Técnicas de programación avanzada

Curso 2023-2024

Tema 3.1 Concepto de patrones

Tema 1: Objetos y memoria.

1.1 Características básicas del lenguaje. Primer programa. Compilación y Ejecución. IDE.

1.2 Sentencias de control. Secuencia, selección e iteración.

1.3 Abstracción. Clases, objetos, métodos y atributos.

1.4 Sobrecarga de métodos y encapsulamiento.

Tema 2. Otros conceptos fundamentales de la Programación Orientada a Objetos

2.1 Herencia. Interfaces y clases abstractas. Agregación.

2.2 Polimorfismo.

2.3 Gestión de Excepciones.

2.4 Genericidad y plantillas.

2.5 Utilidades. Entrada y Salida.

2.6 Anotaciones.

Tema 3. Patrones de Diseño.

3.1 Concepto de Patrones de Diseño.

3.2 Patrones de creación.

3.3 Patrones estructurales.

3.4 Patrones de comportamiento.

Tema 4. Programación de Interfaces.

4.1 Interfaces Gráficas de Usuario.

4.2 Gestión de eventos.

Tema 5. Temas Avanzados.

5.1 Concurrencia.

5.2 Inversión de Control. Definición y ejemplos. Inyección de dependencias.

5.3 Expresiones avanzadas del lenguaje.

3.1 Concepto de Patrones de Diseño

Introducción

Los **patrones de diseño** son soluciones habituales a problemas que ocurren con frecuencia en el diseño de software. No son plantillas, son consejos.

- Los **patrones creacionales** proporcionan mecanismos de creación de objetos que incrementan la flexibilidad y la reutilización de código existente.
- Los **patrones estructurales** explican cómo ensamblar objetos y clases en estructuras más grandes a la vez que se mantiene la flexibilidad y eficiencia de la estructura.
- Los **patrones de comportamiento** se encargan de una comunicación efectiva y la asignación de responsabilidades entre objetos.

3.1 Concepto de Patrones de Diseño.

Tipos

Familias	Se dedican a…	Ejemplos de patrones
Creacionales	Proporcionan mecanismos de creación de objetos que incrementan la flexibilidad y la reutilización del código existente.	- **Factory Method** - **Abstract Factory** - Builder - Prototype - **Singleton**
Estructurales	Explican cómo ensamblar objetos y clases en estructuras más grandes, mientras se mantiene la flexibilidad y eficiencia de la estructura.	- **Adapter** - Bridge - Composite - **Decorator** - Facade - Flyweight - Proxy
De comportamiento	Tratan con algoritmos y la asignación de responsabilidades entre objetos.	- **Chain of responsibility** / - Command - Iterator / - Mediator - Memento / - **Observer** - **State** / - Strategy - **Template method** / - Visitor

Técnicas de Programación Avanzada

```
exit(); //Gracias!
```