



UADY
FACULTAD DE
MATEMÁTICAS

Universidad Autónoma de Yucatán

Facultad de Matemáticas

Licenciatura en Ingeniería de Software

Mantenimiento de Software

M.I.T. Carlos Benito Mojica Ruiz

Estándar de Codificación

Versión 1.0

Integrantes de Equipo:

Canul Ordoñez, Josué Israel

Garcilazo Cuevas, Mónica

Leo Fernández, José Carlos

Pool Flores, Endrick Alfredo

Rodríguez Coral, Samuel David

CONTROL DE DOCUMENTACIÓN

Título:	Estándar de Codificación para el Proyecto “Contador Azul”
Referencia:	Canul, J. I., Garcilazo, M., Leo, J. C., Pool, E. A., & Rodríguez, S. D. (2025, febrero 20). <i>Estándar de codificación para el proyecto “Contador Azul”</i> .
Autor:	Canul, J. I., Garcilazo, M., Leo, J. C., Pool, E. A., & Rodríguez, S. D.
Fecha:	20 de febrero del 2025

HISTÓRICO DE VERSIONES

Versión	Fecha	Estado	Responsable	Nombre de archivo
1.0	20/02/2025	A	José Carlos Leo Fernández	Estándar de Codificación (V1.0).docx

HISTÓRICO DE CAMBIOS

Versión	Fecha	Cambios
1.0	20/02/2025	Definición de las reglas de codificación de sentencias lógicas.

Índice

Introducción.....	4
Reglas de Codificación.....	5
Nomenclatura.....	5
Ajuste de Líneas.....	5
Declaraciones de Clases, Interfaces, Registros y Clases Abstractas.....	5
Sentencias.....	6
Sentencia if, if-else, if-else if- else.....	6
Sentencia for.....	7
Sentencia while.....	8
Sentencia do-while.....	8
Sentencia switch.....	8
Sentencia try-catch.....	9

Introducción

En el presente documento se establecen las reglas y directrices que deben seguirse para la codificación del proyecto denominado “Contador Azul”. A su vez este documento define las mismas reglas que deben de respetar los proyectos que sean introducidos en el funcionamiento del programa.

Este documento tiene como propósito servir como una referencia clara y detallada para todos los miembros del equipo de desarrollo, a fin de evitar malentendidos y garantizar una implementación coherente y estandarizada del código.

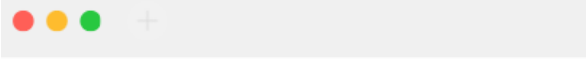
Asimismo, este documento se constituye como un precedente para futuros integrantes del equipo, proporcionándoles la información necesaria para comprender y aplicar las normas de codificación establecidas, asegurando así la continuidad y calidad del desarrollo del proyecto.

Las siguientes reglas se han basado en el documento Java Code Conventions, publicado por Sun Microsystems en 1997. A pesar de su antigüedad, las directrices establecidas en dicho documento siguen siendo relevantes y aplicables en la actualidad.

Reglas de Codificación

Nomenclatura

Toda sentencia escrita en el programa debe de ser en el estilo camel-case. Para declaración de clases, interfaces, registros o clases abstractas, la nomenclatura ha de comenzar con letra mayúscula.




```
public class Object {  
    int variable;  
  
    public native int hashCode();  
}
```

Ajuste de Líneas

Cuando una expresión no encaje en una única línea, se puede separar de acuerdo a los siguientes principios:

- Puedes separar las líneas después de una coma.
- Puedes separar las líneas después de un punto.



```
funcion(expresionLarga1, expresionLarga2, expresionLarga3,  
        expresionLarga4, expresionLarga5);
```

Declaraciones de Clases, Interfaces, Registros y Clases Abstractas

Cuando se declaren clases, interfaces, registros o clases abstractas se deben de seguir las siguientes reglas de formato:

- No debe de existir espacios en blanco entre el nombre del método y el paréntesis de apertura "(".
- La llave de apertura "{" deben de estar al final de la misma línea de la sentencia de declaración

- La llave de cierre "}" debe comenzar en una línea por sí sola, indentada para coincidir con su correspondiente declaración de apertura.
- Los métodos deben de estar separados por una línea en blanco



```
public void metodo(parametros) {
```



```
public void metodo(parametros) {
    //código
}

public void metodo(parametros) {
    //código
}
```



```
public interface interfaz {
```

Sentencias


Solo se debe de tener una sentencia por línea:



```
argv++; argc--; //no se encuentra permitido
```

Sentencia if, if-else, if-else if- else

Las sentencias "if", en cualquiera de sus formas, no deben contener espacios en blanco innecesarios. Además, el corchete de apertura "{" debe estar en la misma línea que la palabra clave if y la condición. Aunque en una sentencia "if" simple se pueden omitir los corchetes, esto no está permitido.




```
if (condición) {
    sentencia
}

if (condición) {
    sentencia
} else {
    sentencia
}

if (condición) {
    sentencia
} else if (condición) {
    sentencia
} else (condición) {
    sentencia
}
```


Sentencia for

La sentencia “for” no debe de contener espacios en blanco innecesarios. Aunado a ello, el corchete de apertura ha de encontrarse en la misma línea de código que la palabra clave “for”.



```
for (inicialización; condición; actualización) {
    sentencias
}
```


No se encuentran permitidas las sentencias “for” de una sola línea:



```
for (inicialización; condición; actualización);
```

Sentencia while

La sentencia “while” no debe de contener espacios en blanco innecesarios. Aunado a ello, el corchete de apertura ha de encontrarse en la misma línea de código que la palabra clave “while”.



```
while (condición) {  
    sentencias  
}
```


No se encuentran permitidas las sentencias “while” de una sola línea:



```
while (condición);
```

Sentencia do-while


La sentencia “do-while” ha de codificarse evitando espacios innecesarios. El corchete de apertura “{” ha de encontrarse en la misma línea que la palabra clave “for”.



```
do {  
    sentencias;  
} while (condición);
```

Sentencia switch


La sentencia “switch” ha de codificarse evitando espacios innecesarios. El corchete de apertura “{” ha de encontrarse en la misma línea que la palabra clave “switch”. Siempre debe de incluirse caso “default”.



```
switch (condición) {
  case ABC:
    sentencias;
    break;
  case DEF:
    sentencias;
    break;
  case XYZ:
    sentencias;
    break;
  default:
    statements;
    break;
}
```

Sentencia try-catch

La sentencia “try-catch” ha de codificarse evitando espacios innecesarios. El corchete de apertura “{” ha de encontrarse en la misma línea que la palabra clave “try”. Mismo caso para el corchete de apertura para el “catch”.



```
try {
  sentencias;
} catch (ExceptionClass e) {
  sentencias;
}
```