Programación Orientada a Objetos en C++

POO

El paradigma de programación orientada a objetos es una metodología especial de programación de software en la que un programa informático se diseña como la interrelación entre un conjunto de instancias conocidas como objetos. Dichos objetos son entidades que tienen un estado que está descrito por sus variables internas que se conocen como atributos o propiedades; y que tienen comportamientos que están plasmados en funciones o métodos que se utilizan para manipular las variables que describen el estado del objeto.

POO

La programación orientada a objetos al igual que la programación estructurada sigue el método de programación imperativa, es decir, el programador escribe los métodos de forma que a través de un algoritmo se dicta la funcionalidad de los métodos.

Los objetos que tienen estado y comportamientos idénticos se agrupan o clasifican en "clases".

La programación orientada a objetos ofrece los conceptos de encapsulamiento, abstracción, herencia y polimorfismo.

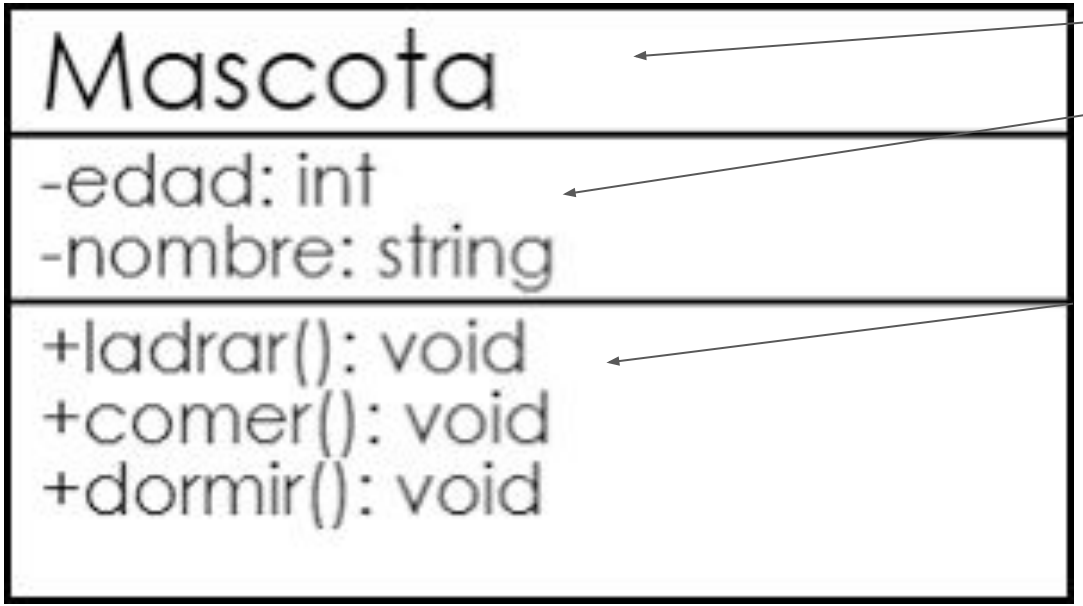
Clases

Una clase es en general un modelo, receta o plantilla que define el estado y comportamiento de cierto tipo de objetos. Una clase puede pensarse como una colección de variables (atributos o propiedades) y funciones (métodos) que permiten representar un conjunto de datos y especificar las operaciones o procedimientos que permiten manipular tales datos. Se puede inclusive entender una clase como un tipo de dato personalizado, similar a las estructuras (structs), donde cada programador define los miembros que va a tener su tipo de dato. De hecho, los tipos de datos nativos de C++ son en clases de realidad.

Objetos

Un objeto es una instancia de una clase, es decir una entidad que se construye a partir de las descripciones consignadas en una clase (datos y funciones). Por tanto, un objeto se puede entender como una "variable" que se declara del tipo de dato de cierta clase. Un objeto es como tal la entidad tangible que permite acceder a los datos y funciones modeladas al interior de la clase

Diagrama de Clase tipo UML

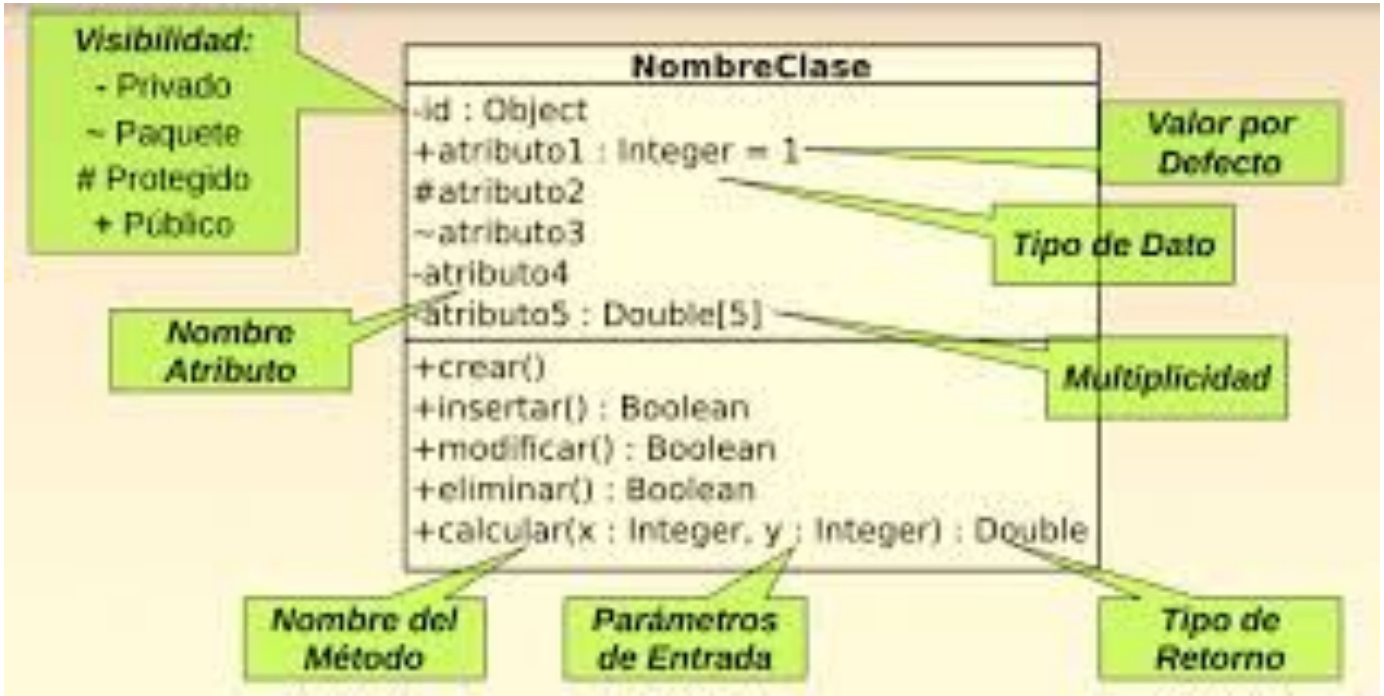


Nombre de Clase

Variables internas conocidas como atributos o propiedades.

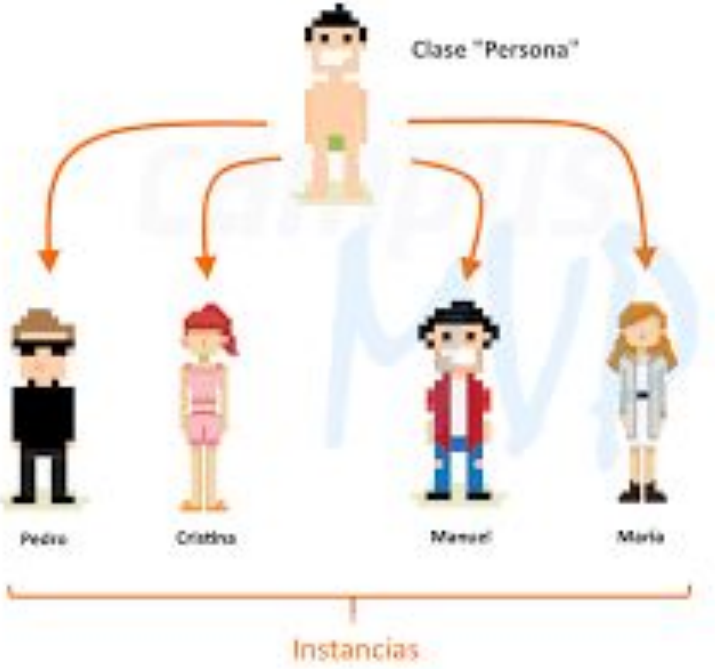
Métodos son las funciones asociadas a la clase.

