

# C-files and Header-files

# Source Code

---

*ABC/LP*

- What should be in .h files?
- What should be in .c files?

# Two types of .h files (“header files”)

---

*ABC/LP*

1.)

```
#include <file>
```

Gives access to standard libraries which include functions, variables and macros with a common purpose

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

Input/output functionality

```
<file>
```

Tells compiler to look for said file where your system stores include files, often /usr/include/ on Unix systems

# Two types of .h files (“header files”)

---

*ABC/LP*

2.)

```
#include "file"
```

Gives access to your own header files which can contain declarations (constants, types), macros and function prototypes

```
#include "my_header.h"
```

```
"file"
```

Tells compiler to look for this file in the current directory (where gcc is being executed)

# What can be in user-defined header files?

---

*ABC/LP*

Declarations of constants (as a macro)

```
#define    PI    3.14159
```

## pi.h

---

*ABC/LP*

```
#define    PI    3.14159
```

## pi.c

---

ABC/LP

```
#include <stdio.h>
#include <math.h>
#include "pi.h"
int main () {
    printf("The value of my pi: %1.15f \n", PI);
    printf("The value of math.h\'s pi: %1.15f \n", M_PI);
    return (0);
}
```

# pi.c after the pre-processor is done

---

*ABC/LP*

```
int main () {  
    printf("The value of my pi: %1.15f \n", 3.14159);  
    printf("The value of math.h\'s pi: %1.15f \n",  
3.14159265358979323846);  
    return (0);  
}
```



# What can be in user-defined header files?

---

*ABC/LP*

## **Declarations of types**

`typedef ...`

Declaration of your own datatypes, for example structures (struct)

More on this later...

# What can be in user-defined header files?

---

*ABC/LP*

- **Declarations of function prototypes**
- Before you use a function C must have knowledge about the type it returns and the parameter types the function expects.
- Prototypes allow for more structured and to read code.
- It allows the C compiler to check the syntax of function calls.
- It's good practice to prototype all functions at the start of the program or in a header file.

# What can be in user-defined header files?

---

*ABC/LP*

- Declare a function prototype by designating:
  - the type the function returns
  - the function name
  - the type of parameters in the order they appear in the function definition.

```
float celsius_to_fahrenheit (float celsius);
```

# What can be in .c files?

---

*ABC/LP*

- Normally .c files contain fully-written functions.
- But they often contain some of the same type of declarations as header files.
- Where they are put depends upon the desired scope of the declarations (more on scope later)
- Your own header files should not contain fully-written functions.