13.5 Focus on Software Engineering: Separating Class Specification from Implementation.

"Usually class declarations are stored in their own header files. Member functions definitions are stored in their own .cpp files."

↳ The header file that contains a class declaration is called a class specification file.

↳ usually nameOfClass.h

↳ The ~~f~~ member function definitions for a class are stored in a separate .cpp file called class implementation file.

↳ usually nameOfClass.cpp.

- #include "file.h"
  When file is in current project directory.
- #include <file.h>
  When file is in the compiler's include file directory.

4 The include file directory is the directory or folder where all of the standard C++ header files are located.

- **Utility Functions:**
  - ↳ Private functions (only to implement logic inside class).
  - ↳ Can't used outside the class.

Example:

```
class Volume{
    int h,w,l,v;
    void calculateVolume(){ v = l*h*w;}
public:
int getVolume (){
        calculateVolume();
        return v;
    }
};
```

Date: __/__/20 M T W T F S

OOP (19/04/22)

Cascaded Function Call:

Student std;

(Computed As) std.fun1().fun2().fun3();

- If std.fun1() returns void.
  void.fun2() will be called **(ERROR)**
- It will work only when fun1() returns an object.
- ~~To~~ Similarly, to ~~to~~ call fun3(), fun2() should also return an object.

Use of 'this'

this is a pointer that contains the address of current obj. (available inside class).

return this; //Returns address of object.

return *this; // Returns the object.

⇒ fun1() & fun2() should be of form:

Good practice to return by reference ← Student & fun1(){

return *this;

}

Object as Return Type

Date: / /20

Another Example of cascaded function call:

$\underline{\underline{std\,4} = \underline{std\,3 = std2}}$;

↳ By default working.

• But When we overwrite = operator as:

```
void operator = (Student &obj){
        gpa = obj.gpa;
}
```

std4 = $\underline{std3 = std2}$;

↳ This will return void (**Thus an ERROR**)

std3 = std2;   **// Works.**

But for (std4 = std3 = std2;), it should return the object.

```
Student & operator = (Student &obj){
        gpa = obj.gpa;
        return *this;
}
```

OOP **(20/04/22)**

Use of 'this' operator:

```
void setValue (int value){
        this -> value = value;
}
```

$\underline{\text{this -> value}}$ ↳ Attribute = $\underline{\text{value}}$ ↳ Parameter.

Date: ___/___/20___ M T W T F S

// Global variable (attributes) have less precedence than local variable (parameter), if we don't write this->value, it will also be considered as parameter value.

e.g (parameter) = (parameter).

& the value of attribute will not be changed.

Objects as Arguments

Passing Object by Value:

```
void fun1(MyClass Obj);
fun1(obj);
```

Passing Object by reference:

```
void fun1(MyClass &Obj);
fun1(obj);
```

Passing Object by pointers:

```
void fun1(MyClass *Obj);
fun1(&obj);
```