Diagrama de Clase tipo UML



Objeto Persona

Instancias de Clase = Objeto



Ejemplo


PLATILLOS
Características (atributos):
- Porción.
- Precio.
- Cantidad.
Comportamientos (métodos):
- Servir.
- Vender.
- Reservar.

Ejemplo:

	Mundo Real	**En OOP**
Clase Generalización de características (atributos y comportamientos)	**Perro** Raza, Color, Edad, Corre,	**Clase** Define datos y métodos
Objeto Instancia de una clase distinguible por sus características específicas	**Tino** Pastor Alemán Marrón 7 meses Veloz	**Objetos** Ocupa espacio, se crea y se destruye

Clases

Para declarar una clase en C++ se utiliza la palabra reservada class, se da un nombre a la clase y luego entre llaves se declaran los miembros de la clase.

Las clases no pueden declararse al interior de funciones, ya que son una definición de un tipo de dato creado por el usuario (programador). En general, las clases se declaran en bibliotecas (librerías) individuales cuyo nombre es reglamentario el mismo nombre de la clase.

Clases

```
class MiClase
{
  //Aquí van los miembros de la clase: Variables y funciones
}; //NO olvidar el ;
```

Objetos

Los objetos, tal como se había mencionado con anterioridad, son variables (instancias) del tipo de dato definido por una clase. Por tanto, los objetos se pueden declarar al interior o por fuera de funciones, tal y como una variable local o global respectivamente. Pueden ser declarados como miembros de otras clases, es decir al interior de otras clases. Luego, para declarar un objeto primero se utiliza el nombre de la clase a la que pertenece el objeto seguido de un nombre para el objeto y de una lista opcional de inicialización entre paréntesis.

Objetos

```
MiClase objetoGlobal;  //Declaración de un objeto global de la clase MiClase


int main()
{
    MiClase objetoLocal; //Declaración de un objeto local de la clase MiClase
}
```

Variables y Métodos

Las variables de clase en C++ son datos de distinto tipo que sirven para describir el estado actual de un objeto de esa clase. Se declaran al interior de una clase de la misma forma en que se declaran variables en una aplicación convencional de C++. Es decir, tipo de dato (calificadores opcionales), nombre para la variable y un valor inicial opcional.

Variables y Métodos

Los métodos de una clase son funciones que sirven para manipular las variables de la clase, de ahí viene la primera característica relevante de la programación orientada a objetos que es el encapsulamiento, ya que en lo posible se va a tratar de que sólo pueda accederse a una variable de clase a través de un método de la clase. Los métodos se declaran y definen de la misma manera que una función cualquiera en una aplicación convencional de C++, dicho de otro modo, en su firma expresan el tipo del valor de retorno, un nombre para el método y una lista de parámetros de entrada. Usualmente se hace la de los métodos al interior de la clase, mientras que la definición se hace por fuera de la clase declaración ayudándose del operador de resolución de ámbito::para indicar que el método que se está definiendo pertenece a la clase en cuestión.

ejemplo

```
class MiClase
{
        int var1; //Variable de clase
        const double var2 = 3.14159;  //Variable de clase

        void cambiarVar1(int a); //Declaración de un método de la clase
        double calcularArea(const double& x, const double& y); //Declaración de un método de la clase
};

void MiClase::cambiarVar1(int a)  //Definición del método por fuera de la clase
{
        var1 = a;
}

double MiClase::calcularArea(const double& x, const double& y) //Definición del método por fuera de la clase
{
        return x*y*var2;
}
```

Encapsulamiento y nivel de acceso a miembros de la clase

El acceso a los miembros de una clase solo puede lograrse a través de una instancia de esa clase, es decir, de un objeto de dicha clase. De modo que para acceder a un miembro en específico de una clase se llama al objeto recién nacido y con ayuda del operador punto .se hace el llamado a la variable o método al cual se requiere acceder.

