



Examen de Programación Orientada a Objetos

EJERCICIOS

1.- (test1) La clase **Computer** tiene dos campos de tipo *string* "nombre" y "modelo", y un campo de tipo entero "precio". Codifica el fichero *computer.h* y *computer.cc* de la clase **Computer** con la siguiente funcionalidad:

- Constructor con 3 parámetros. El nombre será el primer parámetro obligatorio, el modelo será el segundo parámetro obligatorio. El precio como tercer parámetro y con valor por defecto igual a 0. Si "nombre" o "modelo" reciben la cadena vacía, se asignará su respectivo valor a "Ninguno". Si el precio pasado es menor que 0, se asignará al precio el valor 0.
- Modificador *setNombre()* que recibe como parámetro el nuevo nombre. Si el nuevo nombre es la cadena vacía, no se hará el cambio y se devolverá false. En caso contrario se hará el cambio y se devolverá *true*.
- Modificador *setModelo()* que recibe como parámetro el nuevo modelo. Si el nuevo modelo es la cadena vacía, no se hará el cambio y se devolverá false. En caso contrario se hará el cambio y se devolverá *true*.
- Modificador *setPrecio()* que recibe como parámetro el nuevo precio que debe ser mayor que cero, en cuyo caso se devuelve *true*. Si es menor o igual que cero no se asigna y se devuelve false.
- Observadores *getPrecio()*, *getModelo()* y *getNombre()*.
- Observador *getNombreYModelo()* que devuelve una cadena con el nombre seguido de una coma y un espacio, y a continuación el modelo.

PUNTUACIÓN: 2 puntos

2.- (test2) La clase **Cart** representa una cesta de la compra de una tienda online y tiene un campo de tipo entero "id" y una lista de objetos de la clase **Computer** del ejercicio anterior (para la lista **usar el tipo list de la STL de C++**). Codifica los ficheros *cart.h* y *cart.cc* para la clase **Cart** con la siguiente funcionalidad:

- a) Constructor que recibe como parámetro obligatorio el id de la compra.
- b) Observador *getId()*.
- c) El método *addComputer()* de tipo *void* que recibe un objeto de tipo **Computer** y lo añade a la lista.
- d) El método *getN()* que devuelve un entero con el tamaño de la lista.
- e) El método *print()*, de tipo *void*, que muestra en pantalla un **Computer** por linea con la posición



numérica, su nombre y su modelo:
1,nombre,modelo
2,nombre,modelo
3,nombre,modelo
...

Durante la ejecución del test2 deberá mostrarse en pantalla (bajo el test Cart.print):

1,PC1,PV1
2,PC2,PV2
3,PC3,PV3

- f) El método `write()`, de tipo `void`, que escriba lo mismo pero en un fichero texto que se llamará “`salida.txt`”. Despues de la ejecución del test2 el contenido del fichero “`salida.txt`” será:

1,PC1,PVX
2,PC2,PVX
3,PC3,PVX

OJO: comprueba que este fichero se crea y no escribas ningún espacio en blanco en él “`salida.txt`”

PUNTUACIÓN: 2 p

→ 4.- (test4) La clase `Laptop` deriva de la clase `Computer` del ejercicio anterior y además gestiona una variable de tipo `int` denominada “pulgadas”. Escribe el código de la clase `Laptop` en los ficheros `laptop.h` y `laptop.cc` con la siguiente funcionalidad:

- Constructor que recibe como parámetro obligatorio el valor para “pulgadas”, el segundo parámetro obligatorio será el modelo del `Laptop`, y el tercer parámetro obligatorio será el precio del `Laptop`. Si las pulgadas recibidas es menor o igual a cero se asignará el valor 1. El constructor además asignará al campo “nombre” de la clase `Computer` el valor “`Laptop`” y el campo “modelo” y “precio” de la clase `Computer` a los valores recibidos como parámetro.
- Observador `getPulgadas()`.
- Modificador `setPulgadas()` que no permita valores menores o iguales que cero. Si el parámetro es menor o igual a 0 debe devolver false y no asignarlo. Si por el contrario es positivo debe asignarlo y devolver true.
- Observador `getInfo()` que devuelve una cadena formada por la concatenación de la información del `Laptop`. Por ejemplo si el `Laptop` es de 18 pulgadas modelo PV3 y el precio son 300, esta función debe devolver la cadena: “`Laptop de 18 pulgadas modelo PV3 precio 300`” (**NOTA:** usa la función `std::to_string()` para pasar convertir las pulgadas a `string`).

PUNTUACIÓN: 2 puntos

5.- (test5) Haz una copia del fichero `computer.h` que se llame `computer2.h`, y otra de `computer.cc` que se llame `computer2.cc` (Recuerda cambiar `#include "computer2.h"` en `computer2.cc`). Añade la siguiente funcionalidad:

- a) Un extractor para la clase `Computer` (operador `>>`) que saque en pantalla el nombre y el modelo del `Computer` de la siguiente forma. Por ejemplo, si el nombre fuese `Computer1` y el modelo `PIV`, el resultado será el siguiente (es el que saldrá en el test `Computer2.Extractor`):

Nombre: Computer1

Modelo: PIV

- b) Un insertador para la clase `Computer` (operador `<<`) que pida por teclado el nombre y el modelo del `Computer` de la siguiente forma:

Introduce nombre: *(el usuario lo introduce desde el teclado)*

Introduce modelo: *(el usuario lo introduce desde el teclado)*