**IBERO I Rúbrica para la evaluación de estilos.**

**Objetivos de la evaluación**

* Mejorar la calidad del estilo de programación que utilizan los estudiantes en la Universidad Iberoamericana.
* Medir el desempeño docente en las asignaturas de programación en términos del estilo que aprenden los estudiantes.

Criterios de evaluación

* Programación modular
* Calidad de los nombres de variables, funciones y argumentos.
* Calidad de los comentarios.
* Uso correcto de los tipos.

**Otras restricciones**

La evaluación debe poderse completar en hora y media.

Rúbrica de evaluación

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Criterio | Excelente | Bueno | Mediocre | Insuficiente |
| Legibilidad del código | * Se implementan funciones atómicas: Las funciones realizan una tarea a la vez. * Las funciones se encuentran en archivos independientes (modularidad). * Cada función está documentada describiendo el problema que resuelve y dando un ejemplo de uso. * La redacción de los comentarios es clara y concisa. * Se eliminaron los comentarios redundantes o poco informativos p.e. int i; // i es entero. * Los bloques permiten diferenciar los elementos visualmente. * Se usa de forma consistente un estilo de código. | Faltan hasta 2 elementos. | Faltan de 3 a 5 elementos. | Faltan 6 o más elementos. |
| Calidad de identificadores | * Todos los identificadores describen la naturaleza y propósito del objeto que identifican, se expresan en el lenguaje del domino del problema. * Todos los identificadores omiten detalles de implementación. p.e. “students” en vez de “student\_array”. * Todos los archivos describen la función, programa o biblioteca que contienen. * Todas las funciones incluyen un verbo, salvo que sea más claro que sólo sea un sustantivo. * Todas las variables incluyen un sustantivo. * Todos los argumentos incluyen sustantivos. * Todas las variables están definidas dentro de funciones. * Están permitidos indicadores mínimos si están acotados al contexto actual y se entiende su propósito, p.e. “for (int u=0; u <= ACTIVE\_USERS; u++) {…};” | Faltan de 2 a 3 elementos. | Faltan de 4 a 6 elementos. | Faltan 7 o más elementos. |
| Estilo Idiomático | * El código utiliza las bibliotecas estándar. * Las funciones desarrolladas por los estudiantes conforman bibliotecas. * Cada función existe en su propio archivo. * Las funciones semánticamente relacionadas pueden accederse desde un mismo encabezado. * Las funciones usan un valor de retorno, con un tipo apropiado para la aplicación (p.e. void). * Se generan estructuras de datos cuando son adecuadas. * El código muestra dominio de variables por valor y por referencia. | Faltan hasta 2 elementos. | Faltan de 3 a 4 elementos. | Faltan 5 o más elementos. |
| Mantenibilidad del proyecto | * Usa un control de versiones. * Tiene al menos un registro en control de versiones por cada función desarrollada. * Cada registro describe qué se hizo y si es necesario explica por qué. * Cada registro contiene la siguiente estructura:   + La primera línea describe lo que se hizo contiene hasta 72 caracteres.   + La segunda línea debe estar en blanco.   + Si es necesario explicar los cambios, se describen a partir de la tercera línea. * Cada registro tiene un único cambio, organizado semánticamente. * El proyecto cuenta con un sistema automático para compilar y encadenar el programa. | Falta 1 elemento. | Faltan entre 2 y 3 elementos. | Faltan 4 o más elementos. |

**Ejercicios para evaluación**

1. Cada uno de estos programas contiene un error por el cual no compila u obtiene un resultado incorrecto. Corrige el error y registra tu resultado en el control de versiones.

Tipo de variable incorrecto.  
Falta de la sentencia para devolver la variable.  
Error de desbordamiento de buffer.  
Error por uno (Ciclos).  
La variable está llamada por valor y no por referencia.