Composición

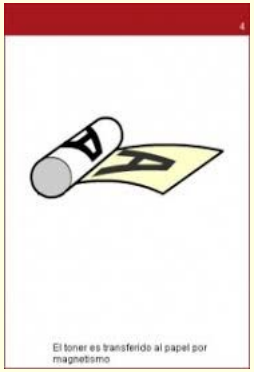
Por:

Yolanda Martínez Treviño

Ma. Guadalupe Roque Díaz de León

El operador de asignación = con objetos

- En C++ se puede utilizar el operador de asignación = para asignar un objeto a otro de la misma clase.
- C++ copia atributo por atributo del objeto de la derecha al objeto de la izquierda.
- A esto se le llama asignación a nivel de miembros.

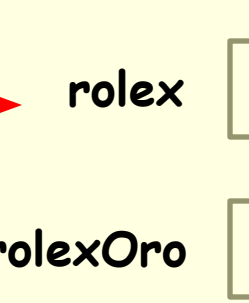


Ejemplo:asignación de objetos

Supón que tenemos la clase Reloj con atributos para horas, minutos y segundos.

Si tenemos las siguientes declaraciones:

Reloj **rolex**{10,30,45},
rolexOro{12,55,2};



	hr	min	seg
rolex	10	30	45

	hr	min	seg
rolexOro	12	55	2

Asignación de objetos:

rolexOro = **rolex**;

Resultado

	hr	min	seg
rolex	10	30	45

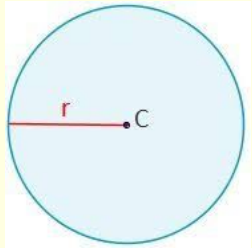
	hr	min	seg
rolexOro	10	30	45

Composición

- Se llama composición cuando una clase tiene algun(os) atributo(s) de otra Clase.

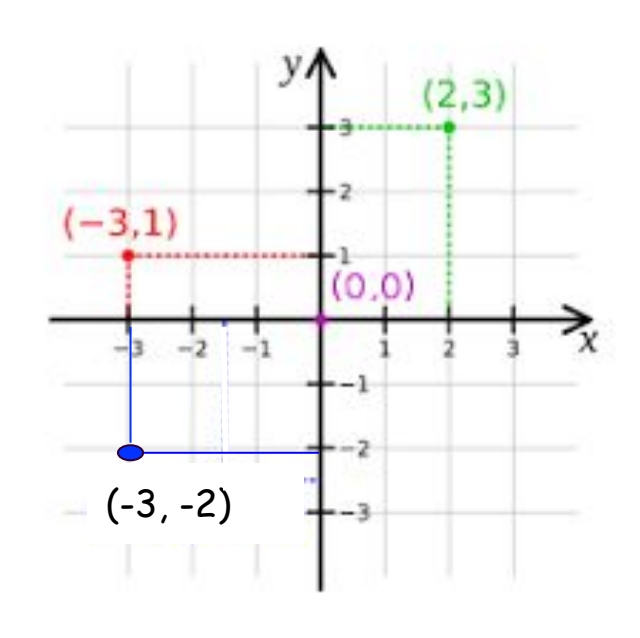
- Ejemplos:
  - La clase Evento requiere tener como atributos: hora de inicio, duración y asunto. Se podría definir el atributo hora de inicio de la clase Reloj - usando la clase Reloj.

  - La clase Círculo, requiere tener como atributos las coordenadas de su centro y su radio. Se podrían definir las coordenadas del centro de tipo Coordenada - usando la clase Coordenada.



Ejemplo de Composición

Clase Coordenada



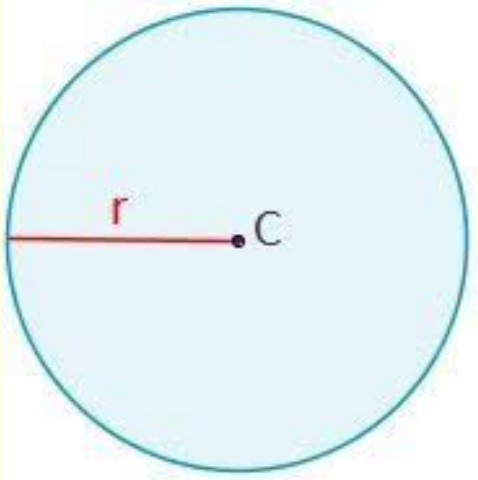
Clase:
Coordenada
Atributos:
- int iX
- int iY

Métodos:
+void setX(int)
+void setY(int)
+int getX( )
+int getY( )
+string str( )

