


Clases Abstractas

Abstract Class

Clases Abstractas – Conceptos Clave

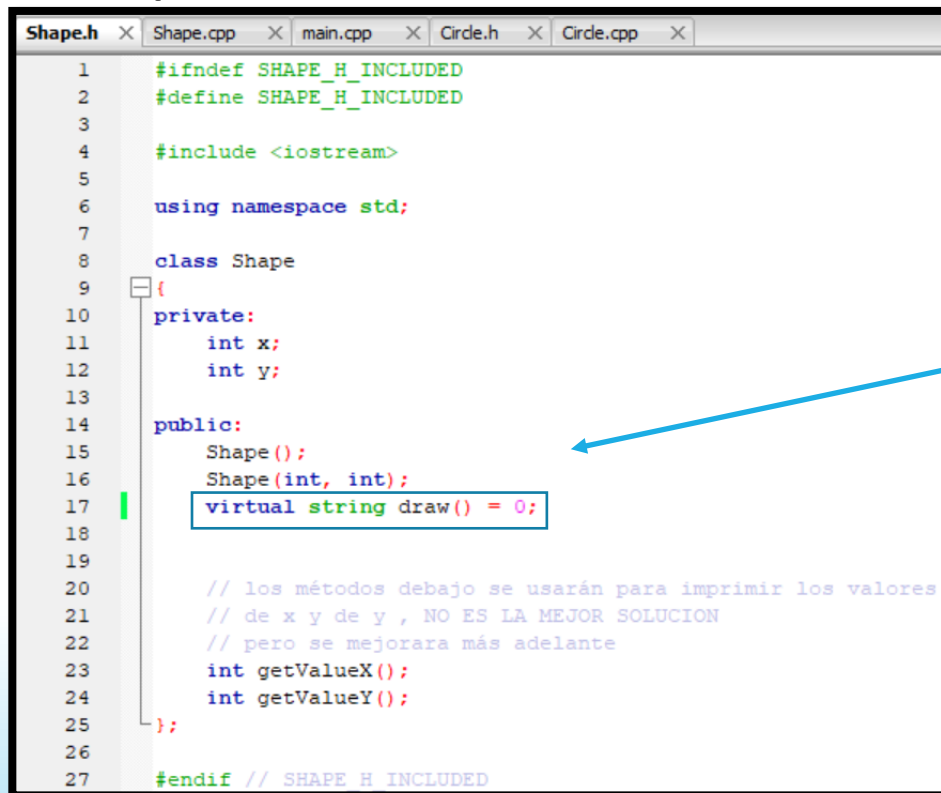
- Clases que no se pueden instanciar
- Generalmente se usan como base para clases hijas
- Para que una clase sea abstracta en C++ debe tener una función virtual pura (virtual y con 0 asignado)



```
class AB {  
public:  
    virtual void f() = 0;  
};
```

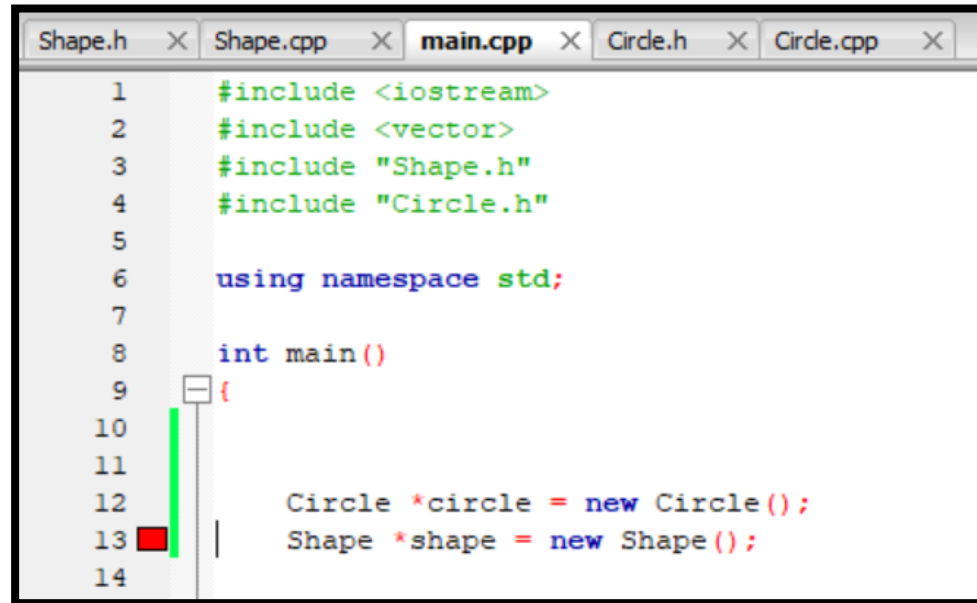
Clases Abstractas – Representación en código

- Usando el mismo ejemplo, el método draw se declarado virtual se iguala a 0 en la clase Shape, lo que hace que no sea posible instanciar Shape



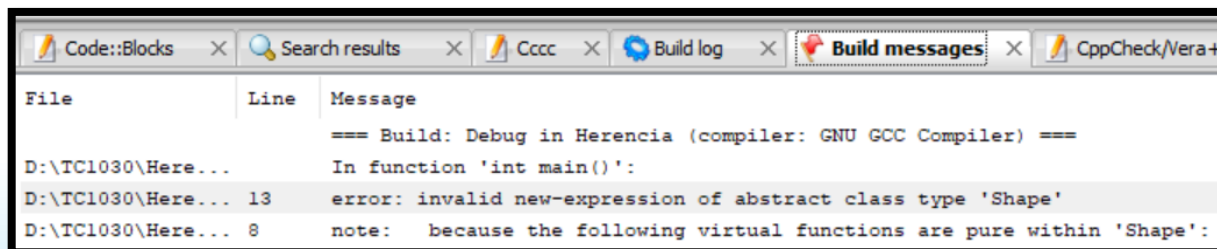
```
1  #ifndef SHAPE_H_INCLUDED
2  #define SHAPE_H_INCLUDED
3
4  #include <iostream>
5
6  using namespace std;
7
8  class Shape
9  {
10 private:
11     int x;
12     int y;
13
14 public:
15     Shape();
16     Shape(int, int);
17     virtual string draw() = 0;
18
19
20     // los métodos debajo se usarán para imprimir los valores
21     // de x y de y , NO ES LA MEJOR SOLUCION
22     // pero se mejorara más adelante
23     int getValueX();
24     int getValueY();
25 };
26
27 #endif // SHAPE_H_INCLUDED
```

Clases Abstractas - Continuación



```
1  #include <iostream>
2  #include <vector>
3  #include "Shape.h"
4  #include "Circle.h"
5
6  using namespace std;
7
8  int main()
9  {
10
11
12     Circle *circle = new Circle();
13     Shape *shape = new Shape();
14 }
```

Notar que si se deseara instanciar la clase Shape no sería posible y marcaría un error de compilación



File	Line	Message
=== Build: Debug in Herencia (compiler: GNU GCC Compiler) ===		
In function 'int main()':		
D:\TC1030\Here...	13	error: invalid new-expression of abstract class type 'Shape'
D:\TC1030\Here...	8	note: because the following virtual functions are pure within 'Shape':

Práctica

- Definir las clases Rectangle y Polygon, ambas heredan de Shape
- Shape debe tener una función virtual Pura. Se recomienda la función area()
- Implementar cómo obtener el área en las clases hijas