PROGRAMACIÓN II - GRUPO de Prácticas C

PRÁCTICA Nº2

6 de marzo 2024 8:30-11:30

INSTRUCCIONES

1. Todos debéis acceder a la **SALA** del campus virtual que tenéis disponible.
2. Durante el desarrollo de la sesión práctica debéis seguir el procedimiento de trabajo en la sala para la realización de las actividades evaluables. Recordad que es **imprescindible GRABAR toda la sesión** y **COMPARTIR la pantalla completa**.
3. La entrega de la práctica sólo se admitirá a través de la **actividad** disponible en el campus virtual de la asignatura de **Prácticas de Programación II antes de la hora de finalización de la sesión de prácticas**.
4. Se debe entregar **un único fichero** en **formato .txt o .cpp, sin comprimir**, con el nombre de ***PIIP2GC.***
5. El fichero entregado debe **incluir el nombre de los integrantes del equipo**.
6. Aunque las prácticas se realizan en grupos de dos integrantes, para su evaluación, **ambos deberán hacer la entrega a través de su campus virtual**. En otro caso, la práctica quedará **sin evaluar** y supondrá un **0 en su calificación**.
7. ***El incumplimiento de alguna de las instrucciones sobre la realización/entrega de la Práctica supondrá su descalificación inmediata (0 puntos).***

Al final del enunciado de la práctica tenéis la puntuación de los diferentes hitos que debéis cumplir durante la sesión. Se recomienda que realicéis de forma previa al desarrollo una lectura comprensiva del enunciado, así como un debate en cada equipo con el que podáis planificar la estrategia a seguir durante la práctica.

Recordad que podéis/debéis utilizar el ***chuletario*** durante la sesión, y que debéis tener en cuenta la rúbrica general de evaluación. Por ser una práctica podéis consultarla durante la sesión sin problema (está disponible en el campus virtual de la asignatura).

EJERCICIO PRÁCTICO [90% puntuación] – 180minutos

En esta segunda práctica vamos a continuar trabajando con una de las clases desarrolladas en la primera práctica, clase Publicacion. Para ello, se debe **partir del fichero que se adjunta en la práctica II** del grupo C.

A continuación, describiremos los requisitos que debéis tener en cuenta para cada una de ellas.

Clase ***Publicacion***: **(9 puntos)**

- Atributos de la práctica nº1:
  - **Título**.
  - **Autoria.**
  - **FechaPublic.**
  - **NumPaginas**.
- Miembros de la práctica nº1:
  - **Métodos getter**.
  - **Métodos setter**.
  - **Método mostrarPublicacion**.
- Constructores:
  - Por defecto [**0,25 puntos**]
  - Parametrizado: debe recibir por argumento el título, los autores, la fecha de publicación y el número de páginas [**0,5 puntos**]
  - Por copia [**0,5 punto**]
- Destructor de la clase: en este caso el destructor no debe realizar ninguna acción, pero debe estar incluido en la clase tanto su declaración como su definición [**0,5 puntos**]
- Modificar el control de integridad implementado en la práctica 1 para que: [**1,5 puntos**]
  - En lugar de asignar al primer autor de la autoría el valor "SIN DETERMINAR", se lance una excepción de tipo ***int*** con el valor 1**.**
  - En lugar de asignar el valor 1 al mes de la fecha de publicación cuando su valor no esté comprendido entre 1 y 12, se lance una excepción de tipo ***int*** con el valor 2.

- En lugar de asignar el valor 1900 al año de la fecha de publicación cuando su valor sea menor a 1900 y mayor que el año actual, se lance una excepción tipo ***int*** con el valor 3.
- En lugar de asignar el valor 0 al número de páginas cuando su valor sea inferior a cero, se lance una excepción tipo ***int*** con el valor 4.

En la función principal del programa vamos a implementar la gestión de una estantería de una biblioteca, para ello debéis realizar las siguientes tareas:

