UNIVERSIDAD DE LAS AMÉRICAS PUEBLA



DEPARTAMENTO DE ELECTRÓNICA, MECATRÓNICA Y SISTEMAS

LENGUAJES Y PARADIGMAS DE LA PROGRAMACIÓN

M. JUAN CARLOS GALÁN

“TAREA 3: CONCEPTOS DE LA POO”

JOSE ALONSO ZAVALA PULIDO 149974

SECCIÓN 1

SAN ANDRÉS CHOLULA, A 02 DE OCUBRE DE 2015

**Variables estáticas**

Una variable estática es una variable que ha sido ubicada estáticamente. Normalmente una variable estática tiene un ámbito más amplio que otras variables. Los valores de variables estáticas se pueden establecer una vez (durante el tiempo de ejecución) o se pueden cambiar en múltiples ocasiones durante la ejecución del programa. La terminología "variable estática" se basa en C y C++, pero también se usa en muchos lenguajes de programación derivados. En lenguajes de diferente origen el mismo concepto puede denominarse "variable global".

**Memoria dinámica (lenguaje C)**

Es memoria que se reserva en tiempo de ejecución. Su principal frente a la estática, es que su tamaño puede variar durante la ejecución del programa. En el lenguaje C, se encarga de liberar esta memoria cuando no se utilice más. El uso de memoria dinámica es necesario cuando no conocemos el número de datos o elementos a tratar; sin embargo es algo más lento, ya que el tiempo de ejecución depende del espacio que se va a usar. Los punteros proporcionan el soporte necesario para el potente sistema de asignación dinámica de memoria de C.

**Clase**

Una clase es una plantilla para la creación de objetos de datos según un modelo predefinido. Las clases se utilizan para representar entidades o conceptos, como los sustantivos en el lenguaje. Cada clase es un modelo que define un conjunto de variables -el estado, y métodos apropiados para operar con dichos datos -el comportamiento. Cada objeto creado a partir de la clase se denomina instancia de la clase.

**Objeto**

En POO un objeto es una unidad dentro de un programa de computadora que consta de un estado y de un comportamiento, que a su vez constan respectivamente de datos almacenados y de tareas realizables durante el tiempo de ejecución. Un objeto puede ser creado instanciando una clase, como ocurre en la programación orientada a objetos, o mediante escritura directa de código y la replicación otros objetos, como ocurre en la programación basada en prototipos.

Estos objetos interactúan unos con otros, en contraposición a la visión tradicional en la cual un programa es una colección de subrutinas (funciones o procedimientos), o simplemente una lista de instrucciones para el computador. Cada objeto es capaz de recibir mensajes, procesar datos y enviar mensajes a otros objetos de manera similar a un servicio.

**Instancia**

Una instancia se refiere a una realización específica de una clase o prototipo determinados. En general, cuando se ejecuta un programa en un computador, se dice que éste se instancia. En lenguajes que crean objetos a partir de clases, un objeto es una instancia de una clase. Esto es, un miembro de una clase que tiene atributos en lugar de variables.

**Herencia**

A través de ella los diseñadores pueden crear nuevas clases partiendo de una clase o de una jerarquía de clases preexistente (ya comprobadas y verificadas) evitando con ello el rediseño, la modificación y verificación de la parte ya implementada. La herencia facilita la creación de objetos a partir de otros ya existentes e implica que una subclase obtiene todo el comportamiento (métodos) y eventualmente los atributos (variables) de su superclase.

La herencia es uno de los mecanismos de los lenguajes de programación orientada a objetos basados en clases, por medio del cual una clase se deriva de otra de manera que extiende su funcionalidad. La clase de la que se hereda se suele denominar clase base, clase padre, superclase, clase ancestro, etc.

**Sobrecarga**

Es la capacidad de un lenguaje de programación, que permite nombrar con el mismo identificador diferentes variables u operaciones.

En programación orientada a objetos la sobrecarga se refiere a la posibilidad de tener dos o más funciones con el mismo nombre pero funcionalidad diferente. Es decir, dos o más funciones con el mismo nombre realizan acciones diferentes. El compilador usará una u otra dependiendo de los parámetros usados. A esto se llama también sobrecarga de funciones.

También existe la sobrecarga de operadores que al igual que con la sobrecarga de funciones se le da más de una implementación a un operador.

**Shadowing**

Se le llama “shadowing” al efecto de llevar el mismo nombre a una variable de clase con una variable local definida, donde el acceso o instancia de la variable puede cambiar de resultado cuando se lleva a cabo la ejecución en orden.

Ciclo de vida de:

**Variables de clase:** Se crean cuando la clase se usa por primera vez y suelen existir para el resto del programa a menos que no esté cargado.

**Variables de método:** Existen desde el punto de definición hasta el final del bloque, solo existen dentro del propio método y no se da acceso a ningún otro método.

**Variables estáticas:** Su tiempo de vida se extiende hasta finalizar el programa.