Encapsulamiento y nivel de acceso a miembros de la clase

```
int main()
{
    MiClase miObjeto; //Declarando un objeto de la clase


    miObjeto.cambiarVar1(5); //Accediendo a un miembro con el operador punto
    double var = miObjeto.calcularArea(34.6, 23.9); //Accediendo a un miembro con el operador punto
}
```

Acceso público (public) y acceso privado (private)

De momento se va a hacer énfasis en los niveles de acceso público y privado. El nivel de acceso público se expresa en la declaración de la clase con la palabra reservada public y permite que un miembro de clase sea accedido directamente a través del operador punto, reglamentariamente los métodos de una clase tienen este nivel de acceso. Por otro lado, el nivel de acceso privado permite la ocultación de ciertos miembros de la clase y restringe el acceso a dichos miembros a solo otros miembros de la misma clase, es decir solo puede accederse a un miembro privado de una clase mediante un miembro público que acceda directamente a ese miembro. En este caso, al hacer un llamado directo del miembro privado con el operador punto resultará en un error de compilación de la aplicación. EL nivel de acceso privado es el nivel de acceso por defecto de los miembros de una clase en C++ y se expresa con el uso de la palabra reservada private.

Acceso público (public) y acceso privado (private)

Por otro lado, el nivel de acceso privado permite la ocultación de ciertos miembros de la clase y restringe el acceso a dichos miembros a solo otros miembros de la misma clase, es decir solo puede accederse a un miembro privado de una clase mediante un miembro público que acceda directamente a ese miembro. En este caso, al hacer un llamado directo del miembro privado con el operador punto resultará en un error de compilación de la aplicación. El nivel de acceso privado es el nivel de acceso por defecto de los miembros de una clase en C++ y se expresa con el uso de la palabra reservada private.

Acceso (private-protected-public)

Modifier	Class	Subclass	World
Private	Y	N	N
Protected	Y	Y	N
Public	Y	Y	Y

ejemplo Parte 1

```
class MiClase
{
        int var1; //Acceso privado por defecto
        const double var2 = 3.14159;  //Acceso privado por defecto

        public: //De aquí en adelante los miembros son públicos a no ser que se exprese lo contrario
        void cambiarVar1(int a); //Acceso público
        double calcularArea(const double& x, const double& y); //Acceso público
};
```

ejemplo Parte 2

```
void MiClase::cambiarVar1(int a)
{  var1 = a; }


double MiClase::calcularArea(const double& x, const double& y)
{ return x*y*var2; }
```

ejemplo Parte 3

```
int main()
{
	MiClase obj;
	obj.cambiarVar1(100); //Ok!
	obj.var1 = 200;	//Error!
	return 0;
}
```

Métodos Get y Set

Los métodos Get y Set son métodos que reglamentariamente se definen para el acceso a una variable privada de la clase, para obtener su estado actual (valor) y para modificarlo respectivamente.

Un método Get es por lo general un método con una sola línea de código que devuelve el valor actual de la variable privada. Tiene valor de retorno del mismo tipo de la variable en cuestión y en general no tiene parámetros de entrada.

Por otra parte, un método Set es un método en el cual por lo menos una de las líneas de código modifica directamente el valor de la variable privada. Es poco habitual que tenga valor de retorno, pero sí debe tener por lo menos un parámetro de entrada para el valor con el que se modificará la variable.

Bibliografía

https://www.codingame.com/playgrounds/50557/clases-y-objetos-en-c-practica-1/clases-y-objetos-en-c

https://www.codingame.com/playgrounds/50557/clases-y-objetos-en-c-practica-1/miembros-de-clase-en-c-variables-y-metodos