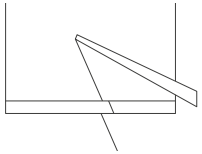
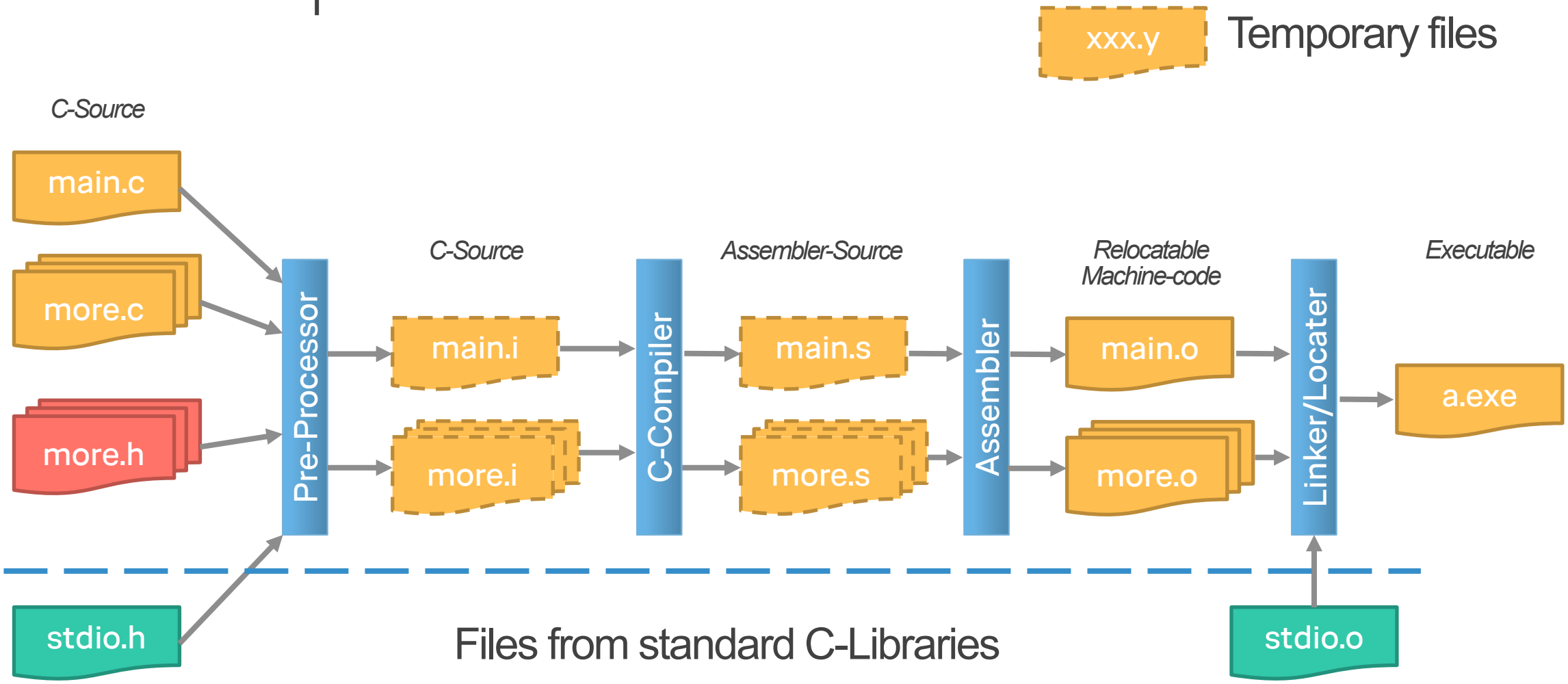
Bring ideas to life
VIA University College



