Object Oriented Features

1. Write a simple class called **Simple** with a constructor that prints something to tell you that it’s been called. In **main( )** make an object of your class.
2. Add a destructor to Exercise 1 that prints out a message to tell you that it’s been called.
3. Create a class containing both a **const** and a non-**const float**. Initialize these using the constructor initializer list.
4. Create a class with both **const** and non-**const** member functions. Create **const** and non-**const** objects of this class, and try calling the different types of member functions for the different types of objects. Create a class with both **const** and non-**const** member functions. Try to call a non-**const** member function from a **const** member function
5. Modify Exercise 2 so that the class contains an **int** member. Modify the constructor so that it takes an **int** argument that it stores in the class member. Both the constructor and destructor should print out the **int** value as part of their message, so you can see the objects as they are created and destroyed.
6. Implementar la clase Cellphone que tiene la siguiente información:

* Brand
* Model
* Color

Agregar constructor, destructor, operador asignacion y constructor por copia.

Agregar setters y getters

Implementar la clase CellphoneVendor que debe tener los siguiente métodos:

* addCellphone() que agrega un Cellphone a la lista de teléfonos disponibles (no pueden haber duplicados)
* removeCellphone() que borra un Cellphone de la lista de teléfonos disponibles
* setPrice() que toma como parámetro el precio del Cellphone y el Cellphone, y setea su precio
* getPrice() que toma como parámetro un Cellphone y devuelve su precio. Si el precio no fue seteado para ese Cellphone, devuelve 0.
* addToStock() que toma como parámetro un Cellphone y la cantidad de a agregar al stock
* checkStock() que toma como parámetro un Cellphone y devuelve la cantidad de teléfonos en stock para ese Cellphone
* sell() que toma como parámetro un Cellphone y decrementa el stock en 1 unidad para ese Cellphone. Devuelve true si pudo realizarse la venta o false si no hubo stock
* getBrands() que devuelve una lista de los brands de teléfonos disponibles
* getModels() que devuelve una lista de los models de teléfonos disponibles
* getColors() que devuelve una lista de los colors de teléfonos disponibles

Agregar constructor, destructor, operador asignacion y constructor por copia.

1. Create a class without any constructors, and show that you can create objects with the default constructor. Now create a nondefault constructor (one with an argument) for the class, and try compiling again. Explain what happened
2. class Foo {

public:

Foo();

private:

int a;

int b;

int c;

};

Foo::Foo()

: a(10)

, c(7)

, b(c)

{

}

int main() {

Foo foo;

return 0;

}

Que valor tienen a, b y c en la instancia foo? Explique