UT03. DISEÑO ORIENTADO A OBJETOS. ELABORACIÓN DE DIAGRAMAS ESTRUCTURALES.

Entornos de Desarrollo

1DAW – C.I.F.P. Carlos III - Cartagena

Índice

1. Introducción a la orientación a objetos

Introducción

- **Enfoque estructurado**
  - Proceso centrado en los **procedimientos**
  - Se codifican mediante funciones que actúan sobre **estructuras de datos** ➔ **programación estructurada.**
  - **Qué** hay que hacer ➔ funcionalidad
- **Enfoque orientado a objetos**
  - Simula los elementos de la realidad asociada al problema de la forma más cercana posible.
  - **Abstracción:** permite representar elementos ➔ **objeto**
    - Conjunto de atributos, **datos.**
    - Conjunto de operaciones, **comportamiento.**
    - **Mensaje.** Orden de ejecución de una operación determinada
  - **Aplicación orientada a objetos**
    - Conjunto de **objetos** que interaccionan a través de **mensajes** para producir resultados.
    - Los objetos similares se abstraen en **clases,** se dice que un objeto es una **instancia de una clase.**

Introducción

- Ejecución de una aplicación OO
  - Creación de objetos a medida que se necesitan
  - Los mensajes se mueven de un objeto a otro (o del usuario a un objeto).
  - Borrado de objetos cuando ya no se necesitan liberación de la memoria.

2. Conceptos de orientación a objetos

Conceptos de orientación a objetos

- **Objeto**
  - Unidad dentro de un programa de computadora que consta de un estado y de un comportamiento, que a su vez constan de datos almacenados y de tareas realizables durante el tiempo de ejecución. Un objeto puede ser creado
    - Instanciando una clase (POO)
    - Mediante escritura directa de código y al replicación (Programación basada en prototipos)
  - Es un elemento del programa que integra sus propios datos y su propio funcionamiento (propiedades y métodos).
- **Clase**
  - Define el tipo de objeto, cómo funciona un determinado tipo de objeto.
- **Método**
  - Operación de un determinado objeto.
  - Mensaje ≅ llamada a una operación de un objeto.

Conceptos de orientación a objetos

- **Abstracción.**
  - Captura características y comportamientos similares de un conjunto de objetos ➔ conjunto de clases.
- **Encapsulación.**
  - Significa agrupar todos los elementos que pueden considerarse pertenecientes a una misma entidad, al mismo nivel de abstracción ➔ cohesión de los componentes del sistema.
- **Modularidad**
  - Subdividir una aplicación en partes más pequeñas (módulos), cada una de las cuales debe ser tan independiente como sea posible de la aplicación en sí y de las restantes partes.
  - En POO, clase ≅ módulo más básico del sistema
- **Cohesión**
  - Alta cohesión ➔ cada módulo realiza una única tarea trabajando sobre una única estructura de datos
- **Principio de ocultación**
  - Aísla las propiedades de un objeto contra su modificación por quien no tenga derecho a acceder a ellas. Reduce la propagación de efectos colaterales cuando se producen cambios.

Conceptos de OO. II

- **Polimorfismo**
  - Consiste en reunir bajo el mismo nombre comportamientos diferentes.
  - La selección del comportamiento dependerá del objeto que lo ejecute
- **Herencia**
  - Relación que se establece entre objetos en los que unos utilizan las propiedades y comportamientos de otros formando una jerarquía. Los objetos heredan las propiedades y el comportamiento de todas las clases a las que pertenecen.
- **Recolección de basura**
  - Destrucción automática de los objetos → desvinculación de la memoria asociada.

Ventajas

- Desarrollo de software en
  - menos tiempo
  - con menos coste
  - mayor calidad gracias a la reutilización➔ código reusable en otras aplicaciones.
- Aumento de la calidad de los sistemas, haciéndolos más extensibles ➔ facilidad para aumentar o modificar la funcionalidad de la aplicación.
- Facilidad de modificación y mantenimiento por la modularidad y encapsulación
- Adaptación al entorno y el cambio con aplicaciones escalables➔ propiedad para ampliar un sistema sin rehacer su diseño y sin disminuir su rendimiento

Clases, atributos y métodos

- Los objetos de un sistema se abstraen en **clases** formada por un conjunto de procedimientos y datos.
- Propósito de la clase: definir **abstracciones** y favorecer la **modularidad**
- Miembros:
  - **Nombre**
  - **Atributos**: conjunto de características asociadas a una clase. Definen el **estado** del objeto. Se definen por su nombre y su tipo, que puede ser simple o compuesto como otra clase.
  - **Protocolo**: Operaciones (métodos, mensajes) que manipulan el estado.
    - Un ***método*** es el procedimiento o función que se invoca para actuar sobre un objeto.
    - Un ***mensaje*** es el resultado de cierta acción efectuada por un objeto. El conjunto de mensajes a los cuales puede responder un objeto se le conoce como *protocolo del objeto*.

2.2.1.- Polimorfismo o sobrecarga

Vamos a ver un ejemplo de método sobrecargado o con la propiedad de polimorfismo.

Esto significa crear distintas variantes del mismo método. Ejemplo:

```
public class Matemáticas{
    public double suma(double x, double y) {
    return x+y;
    }
    public double suma(double x, double y, double z){
    return x+y+z;
    }
public double suma (double[] array){
double total =0;
for(int i=0; i<array.length;i++){
total+=array[i];
}
return total;
}
}
```

La clase Matemáticas posee tres versiones del método suma: una versión que suma dos números double, otra que suma tres y la última que suma todos los miembros de un array de números decimales. Desde el código se puede utilizar cualquiera de las tres versiones según convenga.En definitiva, el método suma es polimórfico.

Visibilidad

- **Principio de ocultación** → aísla el **estado** de manera que sólo se puede cambiar mediante las operaciones definidas en una clase → protege los datos de modificaciones por alguien que no tenga derecho a acceder a ellos → las clases se dividan en dos partes:
  - **Interfaz**: visión externa de una clase.
  - **Implementación**: representación de la abstracción, y mecanismos que conducen al comportamiento deseado.
- **Niveles de ocultación → visibilidad →** define el tipo de acceso que se permite a atributos.
  - **Público**: Se pueden acceder desde cualquier clase y cualquier parte del programa.
  - **Privado**: Sólo se pueden acceder desde operaciones de la clase.
  - **Protegido**: Sólo se pueden acceder desde operaciones de la clase o de clases derivadas en cualquier nivel.

Visibilidad. II

- Norma general
  - Estado ➔ **privado**
  - Operaciones del comportamiento➔ **públicas**
  - Operaciones auxiliares para definir el comportamiento ➔ **privadas/protegidas**

Objetos. Instanciación

- Clase → abstracción
  - Una clase es "un conjunto de objetos que comparten una estructura común y un comportamiento común." (Booch)
- Creación de objeto de clase → instancia de clase.
- Un objeto se define por:
  - **Su estado**: definido por el conjunto de valores de atributos.
  - **Su comportamiento**: definido por los métodos públicos de su clase.
  - **Su tiempo de vida**: intervalo de tiempo a lo largo del programa en el que el objeto existe, desde su creación (**instanciación**) hasta la destrucción del objeto.
- **Clase abstracta**: no puede se instanciada.
  - Uso: definir métodos genéricos para sus clases derivadas