**UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA**

**FACULTAD DE INGENIERIA DE PRODUCCION Y SERVICIOS**

**ESCUELA PROFESIONAL DE CIENCIAS DE LA COMPUTACIÓN**

Un dibujo de una cara feliz

Descripción generada automáticamente con confianza baja

**LABORATORIO 20 – Smart Pointers.**

**DOCENTE:**

Enzo Edir Velásquez Lobatón

**ALUMNO:**

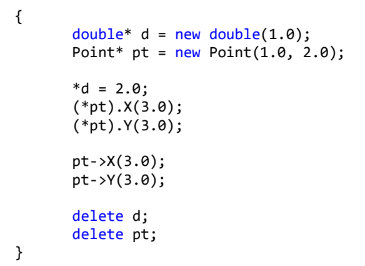
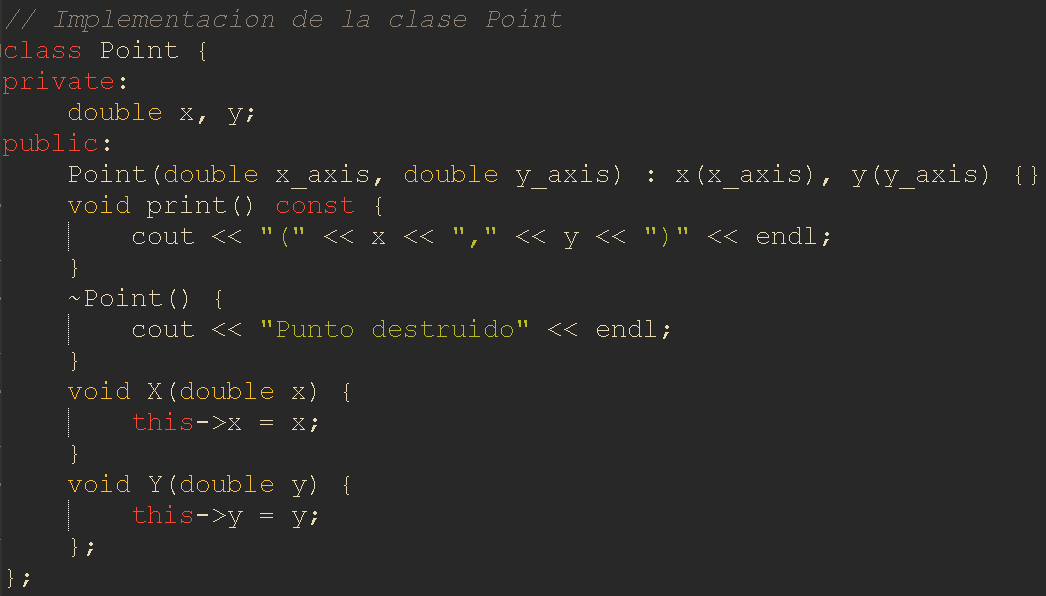
Owen Haziel Roque Sosa.