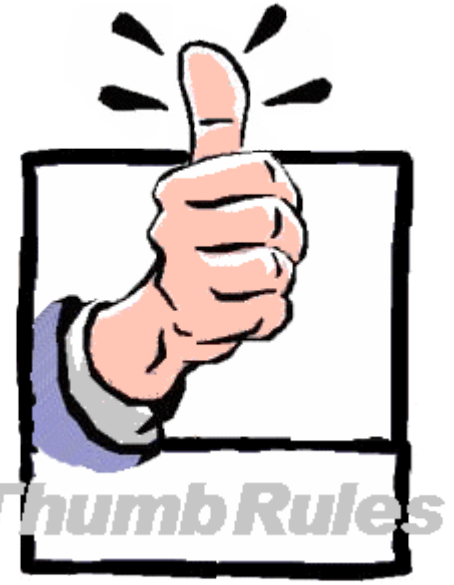
C Program Structure

ESW1

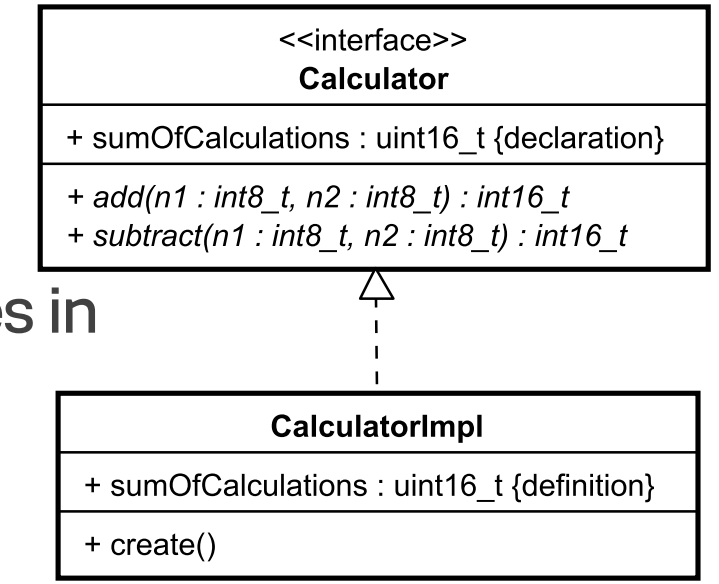
The Compilation Process



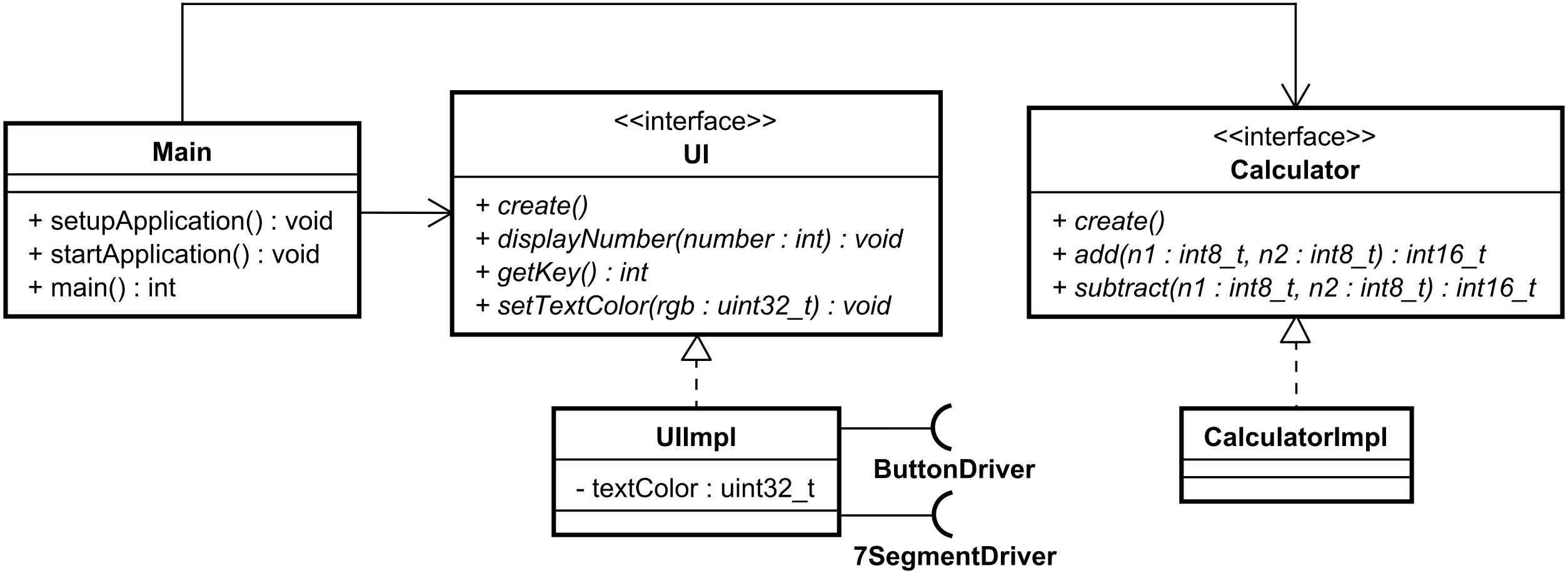
Source Files



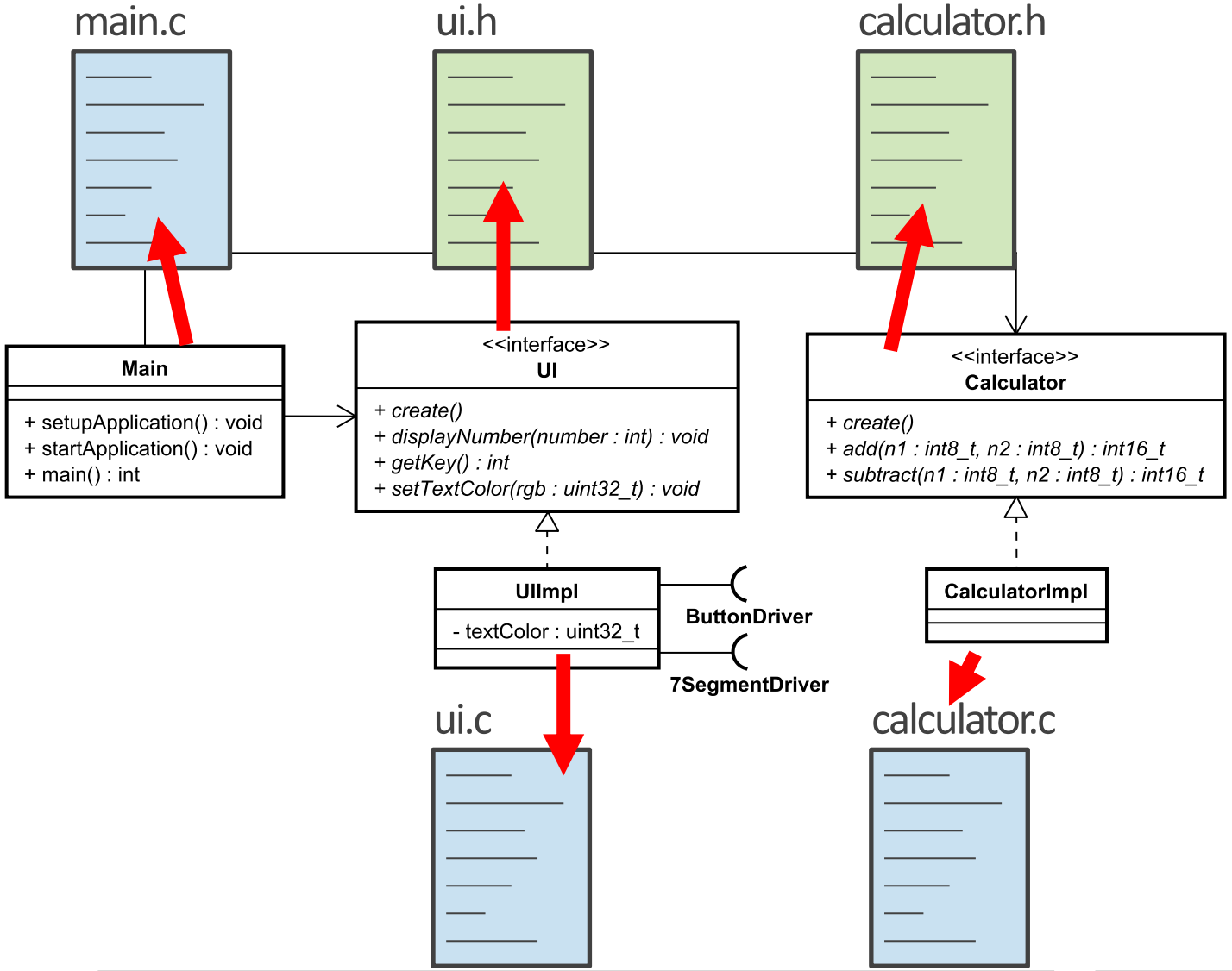
- Filenames, use underscore '_' as word delimiter or **camelCase**
- Header files (*.h)
  - Declaration of functions and global variables
  - Can be compared to Java's interfaces
- C Source files (*.c) – (modules)
  - Definition of functions (methods in Java) and variables in module scope or global scope

