18 if (edad >= 18) { // <-- ya se entiende por el código registrarUsuario(); } // método para calcular el área public double calcularArea(double base, double altura) { // <-- no dice qué hace realmente return base * altura; } // reinicia el contador contador = 0; // <-- repite lo mismo que la instrucción</pre>
Major Sections	Precede major program sections by a block comment that describes the processing that is done in the next section
Example	<pre>/* * Sección que procesa la lista de estudiantes y calcula los promedios. */</pre>
Blank Spaces	<ul style="list-style-type: none"> • Write programs with sufficient spacing so they do not appear crowded. • Separate every program construct with at least one space.
Indenting	<ul style="list-style-type: none"> • Indent every level of brace from the previous one.

	<ul style="list-style-type: none">• Open and closing braces should be on lines by themselves and aligned with each other.												
Indenting Example	<pre>if (condicion) { ejecutar(); } else { detener(); }</pre>												
Capitalization	<ul style="list-style-type: none">• Capitalized all defines.• Lowercase all other identifiers and reserved words.• Messages being output to the user can be mixed-case so as to make a clean user presentation.												
Capitalization Example	<table><tr><th><u>Elemento</u></th><th><u>Regla</u></th><th><u>Ejemplo</u></th></tr><tr><td><u>Clases</u> / Interfaces</td><td><u>Mayúscula inicial y CamelCase</u></td><td><u>UsuarioCliente</u></td></tr><tr><td><u>Métodos</u> / Variables</td><td><u>Minúscula inicial y CamelCase</u></td><td><u>obtenerSaldo()</u>, <u>totalVentas</u></td></tr><tr><td><u>Constantes</u></td><td><u>Mayúsculas con guion bajo</u></td><td>MAX_TAMANO, VALOR_DEFAULT</td></tr></table>	<u>Elemento</u>	<u>Regla</u>	<u>Ejemplo</u>	<u>Clases</u> / Interfaces	<u>Mayúscula inicial y CamelCase</u>	<u>UsuarioCliente</u>	<u>Métodos</u> / Variables	<u>Minúscula inicial y CamelCase</u>	<u>obtenerSaldo()</u> , <u>totalVentas</u>	<u>Constantes</u>	<u>Mayúsculas con guion bajo</u>	MAX_TAMANO, VALOR_DEFAULT
<u>Elemento</u>	<u>Regla</u>	<u>Ejemplo</u>											
<u>Clases</u> / Interfaces	<u>Mayúscula inicial y CamelCase</u>	<u>UsuarioCliente</u>											
<u>Métodos</u> / Variables	<u>Minúscula inicial y CamelCase</u>	<u>obtenerSaldo()</u> , <u>totalVentas</u>											
<u>Constantes</u>	<u>Mayúsculas con guion bajo</u>	MAX_TAMANO, VALOR_DEFAULT											