Ejemplo de Composición

Clase **Circulo**

- Clase: **Circulo**
- Atributos:
  - Coordenada centro
  - int radio
- Métodos:
  - + void setCentro(Coordenada)
  - + void setRadio(int)
  - + Coordenada getCentro( )
  - + int getRadio( )
  - + string str( )



Composición- Declaraciones Clases Coordenada y Circulo

Coordenada.h

```
class Coordenada
{
  private:
        int iX;
        int iY;
   public:
        Coordenada();
        Coordenada(int, int);
        void setX(int);
        void setY(int);
        int getX();
        int getY();
        string str();
};
```

Circulo.h

```
#include "Coordenada.h"
class Circulo
{
   private:
        Coordenada centro;
        int iRadio;
   public:
        Circulo();
        Circulo(Coordenada, int);
        void setCentro(Coordenada);
        void setRadio(int);
        Coordenada getCentro();
        int getRadio();
        string str();
};
```

Definición - codificación Clase **Coordenada**:

```
#include <iostream>
using namespace std;
#include "Coordenada.h"

Coordenada :: Coordenada()
{       iX = 0;
        iY = 0;
}

Coordenada :: Coordenada(int _iX, int _iY)
{       iX = _iX;
        iY = _iY;
}

void Coordenada ::setX(int _iX)
{       iX = _iX;
}
```

```
void Coordenada ::setY(int _iY)
{       iY = _iY;
}

int Coordenada :: getX()
{       return iX;
}

int Coordenada :: getY()
{       return iY;
}
//http://www.cplusplus.com/reference/string/to_string/
string Coordenada :: str()
{
return '(' + to_string(iX) + ',' +
            to_string(iY) + ')';
}
```

Grabar en archivo - Coordenada.cpp

Definición - codificación Clase **Circulo**

```
#include <iostream>
using namespace std;
#include "Circulo.h"

Circulo::Circulo()
{ // Inicializar los atributos
   Coordenada  c(1,1);
   centro = c;
   iRadio = 1;
}

Circulo::Circulo(Coordenada cen, int valorRadio)
{
   centro = cen;
   iRadio = valorRadio;
}

void Circulo::setCentro(Coordenada c)
{
   centro = c;
}
```

```
void Circulo::setRadio(int valorRadio)
{
   iRadio = valorRadio;
}

Coordenada Circulo::getCentro()
{
   return centro;
}

int Circulo::getRadio(){
   return iRadio;
}

string Circulo::str(){
return  "Centro:" + centro.str( ) +
", Radio:" + to_string(iRadio);
}
```