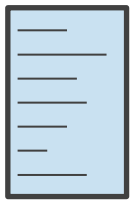


Source Files – An Example



Source Files – An Example





C-file (*.c)

Header-file (*.h)

Header Files

E.g. of file *buffer.h*

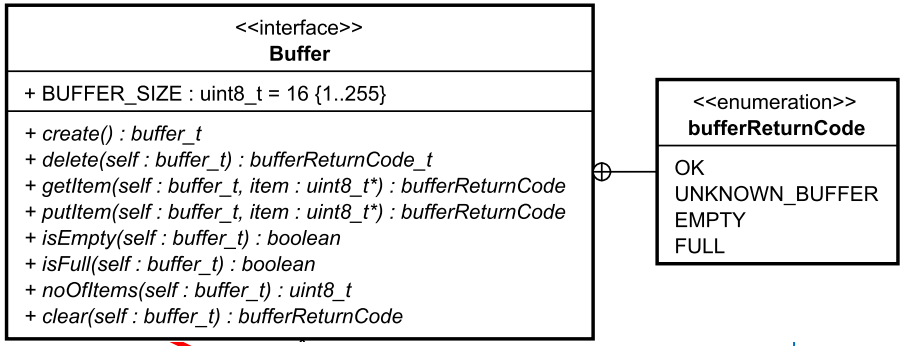
In new versions of C-compilers it is ok to just write:

```
#pragma once
```

As the first line in the file

Prevents the compiler from including the file more than once!

```
#ifndef BUFFER_H_
#define BUFFER_H_
#include <stdint.h>
#include <stdbool.h>

#define BUFFER_SIZE 16 // 1..255
typedef struct bufferStruct * buffer_t;

typedef enum {
    BUFFER_OF
    ,BUFFER_UNKNOWN_BUFFER
    ,BUFFER_EMPTY
    ,BUFFER_FULL
} bufferReturnCode_t;

buffer_t buffer_create(void);
bufferReturnCode_t buffer_delete(buffer_t self);
bufferReturnCode_t buffer_getItem(buffer_t self, uint8_t * item);
bufferReturnCode_t buffer_putItem(buffer_t self, uint8_t * item);
bool buffer_isEmpty(buffer_t self);
bool buffer_isFull(buffer_t self);
uint8_t buffer_noOfItems(buffer_t self);
bufferReturnCode_t buffer_clear(buffer_t self);
#endif /* BUFFER_H_ */
```



Standard in all Header files

Header Files – How to include

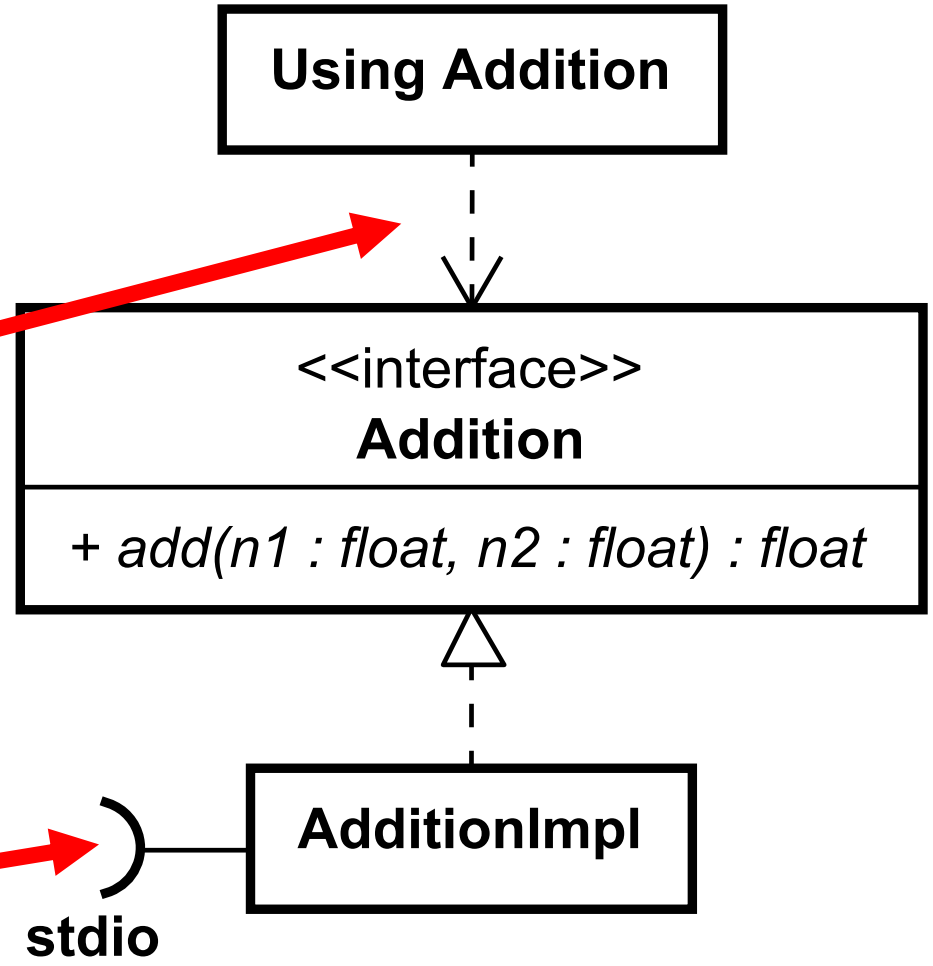
How to include/use header files in C-source files

