

## Ejercicios de Programación Orientada a Objetos

10%

1. Implemente una clase en C++ llamada `Vector3D` que represente un vector en un espacio tridimensional. La clase debe incluir:

- Un constructor que inicialice las coordenadas del vector.
- Métodos para calcular la magnitud del vector y para normalizarlo.
- Sobrecarga de los operadores de suma, resta y producto escalar entre dos vectores.
- Un método para imprimir las coordenadas del vector en la consola.

10%

2. Cree una clase en C++ llamada `Matriz` que represente una matriz cuadrada de tamaño  $n \times n$ . La clase debe incluir:

- Un constructor que inicialice la matriz con valores aleatorios.
- Un método para calcular la transpuesta de la matriz.
- Un método para multiplicar la matriz por otra matriz del mismo tamaño.
- Un método para imprimir la matriz en la consola.
- Sobrecarga del operador de acceso para permitir la indexación de elementos de la matriz.
- Un destructor que libere la memoria asignada dinámicamente.

10%

3. Diseñe una clase en C++ llamada `Polinomio` que represente un polinomio de grado  $n$ . La clase debe incluir:

- Un constructor que inicialice los coeficientes del polinomio.
- Un método para evaluar el polinomio en un valor dado de  $x$ .
- Sobrecarga de los operadores de suma y multiplicación entre dos polinomios.
- Un método para imprimir el polinomio en la forma estándar.
- Un destructor que libere la memoria asignada dinámicamente.
- Un método para derivar el polinomio y devolver un nuevo objeto `Polinomio` que represente la derivada.

10%

4. Declarar dos clases: números complejos en forma binomial ( $a+ib$ ) y números complejos en forma polar ( $a.e^b$ ). La primera se llamará `complex` y la segunda `polar`. Implementar los operadores `+`, `*`, `-`, `/`, `conj` para los nuevos tipos de datos. Asimismo, implementar las necesarias funciones de conversión. Implementar los constructores flexibles. Implementar una función llamada `acumula` que sume un número indefinido de números complejos y polares.

20 %

5. Se quiere escribir un programa para manipular ecuaciones algebraicas o polinómicas dependientes de una variable. Por ejemplo:

$$P(x) = 4x^3 + 3x^2 - 2x + 7 \text{ más } Q(x) = 5x^2 - 6x + 10 \text{ igual a } R(x) = 4x^3 + 8x^2 - 8x + 17; \quad (1)$$

Cada término del polinomio será representado por una clase `CTermino` y cada polinomio por una clase `CPolinomio`. La clase `CTermino` tendrá dos atributos privados: coeficiente y exponente, y los métodos necesarios para permitir al menos:

- Construir un término, iniciado a 0 por omisión.
- Acceder al coeficiente de un término.
- Acceder al exponente de un término.
- Obtener la cadena de caracteres equivalente a un término con el formato siguiente:

$$\{\pm\}7x^4.$$

La clase `CPolinomio` tendrá un atributo privado (polinomio) que será una matriz que almacenará los términos del polinomio, así como los métodos necesarios para permitir al menos:

- Construir un polinomio, iniciado con cero términos por omisión.
- Obtener el número de términos que tiene actualmente el polinomio.
- Asignar un término a un polinomio, colocándolo en orden ascendente del exponente. Si el término existe, se sumarán los coeficientes. Si el coeficiente es nulo, no se realizará ninguna operación. Cada vez que se inserte un nuevo término, se incrementará automáticamente el tamaño del polinomio en 1. El método encargado de esta operación tendrá un parámetro de la clase `CTermino`.
- Sumar dos polinomios. El polinomio resultante quedará también ordenado en orden ascendente del exponente. El método encargado de esta operación tendrá un parámetro de la clase `CPolinomio` y devolverá un nuevo objeto de la clase `CPolinomio`.
- Multiplicar dos polinomios. El polinomio resultante quedará también ordenado en orden ascendente del exponente. El método encargado de esta operación tendrá un parámetro de la clase `CPolinomio` y devolverá un nuevo objeto de la clase `CPolinomio`.
- Obtener la cadena de caracteres equivalente a un polinomio con el formato siguiente:

$$4x^3 + 3x^2 - 2x + 7.$$

40 %

6. Se requiere desarrollar el juego de dominó utilizando Programación Orientada a Objetos en C++. El juego debe incluir las siguientes características:

- Una clase `Ficha` que represente una ficha de dominó con dos valores (números) en cada extremo.
- Una clase `Jugador` que represente a un jugador del juego, incluyendo su nombre y las fichas que posee.
- Una clase `JuegoDomino` que gestione el flujo del juego, incluyendo la distribución de fichas, el turno de los jugadores y la verificación de las reglas del juego.
- Implementar métodos para que los jugadores puedan colocar fichas en la mesa, verificar si un jugador puede jugar una ficha y determinar el ganador del juego.

- Incluir una interfaz de usuario simple en la consola para interactuar con el juego, permitiendo a los jugadores ver sus fichas, el estado del juego y realizar sus movimientos.
- Asegurarse de que el código esté bien estructurado, utilizando principios de POO como encapsulación, herencia y polimorfismo donde sea apropiado.
- El juego debe permitir entre dos y cuatro jugadores humanos competir entre sí.
- Implementar un sistema de puntuación que registre las victorias de cada jugador a lo largo de múltiples rondas.
- Incluir comentarios en el código para explicar la funcionalidad de cada clase y método.
- Proporcionar un archivo README que explique cómo compilar y ejecutar el juego, así como las reglas básicas del dominó.
- Incluir un mecanismo para reiniciar el juego sin necesidad de reiniciar el programa completo.
- Asegurarse de manejar adecuadamente los casos en los que un jugador no pueda realizar un movimiento válido.
- Se debe tener en cuenta que cada juego comienza con un conjunto completo de fichas de dominó (28 fichas en total). Además, las fichas deben ser distribuidas aleatoriamente entre los jugadores al inicio de cada partida.
- Las partidas son bucles hasta que un jugador gane o se llegue a un estado de bloqueo donde ningún jugador pueda realizar un movimiento válido.

## Entrega

El trabajo práctico debe ser entregado en formato digital a través de un repositorio en la plataforma Github. Asegúrese de incluir un archivo README con instrucciones claras sobre cómo compilar y ejecutar su código. También, el archivo README debe explicar claramente la metodología que utiliza cada programa para resolver el problema dado. La fecha límite para la entrega es el 11 de Noviembre del 2025 a las 23:59 horas. No se aceptarán entregas tardías sin una justificación válida.