Manuel Arturo Chinchilla Sánchez - 2013009344

Resumen #2

Object Oriented Programming / Advance Concepts - Tutorial C++

Páginas: 86 - 133

Una clase es un concepto expandido de una estructura de datos, en la cual se pueden almacenar no solo datos, sino también funciones. Un objeto es la instanciación de una clase, en términos de variables, una clase es un tipo, y un objeto sería la variable. Cuando declaramos clases se sigue el siguiente formato

class class_name{ //Cuerpo de la clase }

Donde la palabra class, es una palabra reservada en la mayoría de lenguajes de programación, que aceptan el paradigma de POO, para declarar que va a crear una clase.

Además una clase puede tener una función especial llamada constructor, la cual es llamada automáticamente cuando un nuevo objeto de esa clase es creado, este constructor tiene el mismo nombre que la su clase, y no puede retornar ningún tipo de valor. EL constructor es usado generalmente para inicializar variables o asignar memoria dinámicamente durante su proceso de creación.

Al igual que con las estructuras de datos, podemos asignar punteros a objetos de clase, con esto, podemos referirnos directamente a objeto apuntado.

La palabra reservada *this* representa un puntero al objeto cuya función está siendo ejecutada. Es un puntero al objeto en sí mismo.

En un principio los miembros privados y protegidos de una clase solo pueden ser accedidos en la misma clase donde son declarados, sin embargo esta regla no aplica para funciones amigas, estas funciones se declaran como tales, esto se hace de la siguiente forma:

void Perro (parámetros){ //Cuerpo de la función }

Perro duplicate (Perro new_param){ //Cuerpo de la función }

Usando la palabra *duplicate*, indicamos que la nueva función es amiga de la primera.

Una característica clave de las clases en C++ es la herencia, la cual permite crear clases derivadas de otras clases, esto para que se incluyan automáticamente los parámetros y funciones de la clase madre, además de los de la clase propia. Por ejemplo podemos crear una clase polígono, y a su vez crear la clase rectángulo y la clase triangulo. Desde la clase polígono podemos heredar parámetros como perímetro, área, y funciones para sacar las mismas; en las clases heredadas

podemos conservar estos tal y como se declararon en la clase madre, o podemos reescribirlos para adaptarlos a las necesidades.

En C++ es posible que una clase herede miembros de más de una clase, a este mecanismo se le llama Herencia Múltiple, y solo es necesario separar las diferentes clases mediante una coma.

class CRectangle: public CPolygon, public COutput { //Cuerpo de la clase };

Un miembro de una clase que puede ser redefinido en sus clases derivadas se conoce como un miembro virtual. Para declarar un miembro de una clase como virtual, debemos proceder a su declaración con la palabra reservada virtual.

Existen clases de base abstractas, las cuales tienen una funcionalidad mínima, estas clases simplemente sirven como base para nuevas clases.

Las plantillas de funciones son funciones especiales que pueden operar con tipos genéricos, esto nos permite crear una plantilla de función que puede ser adaptada a más de un tipo de dato o clase, repitiendo la mayor parte del código para cada tipo.

También podemos escribir plantillas de clases, de modo que una clase puede tener miembros que utilizan los parámetros de la plantilla así como los tipos.

En C++ tenemos la posibilidad de usar excepciones. Las excepciones son una manera de controlar circunstancias inesperadas, como en tiempo de ejecución, transfiriendo el control a funciones especiales llamados *“handlers”.*

Para atrapar una excepción debemos poner un trozo de código bajo la excepción. La función que queremos inspeccionar (que podría generar un error) la colocamos entre un *“try block”* y luego declaramos un *“catch”*, el cual se encargará de manejar la excepción. Además podemos crear nuestras propias excepciones.

Una técnica muy utilizada en la programación es el casteo, el cual consiste en convertir un tipo de dato en otro tipo. Esto se puede hacer de diferentes formas, las principales son:

Conversión implícita	Conversión explícita

short a = 10;
int b;
b = a;
	short a = 10;
	int b;
	b = int (a);
	b = (int) a;