- Including Header files locally in project:
-

```
#include "addition.h"
```

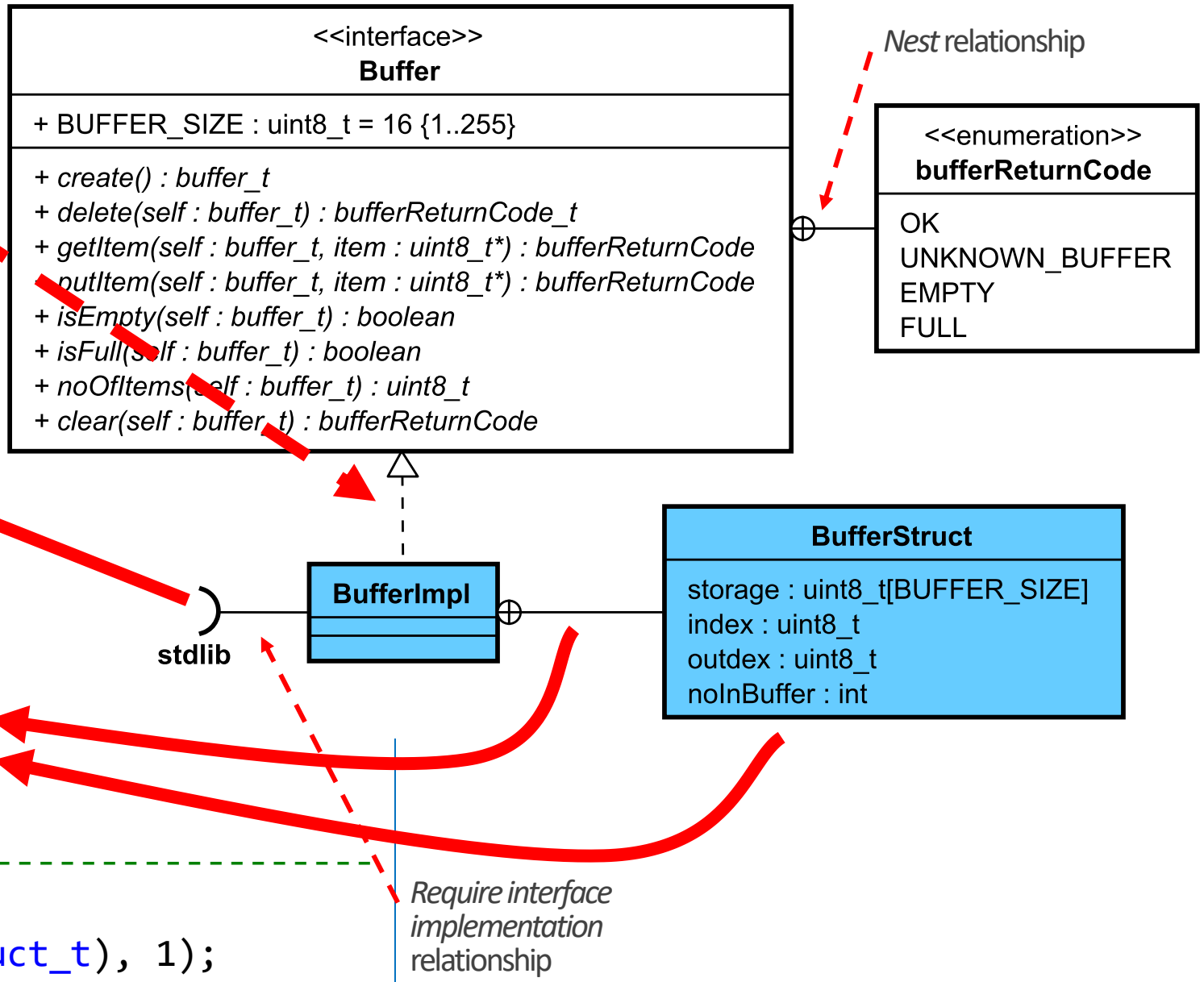
- Files that the compiler has an include path to (-I) e.g. Standard Library-files
(files outside current project)

```
#include <stdio.h>
```

