Programación orientada a objetos - Informática II

Clase 2 – El paradigma

¡Hola! ¿Cómo están? Continuamos bajo la modalidad en línea de Informática II y esta semana comenzamos a trabajar sobre un nuevo lenguaje, en la clase de hoy nos centraremos en el paradigma de la programación orientada a objetos.

Programación orientada a objetos

Cuando trabajamos en Informática I la programación en C lo hicimos bajo un paradigma denominado programación estructurada.

La programación orientada a objetos se apoya en la abstracción de un elemento de la vida cotidiana o de un recurso que interactúa con otros elementos a partir de su descripción (lo que es) y de su comportamiento (como se relaciona).

Este nuevo paradigma se apoya en los siguientes pilares:

- Abstracción
- Encapsulamiento
- Polimorfismo
- Herencia

Les proponemos ir viendo estas características una por una, de forma de ir comprendiendo este nuevo paradigma:

Abstracción

Nos permite describir las características esenciales de un objeto, como pueden modificarse sus características y cómo se comunica con otros objetos dentro de un sistema. Lo que no debemos perder de vista es que la abstracción nos permitirá describir un modelo y como todo modelo intentará representar a la realidad pero no es la realidad, y dependerá de cada observador.

Encapsulamiento

Nos permite definir qué acciones serán propias del objeto y que acciones son las que permitirán vincularse con el mundo exterior. Es decir, que podemos definir cuáles son los elementos que hacen al **comportamiento interno** y cuales al **comportamiento externo**, es decir lo que permite que se relacione con otros objetos.

Polimorfismo

En el lenguaje C, tal como comentamos en la clase pasada, una función poseía un prototipo y eso no cambiaba bajo ningún concepto, por ejemplo si teníamos una función que sumaba enteros su prototipo probablemente sea:
int suma (int valor1, int valor2);

Ahora si queremos que la función sume dos valores reales, probablemente su prototipo será:
float sumaReales(float valor1, float valor2);

El polimorfismo nos permite describir una acción denominada **suma** la cual forme parte de un **objeto** *calculadora.*

Dicha función se vinculará con el mundo exterior: es decir un *usuario*, un *operador*, otra *calculadora* a través de una interfaz común denominada **suma.** Puertas adentro del objeto esa función **suma** será resuelta por el programador (ustedes), de diferentes formas.

Entonces el encapsulamiento, nos permite describir una misma función o acción la cual podrá tener diferentes interfaces (firmas) y se utilizará de determinada manera en función del contexto. Es decir, comportamientos diferentes en función del contexto de uso. A esto lo llamamos **sobrecarga** de una acción.

Herencia

Los objetos no se encuentran aislados sino que se relacionan entre sí. La herencia posibilita contar con características y comportamientos previamente definidos y aprovechar los recursos ya desarrollados. Se trata de no reinventar la rueda sino de un concepto pensando en la reutilización de código.

La herencia nos permite definir jerarquías, y extender el comportamiento de cierto objeto a un objeto hijo. En POO, hablamos de clases base y clases derivadas pero esto lo iremos viendo a poco en las próximas clases.

Final de esta clase

Bueno, este es el final de la clase de C++ sobre este nuevo paradigma de programación.

Los y las invitamos a ver la video-clase que complementa este material de lectura inicial.

Si surgen dudas o consultas, pueden escribirnos en el foro correspondiente.

En la próxima clase, comenzaremos a trabajar sobre la sintaxis y las clases en C++
¡Nos leemos!