**FECHA:**

14/08/2022

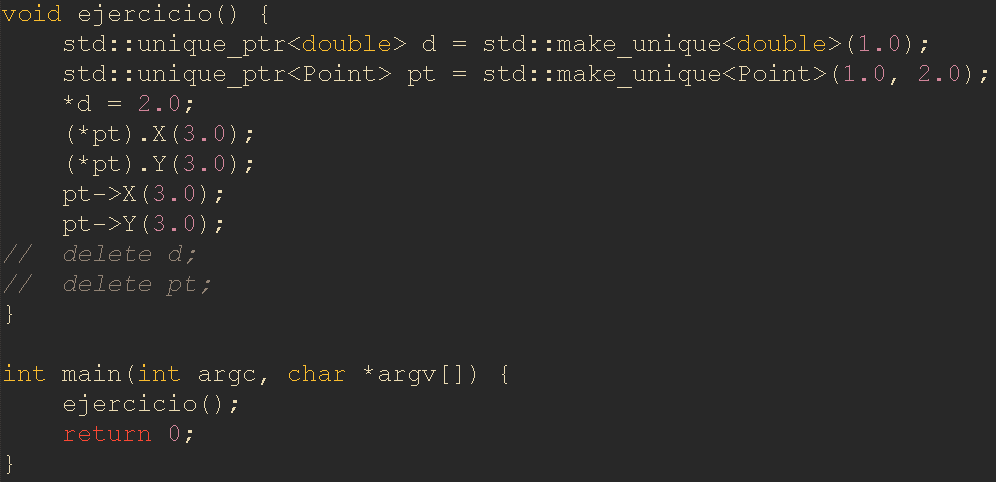
**Arequipa – Perú**

1. Implemente el siguiente código que usa punteros sin procesar y explique lo que hace:

* Se crean punteros a double y a objeto Point, con sus respectivos valores.
* Se hace una dereferencia y se llaman a las funciones miembro (métodos) de la clase Point (o struct, depende).
  + Se modifican las coordenadas X, Y a través del operador de indirección (\*)
* Se modifican las coordenadas X, Y a través del mismo puntero (pt)
* Se limpia la memoria a través de *delete.*

1. Transfiera el código anterior reemplazando los punteros sin formato por std::unique\_ptr.



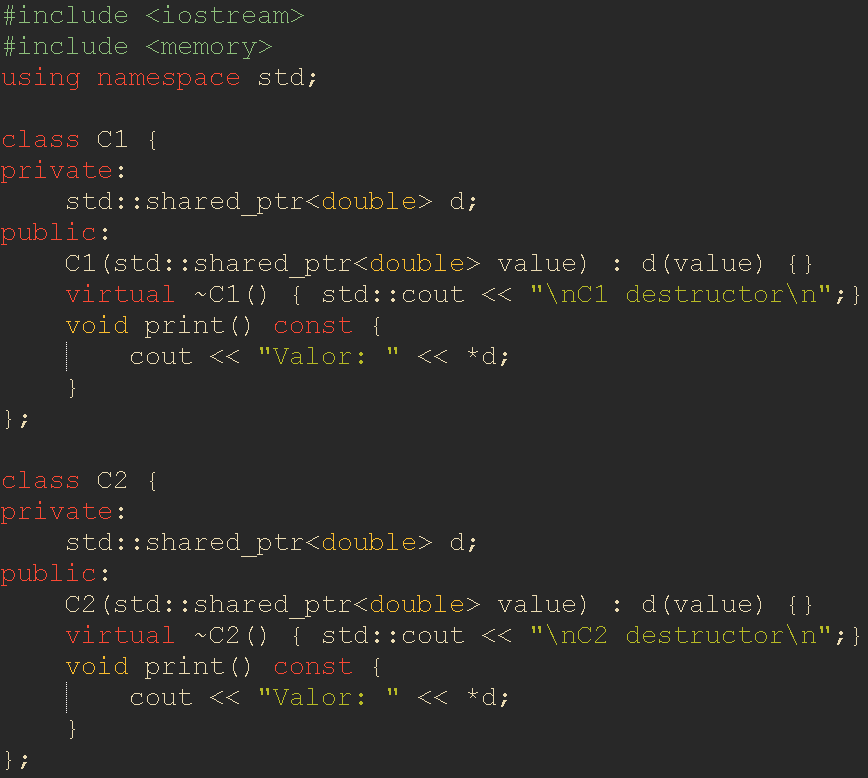
Texto

Descripción generada automáticamente

1. Implementar el código para las clases C1 y C2, cada una de las cuales contiene el objeto compartido d anterior, por ejemplo:

Texto, Carta

Descripción generada automáticamente



1. Transfiera el código anterior reemplazando los punteros sin formato por std::shared\_ptr<Point> p;



Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

1. Al anterior código implemente un puntero débil a un puntero el cual no puede estar vacío.

* Mismas Clases Point, C1, C2 del ejercicio anterior

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Con la función use\_count() de std::weak\_ptr se verifica la cantidad de objetos std::shared\_ptr gestionados por *wp*, que son 3:

* El std::shared\_ptr<Point> p Inicial (definido en main),
* std::shared\_ptr<Point> d de la clase C1,
* std::shared\_ptr<Point> d de la clase C2