TP N°12 – Sobrecarga de Metodos – Colaboracion y Herencia

1. Los modificadores de acceso son aquellas declaraciones que indican quien puede acceder a los atributos y/o métodos de una clase en cuestión.

Entre los distintos modificadores se encuentran; private (solo accesible para la clase que tiene esas propiedades), public (accesible desde dentro y fuera de la clase) y protected (es como private, solo que pueden acceder las clases hijas de la clase padre que contenga un atributo o método protegido). El modificador predeterminado es private.

Para utilizarlos simplemente tenemos que escribir el modificador junto con dos puntos, y en las líneas restantes, escribir los atributos y métodos que querramos que tengan ese modificador de acceso:

public:

int n1,n2;

1. Un constructor es un método de clase predefinido que se ejecuta automáticamente cuando un objeto es creado, lo podemos usar para instanciar o inicializar un objeto de una clase, para definirlo, es como un método normal, solo que el constructor debe tener el mismo nombre que la clase en cuestión:

class Alumno {

string nombre;

public:

Alumno(){

cin>>nombre;

}

};

1. Un destructor es un método de clase predefinido que se ejecuta automáticamente cuando un objeto es eliminado, esto ocurre principalmente cuando finaliza el programa. El destructor se suele usar para mostrar información sobre el objeto cuando finaliza el programa, para definir un destructor, es casi lo mismo que con el constructor, solo que se escribe una virgulilla o tilde al principio (~):

class Alumno{

string nombre;

public:

~Alumno(){

cout<<nombre;

}

1. Los atributos private de una clase son aquellos a los que solo se puede acceder o manipular dentro de la misma clase, y los public son aquellos que pueden ser utilizados o manipulados desde cualquier lado, dentro de la clase, fuera, o incluso desde otra clase.

Para incluir los dos modificadores dentro de una clase, escribimos el modificador, dos puntos y a continuación ingresamos los atributos que tendrán ese modificador:

class Alumno{

private:

int n1,n2;

public:

int getPrimerNumero(){return n1;}

};

1. La sobrecarga de métodos comprende la acción de nombrar varios métodos con el mismo nombre, la sobrecarga de métodos nos ayuda a no estar obligados o sometidos a nombrar cada método de manera distinta, es decir, si tenemos 20 metodos, gracias a la sobrecarga de métodos, podemos saber que por lo menos 5 tienen el mismo nombre y cumplen funciones similares.

Para usar la sobrecarga, debemos definir varios métodos con un mismo nombre, pero con distinta cantidad de parámetros o distintos tipos de datos:

class X{

public:

int numeros(int a, int b, int c){

return a+b+c;

}

int numeros(int a, int b){

return a-b;

}

};

1. La colaboración de clases es la acción de que dos o mas clases interactúen entre si, por ejemplo cuando una clase tiene como atributos los objetos de otra clase, puede servir para poder manipular objetos mas fácil con los métodos o atributos privados de otra clase.

Asi se implementa la colaboración de clases:

class X{

private:

int n1,n2;

public:

int numeros(){

return n1+n2;

}

};

class Y{

private:

X obj1,obj2,obj3;

public:

int num(){

int a=obj1.numeros();

int b=obj2.numeros();

int c=obj3.numeros();

return a+b-c;

}

};

1. La herencia de clases es cuando nosotros creamos una clase a partir de otra clase, es decir que a partir de una clase base (clase “padre”) creamos una clase secundaria rama de esta (clase “hija”) que heredara todos los atributos y métodos públicos y protegidos de la clase base.

Son tres los tipos de herencia según niveles de acceso:

•La herencia publica es cuando todos los elementos públicos y protegidos mantienen el mismo nivel de acceso en la clase hija.

•La herencia protegida es cuando todos los elementos públicos de la clase padre se convierten a elementos protegidos en las clases hijas

•La herencia privada es cuando todos los elementos públicos y protegidos adquieren el nivel de acceso privado en la clase hija

Y existen 5 tipos de herencia según su estructura de jerarquía:

•La herencia simple, la herencia mas común, cuando una clase es derivada de una única clase base.

•La herencia multiple, cuando la clase deriva de varias clases bases.

•La herencia multinivel, cuando la clase deriva de una clase derivada de la clase base, como si fueran distintos niveles de clases.

•La herencia jerarquica, cuando varias clases son hijas de una misma clase padre.

•La herencia hibrida, cuando se combinan los distintos tipos de herencia en una sola.

La herencia de clases puede servir para poder crear clases similares sin escribir lo mismo varias veces, por ejemplo un subtipo de algo, haciendo que herede todas las similitudes con el tipo principal. La desventaja principal de la herencia de clases es que las subclases dependen completamente de la clase principal, por lo que las subclases son mas difíciles de reutilizar.

Para implementar la herencia de clases simplemente hay que especificar con “:” a que clase pertenece la subclase que estamos creando:

class Padre{

};

Class Hijo : public Padre{

};

1. La diferencia entre la colaboración y la herencia es que en la colaboración simplemente una clase manipula los objetos de otra clase como atributos propios, mientras que la herencia hereda los atributos, manipulando sus propios objetos.

B. 1,2) El código compila correctamente, y muestra: unaVar = 50