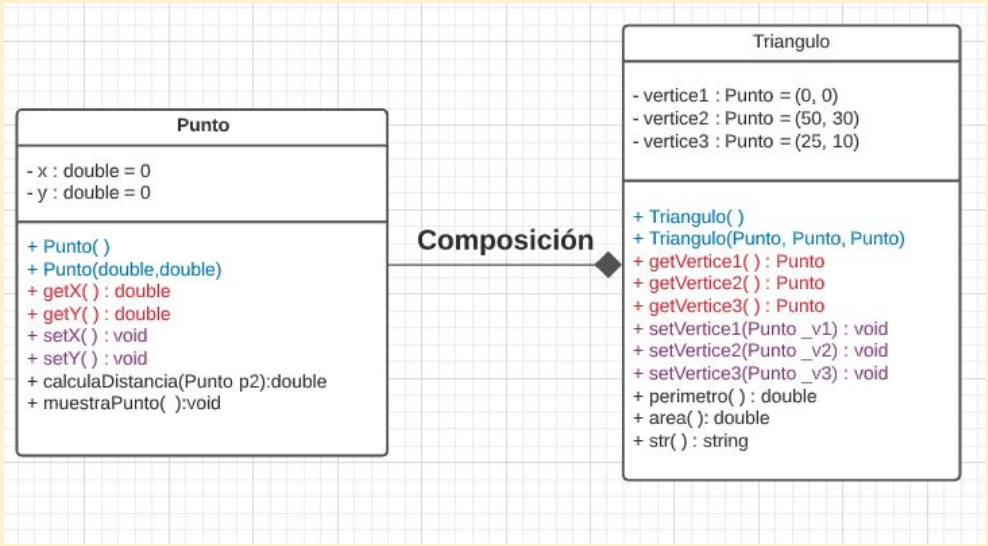
grabar en archivo - Circulo.cpp

Aplicación que usa objetos de tipo Coordenada y Círculo

```
#include "Circulo.h"
#include <iostream>
using namespace std;
int main()
{
// Declaración de objetos
Coordenada coor1(10,5), coor2(20,15), coor3;
Circulo c1(coor2,10), c2;

 // Desplegar los valores actuales de los objetos usando el método str( ) de cada clase.
  cout << "Los datos de la coordenada coor1 son " << coor1.str( ) << endl;
  cout << "Los datos del circulo: c1 son " << c1.str( ) << endl;
  cout << "Los datos del circulo: c2 son " << c2.str( ) << endl;

 // Desplegar los valores actuales  del objeto c2 usando los métodos get de cada clase.
  cout << "Los datos del circulo: c2 son ";
  coor3 = c2.getCentro( );
  cout << "Centro : (" << coor3.getX( ) << "," << coor3.getY( ) << "), Radio : "<< c2.getRadio( ) << endl;

 // Añade las instrucciones para desplegar la coor1 usando los métodos get de la clase
  return 0;
}
```

Ejercicio – clase Evento

- Escribe la clase **Evento** que tenga como atributos
  - inicio - de clase **Reloj**
  - duración - de tipo int, representa los minutos que dura
  - asunto - de tipo string
  - Agrega un constructor default y los método de acceso y modificadores para cada atributo.
  - Agrega el método str( ) que retorne un string con todos los atributos del objeto.
- Diseña una aplicación que permita dar de alta 3 eventos y luego los despliegue en la pantalla.



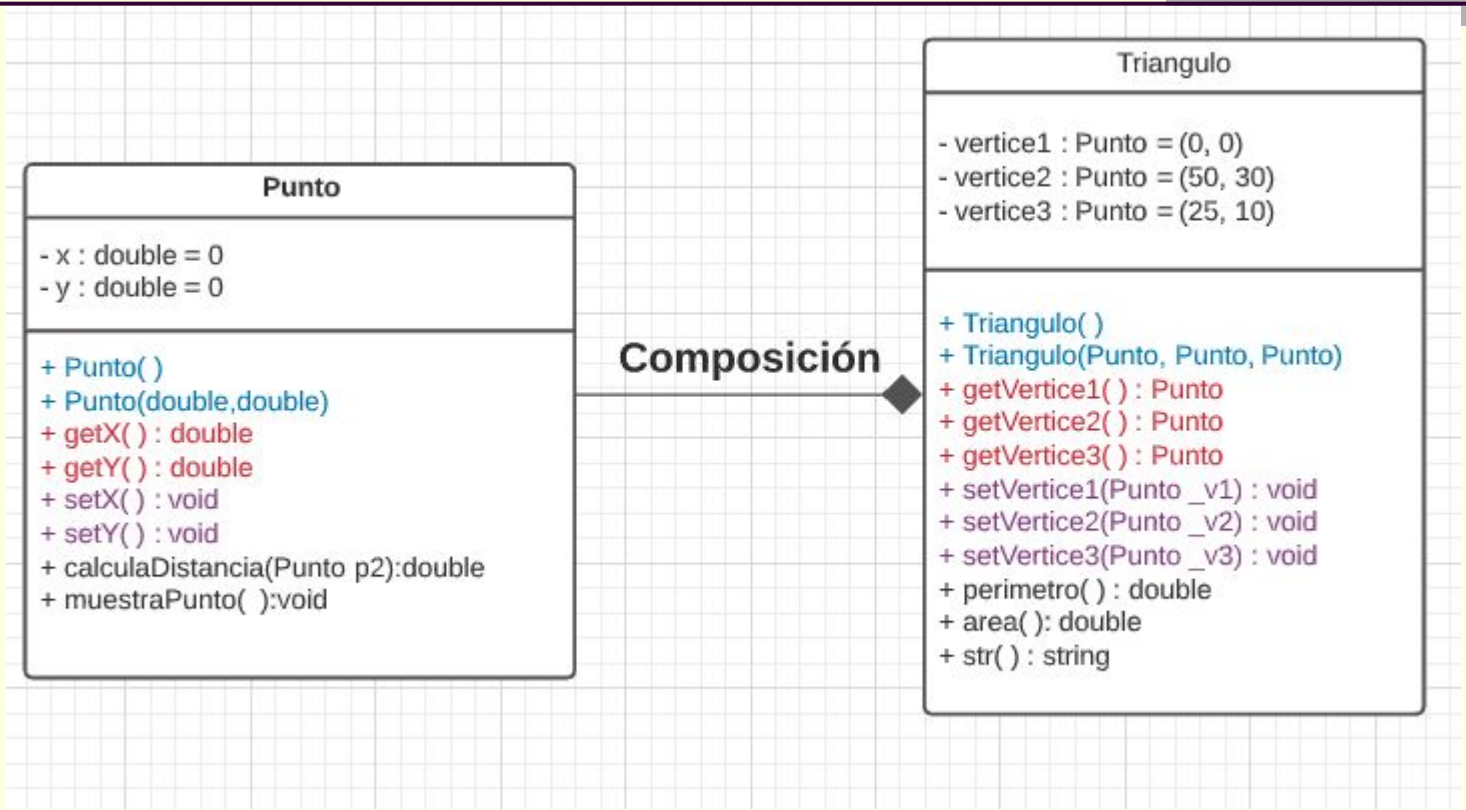
Composición
Tarea - Actividad

Ing. Ma. Guadalupe Roque Díaz de León

Composición

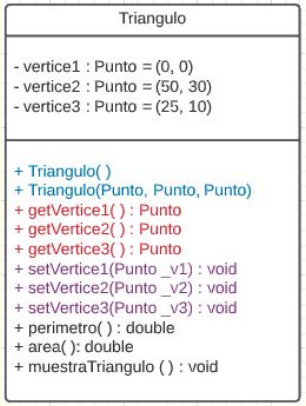
- Se llama composición cuando una Clase tiene algun(os) atributo(s) de otra Clase.
- La Composición es un tipo de relación de alto grado de dependencia entre la clase contenedora (Triangulo) y las clases que la componen (Punto), cuando se crea una instancia de la clase contenedora, deben crearse, instancias de los objetos que la componen, no tendría utilidad, desde el punto de vista de la instancia de Triangulo, crear una instancia sin vertices;
- Durante toda la vida del objeto de la clase Triangulo deben existir los objetos de la clase Punto (vertices)

UML Triangulo, Punto



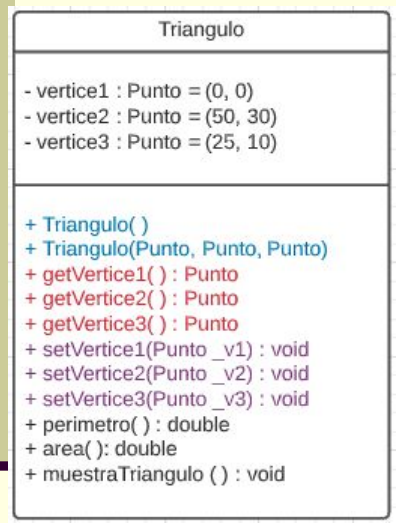
Ejercicio – Clase Triangulo

- Escribe la clase **Triangulo** que tenga como atributos
- Privados
  - vertice1 : Punto
  - vertice2 : Punto
  - vertice3 : Punto
- Público:
  - Constructor default
  - Constructor con parámetros
  - Métodos de acceso para cada atributo.
  - Métodos modificadores para cada atributo.
  - Métodos perimetro, area y str( ) que retorne un string con todos los atributos del objeto.



Ejercicio – clase Triangulo

- Crea la clase **Triangulo, divide en archivos diferentes Triangulo.h y Triangulo.cpp**



Usa las siguientes fórmulas matemáticas para calcular el perímetro y área del triángulo:

Sean tres puntos en el plano cartesiano

$$P_1(X_1, Y_1)\ , P_2(X_2, Y_2)\ \ y\ \ P_2(X_3, Y_3)$$

$$\textbf{PERÍMETRO} = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} + \sqrt{(X_3 - X_2)^2 + (Y_3 - Y_2)^2} + \sqrt{(X_3 - X_1)^2 + (Y_3 - Y_1)^2}$$

$$\textbf{ÁREA} = \frac{1}{2}\{[(X_1 * Y_2) + (X_2 * Y_3) + (X_3 * Y_1)] - [(X_1 * Y_3) + (X_3 * Y_2) + (X_2 * Y_1)]\}$$

calcula la

Crea la aplicación en un archivo main.cpp

Aplicación main.cpp

- Diseña una aplicación que permita crear 2 objetos de la clase Triangulo, uno con los valores default, y otro con los valores ingresados por el usuario, la app debe mostrar en pantalla los datos de los mismos, y calcular el área y perímetro respectivamente.

- Triangulo tri1{}, tri2{p1,p2,p3};
- Diseña 2 casos de Prueba
  - (-2,-3), (-1,3), (5,-4)
  - perimétro:22.37
  - (-5,-5), (1,3), (4,-6)
  - área: 39