CS1112: Programación II

Unidad 6: Relaciones entre clases

Sesión de Laboratorio - 10A

Profesores:

María Hilda Bermejo mbermejo@utec.edu.pe Jonathan Silva jsilva@utec.edu.pe Jorge Villavicencio jvillavicencio@utec.edu.pe Henry Gallegos hgallegos@utec.edu.pe Ian Paul Brossard ibrossard@utec.edu.pe Jose Chavez jchaveza@utec.edu.pe Wilder Nina wnina@utec.edu.pe José Fiestas jfiestas@utec.edu.pe

Material elaborado por:

Maria Hilda Bermejo





Logro de la sesión:

Al finalizar la sesión, los alumnos diseñan e implementan Programas Orientados a Objeto, estableciendo relaciones entre clases.



Repaso clase 9

Programación Orientada a Objetos



Como se define una clase en C++

Robot

Nombre Color(es) Tamaño Vidas

Moverse Atacar Girar Morir

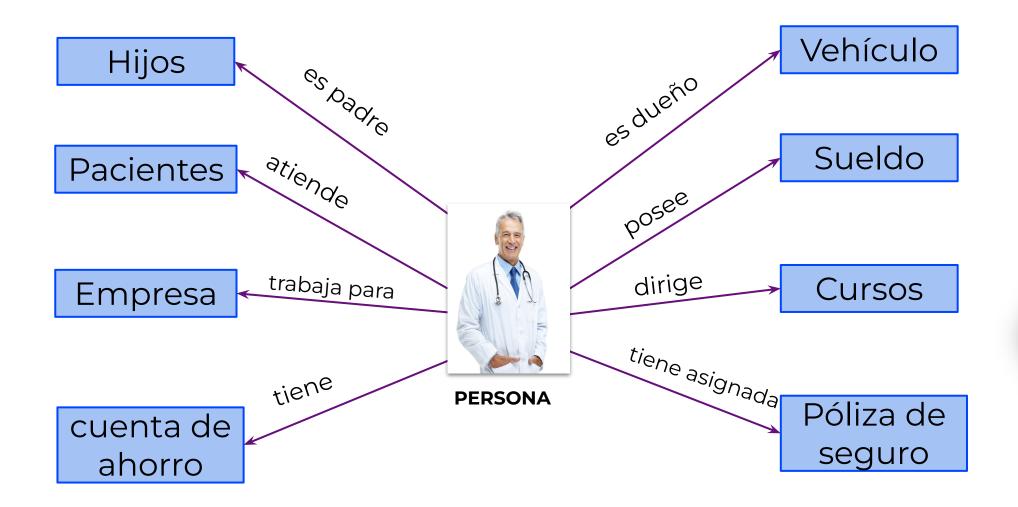
```
class CRobot {
 private:
   string m_Nombre;
   string m_Color;
   int m_Tamanio;
   int m_Vidas
   void Girar();
public:
  float m_PosX;
  float m_PosY;
  void Moverse();
  void Atacar();
```

