Programación Orientada a Objetos

Práctica 4 – Herencia. Clases Product, Computer y Tv.

Herencia

- Nos permite definir una clase como una *especialización* de otra.
  - E.g. “coche” y “moto” como especialización de “vehículo”.
- La nueva clase *hereda* y *extiende* atributos y métodos de la clase base.
- Refleja relaciones del tipo “Clase_A es un/a Clase_B”.
  - En este caso, Clase_A heredaría de Clase_B.

Tipos de herencia

- Pública: Los miembros públicos y protegidos siguen siéndolo en la clase derivada.
  - Se denota con la palabra clave `public`
- Protegida: Los miembros públicos y protegidos pasan a ser protegidos en la clase derivada.
  - Se denota con la palabra clave `protected`
- Protegida: Los miembros públicos y protegidos pasan a ser privados en la clase derivada.
  - Se denota con la palabra clave `private`

Herencia en C++

- ```
  class Class_A : public Class_B
  {
  private:
      <nuevos campos/métodos>
  public:
      <nuevos campos/métodos>
  };
  ```

- La nueva clase (Class_A) contiene todos los miembros de la clase base (Class_B), más los que se añadan en su definición.
  ```

Herencia en C++

- El constructor de la clase derivada pasa en su prototipo los parámetros requeridos al constructor de la clase base:

```
Class_A(tipo1 param_base1, tipo2
param_base2, ..., tipoN param_propio1):
Class_B(param_base1, param_base2, ...)
{
    <código constructor Class_A>
}
```

- No es necesario seguir un “orden” (los parámetros se pasan al constructor de la clase base con su nombre).

Enumerados

- Un enumerado (*enum*) es un tipo de dato que representa un rango restringido de constantes con nombre (*enumeradores*).
  - E.g. cuando necesitamos un valor que represente una “categoría” de objeto, de entre un grupo fijo de categorías.
- En C++ (desde C++11) la forma preferida es el uso de clases enum. A diferencia de los enum de C (constantes enteras), los valores son locales a la enumeración y no se convierten implícitamente a otros tipos.
  - Un enum (no enum class) es *básicamente* un int!

Enumerados

- Para definir una clase enum, podemos hacerlo así:

```
enum class CategoríaObjeto
{
    Categoría1,
    Categoría2,
    Categoría3,
    ...
};
```

- Podemos crear una variable del tipo `CategoríasObjeto` de la siguiente manera:

```
CategoríaObjeto categoría =
CategoríaObjeto::Categoría1
```

Ejercicio(s)

- Ejercicio 1: Crear la clase *Product*.
  - Con atributos id, name, price, maker, seller.
  - El constructor recibe los parámetros en ese orden; id es obligatorio.
  - Getters y setters para todos los parámetros.
  - Hacer un programa product-main.cc para probarlo.
- Ejercicio 2: Crear la clase *Tv*.
  - Hereda de la clase *Product*.
  - Añade un atributo “inch_” (tamaño del televisor).
  - Getter y setter para el nuevo atributo.
  - Hacer un programa tv-main.cc para probarlo.

Ejercicio(s)

- Ejercicio 3: Crear la clase *Computer*.
  - Hereda de la clase *Product*.
  - Añade un atributo “type_” (tipo de ordenador).
    - Implementar usando enum class.
  - Getter y setter para el nuevo atributo.
  - Hacer un programa computer-main.cc para probarlo.

- Ejercicio 4: Testing.
  - Compilar y ejecutar todos los programas main.
  - Ejecutar todos los test (código disponible en Moodle); corregir errores hasta que pasen.