- Declarar una variable de tipo contenedor que consideres oportuno para que sea capaz de representar la ocupación de los diferentes estantes que tiene una estantería de la biblioteca pensada para ubicar un total de 100 publicaciones. Para ello hay que considerar que las publicaciones están dispuestas en forma de matriz, con un total de 5 estantes (filas) y 20 posiciones (columnas) para cada publicación. En un primer momento la estantería estará vacía, es decir, no hay publicaciones en ninguno de sus estantes. Esa situación se debe representar con objetos inicializados con el constructor por defecto [**0,5 puntos**]
- Implementar una función que nos sirva para gestionar el **menú** del programa (mostrar las diferentes opciones disponibles y capturar la selección realizada por el usuario). Dicho menú debe proporcionar las siguientes opciones: [**0,5 puntos**]
  - Visualizar el estado de ocupación de la estantería.
  - Colocar una publicación en la estantería.
  - Bucar una publicación de la estantería.
- Implementar un bucle en el que, a partir de la visualización de las opciones del menú y la captura de la selección realizada por el trabajador de la biblioteca, se vayan atendiendo las diferentes opciones posibles a realizar para la gestión de la estantería [**0,75 puntos**].
- **Importante**, para optar al 100% de la calificación de cada apartado hay que utilizar funciones. Al menos una por opción del menú:
  - **Visualizar** el estado de ocupación de la estantería [**1 puntos**]

- **Colocar** una publicación en la estantería a partir de una posición (fila/estante y columna/ubicación en el estante). En caso de que dicha posición en la estantería esté ocupada, la función lanzará una excepción de tipo ***string*** con el texto “Sitio ocupado con otra publicación”. [**1,5 puntos**]
- **Buscar** una publicación de la estantería a partir de una posición (fila/estante y columna/ubicación en el estante) y mostrar su contenido. En caso de que dicha posición esté vacía, la función lanzará una excepción de tipo ***string*** con el texto “Sitio vacio, no hay publicación”. [**1,5 puntos**]

- Se deben de gestionar de forma adecuada todas las excepciones de la clase Publicacion (capturar la excepción y gestionarla mostrando por pantalla que situación anómala se ha dado y la información que la excepción a transmitido).

Recuerda que:

- Todos los atributos (variables miembro) deben ser privados.
- En el nivel de acceso público de la clase deben estar los métodos que queramos ofrecer (interfaz pública de la clase).
- El constructor por defecto es el constructor que no recibe ningún argumento.
- Los constructores de una clase son unos métodos particulares que no "devuelven nada", se llaman de la misma forma que la clase y se pueden sobrecargar (puede tener varios constructores que reciban distintos argumentos).
- Debes poner "los const & en su sitio…", recuerda la rúbrica de evaluación general y la penalización al 40% que supone no hacerlo bien…
    - Paso por referencia constante en argumentos de tipo no simple
    - Retorno de valores de tipo no simple que "ya existan"
    - Los métodos que no modifiquen el estado de la clase deben ser declarados como métodos constantes.

RÚBRICA DE CALIFICACIÓN

- El 10% de la nota de la práctica, se obtendrá sobre trabajo realizado durante la sesión (resolución de preguntas, dominio de la materia, etc.). Esta nota será individual, y se considerarán para su valoración aspectos como:
    - **Defensa individual** a cuestiones que plantee el profesor durante la práctica.
    - **Trabajo en equipo**.
- El 90% restante será sobre la puntuación del trabajo entregado antes de la finalización de cada práctica (misma nota para cada integrante del equipo), siguiendo la siguiente rúbrica general y particular para cada apartado puntuable:

Criterios generales de evaluación

	% máx. (*)
Funciones/Métodos: Si no se usa el paso por referencia constante cuando las variables de los parámetros de entrada no son de tipo simple.	40%
Tipos de datos y variables: • Uso de variables globales (fuera del ámbito de una función). • Si no se usan los tipos contenedor vistos en clase (std::array; std::vector; std::set; std::string, etc.) para las variables que lo necesiten. • Si no se usan punteros inteligentes (std::unique_ptr; std::shared_ptr) cuando sea necesario. • Si se utilizan punteros clásicos.	 0% 0% 0% 0%
No utilizar funciones/métodos cuando sea apropiado o se **indique de forma explícita** en el apartado/ejercicio de la **actividad/examen**.	40%
El **programa no compila** o **no se asemeja** a lo pedido**.**	0%
Si no se cumplen los **criterios de entrega** indicados en la **actividad/examen.**	0%
Si los apartados/ejercicios de la **actividad/examen** son resueltos de forma **NO algorítmica**, es decir, proponiendo una **solución particular** para unos determinados datos/valores en lugar de ser una **solución general** independiente de los datos/valores.	0%

Criterios particulares de evaluación

	% máx. (*)
El elemento evaluable no compila o no se asemeja a lo que se pide	0%
El elemento evaluable no se aproxima suficientemente a lo pedido	40%
El elemento evaluable se aproxima suficientemente a lo pedido	60%
El elemento evaluable funciona correctamente y las estrategias y elementos de código elegidos son adecuados.	100%

(*) El porcentaje (% máx.) representa el valor máximo sobre el que se evalúa el elemento indicado.