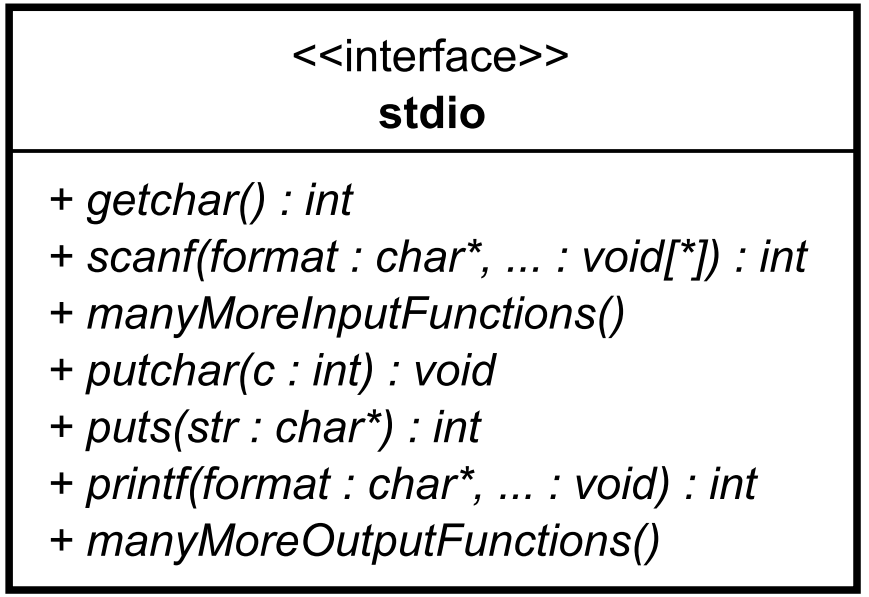


C-Source Files Implementing Interface (**buffer.c**)

```c
#include "buffer.h"
#include <stdlib.h>

// ------------------------------
typedef struct bufferStruct {
    uint8_t storage[BUFFER_SIZE];
    uint8_t index;
    uint8_t outdex;
    uint8_t noInBuffer;
}bufferStruct_t;

// ----------------------------------------
buffer_t buffer_create() {
    return calloc(sizeof(bufferStruct_t), 1);
}

// ----------------------------------------
bufferReturnCode_t buffer_delete(buffer_t self) {
```



Input/Output to console



Input from the keyboard (stdio.h)

```
int getchar() // Blocking
// Formatted input from stdin
int scanf( const char* format, ... )
```

Output to screen (stdio.h)

```
void putchar(int c) // Single character
int puts( const char* str )  // Zero terminated c-string
// Parameters like printf in Java
int printf( const char* format, ... )
```

Lets shift to Visual Studio

- **Visual Studio makes your life easier**
- **Short Demo**

Exercise Session 2

Follow this video to create your first C-Program in VS 2022:
https://www.youtube.com/watch?v=0PUZbgcIMzg

- Do *ESW1 Session 1 – Exercises*
  – but now with Visual Studio 2022

- Do *ESW1 Session 2 – Exercises*
  - can be found in ItsLearning