Con acceso restringido

Con acceso libre



Definición de relaciones





Tipos de relaciones entre clases



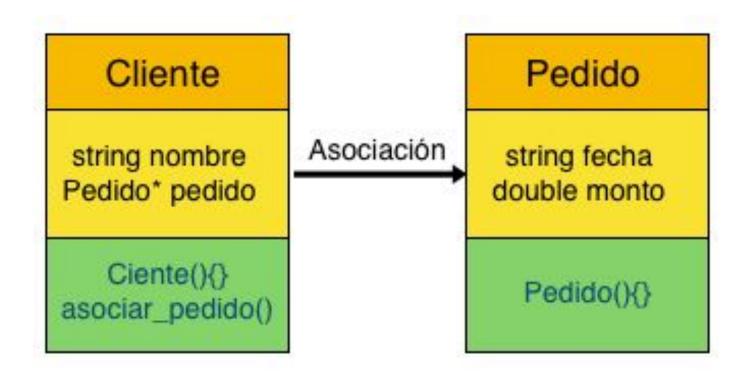
- Agregación
- Asociación
- Herencia





Ejercicio 1

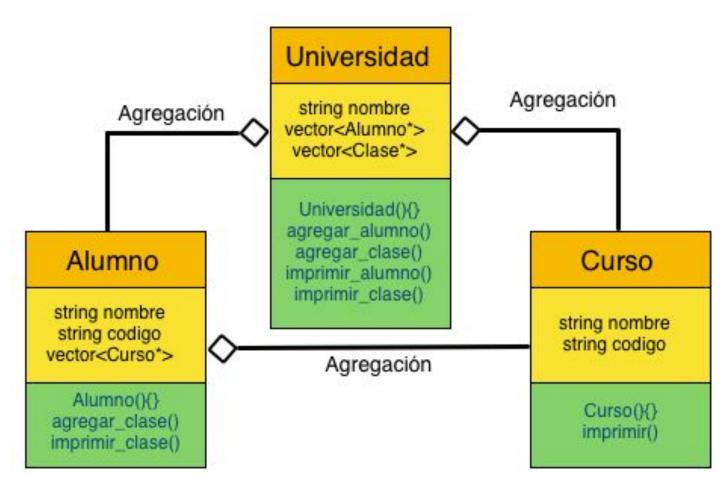
 Generar un Cliente que realice un Pedido con una fecha de entrega y cancele el pedido





Ejercicio 2

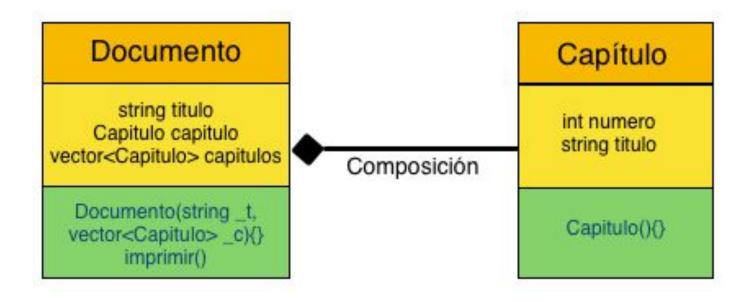
 Objetos de clase Alumno y Curso se agregan a Universidad. Asimismo, objetos de clase Curso se agregan a Alumno





Ejercicio 3

• Generar un Documento con datos de la clase Capitulo





Ejercicio 4.a

- Utilice el código de http://bit.ly/368ix4t
 (punteros) que simula la creación de BATTLEBOT. El programa muestra un tablero y permite alojar, mostrar a Robots. De cada robot, se conoce su nombre, su ubicación (coordenada x, y) y el color.
- Agregue un nuevo método Golpear(), que reciba como parámetro un Objeto al cual va golpear y cuando suceda tal situación quite vidas al objeto golpeado.

Ejemplo:

```
void Robot::golpear(const Robot &enemigo) {
  int averia = (peso >= 35) ? 2 : 1;
  enemigo.setVidas(enemigo.getVidas() - averia);
}
```



Ejercicio 4.b

 Agregue una nueva clase <u>CCoordenada</u> y relacione con la clase <u>CRobot</u>, tal como se muestra en el diagrama de clases

Ojo, debes settear el valor de m_PosX y m_PosY mediante el constructor de CCoordenada



CBattleBox TipoEntero m Altura; TipoEntero m_Ancho; TipoEntero m_CantRobots; TipoEntero m CantObstaculos; vector<vector<TipoCaracter>> m Plano; vector<CRobot*> m_Robots; vector<CObstaculo*> m Obstaculos; CBattleBox(); CBattleBox(TipoEntero pAltura, TipoEntero pAncho); virtual ~CBattleBox(); void adicionarRobot(CRobot* pRobot); CRobot* removerRobot(TipoString pNombre); CRobot* buscarRobot(TipoString nombre); void imprimirRobots(); TipoEntero getAltura(); TipoEntero getAncho(); TipoEntero getCantidadObjectos(); void dibujarBattleBox(); void actualizarBattleBox(); **CRobot** TipoString m Nombre; TipoCaracter m_Color; CCoordenada m Coordenada; TipoEntero m Vidas: **CCoordenda** TipoEntero m_PosX; CRobot(): CRobot(TipoString pNombre, TipoCaracter TipoEntero m PosY; pColor, TipoEntero pPosX, TipoEntero pPosY); virtual ~CRobot(); CCoordenada(); void setNombre(TipoString pNombre); virtual ~CCoordenada(); void moverse(TipoEntero pPosX, TipoEntero pPosY); CCoordenada(TipoEntero pPox, TipoEntero void Chocar(CObstaculo &pObstaculo); pPosY); TipoString getNombre(); TipoEntero getPosX(): TipoEntero getPosX(); TipoEntero getPosY(); TipoEntero getPosY(); TipoCaracter getColor(); TipoString mostrarPosicion();

Resumen

En esta sesión aprendimos:

- Relaciones entre clases
 - Asociación
 - Composición
 - Agregación





