Documentación de una clase

una clase se separa en 2 archivos el .h y el .cpp.
en el .h se pone por así decirlo el esqueleto de la clase, lo que es y para lo que sirve en este caso tiene 3 constructores uno explicito y 2 implícitos.
luego tiene un destructor.
después le enseñas a usar el signo igual y Como aceptar valores tipo int.
tiene los setter y getters.
al final tiene el valor guardado.

```
#ifndef INTCELL_H
#define INTCELL_H

class IntCell {
public:
    // explicit constructor, sirve para r valores
    explicit IntCell(int newValue = 0);
    // constructores por copia y movimiento sirve para l valores
    IntCell(const IntCell &rhs);
    IntCell(IntCell &&rhs) noexcept;
    //destructor
    ~IntCell() = default;

    // aqui le enseñas a usar el valor de igual
    IntCell &operator=(const IntCell &rhs);
    IntCell &operator=(IntCell &&rhs) noexcept;

    // Overloaded assignment operator to accept primitive int
    IntCell &operator=(int rhs);

    // asignar y mostrar el valor guardado
    int getValue(int i) const;
    void setValue(int newValue);

private:
    // valor guardado
    int storedValue;
```

```
};
#endif  // INTCELL_H
```

La segunda parte del codigo esta en un archivo .cpp y le dices a la clase como hacer todo lo que le pides en el .h.
lo primero que te muestra son los 3 constructores, el explicito o default, el por copia y por movimiento.
luego te muestra los operadores por asignación.
y por ultimo los getter y setters que con para obtener y mostrar el valor guardado

```
#include "IntCell.h"

// Default constructor
IntCell::IntCell(int newValue) : storedValue(newValue) {}

// Copy constructor
IntCell::IntCell(const IntCell &rhs) : storedValue(rhs.storedValue) {}

// Move constructor
IntCell::IntCell(IntCell &&rhs) noexcept : storedValue(rhs.storedValue) {
   // rhs.storedValue = 0;
}
// Copy assignment operator
IntCell &IntCell::operator=(const IntCell &rhs) {
    if (this != &rhs) {
        storedValue = rhs.storedValue;
    }
    return *this;
}

// Remove Move assignment operator
IntCell &IntCell::operator=(IntCell &&rhs) noexcept {
    if (this != &rhs) {
        storedValue = rhs.storedValue;
        rhs.storedValue = 0;
    }
    return *this;
}

// Overloaded assignment operator to accept only integer data
```

```
IntCell &IntCell::operator=(int rhs) {
    storedValue = rhs;
    return *this;
}
// getters and setters
// getter
int IntCell::getValue(int i) const {
    return storedValue;
}
// setter
void IntCell::setValue(int newValue) {
    storedValue =newValue;
```

así se vería el uso en un main:

```
#include <iostream>
#include "IntCell.h"

int main()
{
    IntCell intCell;
    int x;
    std::cout << " e creado una clase mmlona" << std::endl;
    std::cout << "ingrese un numero" << std::endl;
    std::cin >>  x;
    intCell.setValue(x);

    std::cout << "El valor guardado en InteCell es: "<< intCell.getValue(x)
<<std::endl;

}
```

este código mostraría lo siguiente en consola: