

WELCOME TO CSCUV4

WEEK 5

Cristian Lepore

PLAN FOR TODAY



15 mins

–

Presentation

10 mins

–

Q&A

35 mins

–

Exercises & checkpoints

WEEK 2

WEEK 3

WEEK 4

TODAY

TIME



Main function



Standard I/O



Typedef



Functions



GCC compiler



Arrays



Struct



First program



Standard I/O

ABOUT FUNCTIONS


1. Defining a Function
2. Function Declarations
3. Calling a Function
4. Function Arguments



WHAT IS A FUNCTION

Other functions

```
1
2  ✓ void max() {
3    }
4
5  ✓ void average() {
6    }
7
8  ✓ int main(){
9      /* Do something */
10
11     return 0
12 }
13
```

A red line underlines the text 'Other functions'. From the end of this line, two red arrows branch out. The upper arrow points to the opening curly brace of the 'max()' function on line 2. The lower arrow points to the opening curly brace of the 'average()' function on line 5. Both functions are enclosed in red hand-drawn brackets on the left side of the code block.

DEFINING A FUNCTION

Return_type Name Parameters (optional)

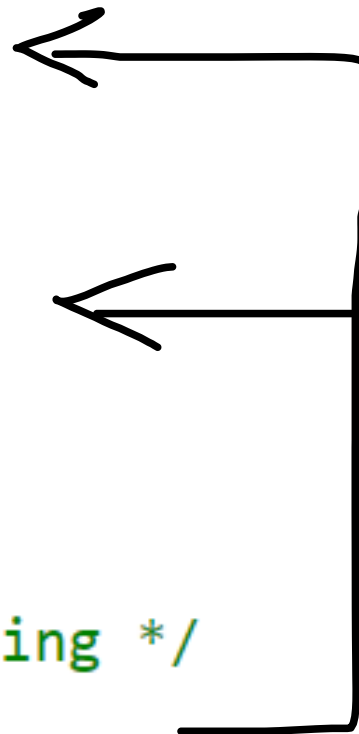
Header → 1
2 int my_function(int parameters) {
3 /* Body */
4
5 return /*something*/;
6 }
7

Body →

The diagram illustrates the components of a C function definition. The code is shown across seven lines, numbered 1 to 7. Line 1 is a blank line. Line 2 is the function header: 'int' (return type, highlighted green), 'my_function' (name, highlighted yellow), '(int parameters)' (parameters, highlighted cyan), and '{'. Line 3 is the first line of the function body: '/* Body */'. Line 4 is a blank line. Line 5 is the return statement: 'return /*something*/;'. Line 6 is the closing brace '}'. Line 7 is a blank line. Annotations include: 'Return_type' with an arrow pointing to 'int'; 'Name' with an arrow pointing to 'my_function'; 'Parameters (optional)' with an arrow pointing to '(int parameters)'; 'Header' with a blue arrow pointing to line 2; and 'Body' with a red bracket and arrow pointing to lines 3 through 6.

POSITION IN OUR CODE

```
1
2 ✓ void max() {
3   }
4
5 ✓ void average() {
6   }
7
8 ✓ int main(){
9     /* Do something */
10
11     return 0;
12 }
13
```



POSITION IN OUR CODE



```
1
2 ✓ void max() {
3   }
4
5 ✓ void average() {
6   }
7
8 ✓ int main(){
9   /* Do something */
10
11   return 0;
12 }
13
```



```
1
2
3
4 int main(){
5   /* Do something */
6
7   return 0;
8 }
9
10 void max() {
11 }
12
13 void average() {
14 }
15
```


POSITION IN OUR CODE

```
1
2 ✓ void max() {
3   }
4
5 ✓ void average() {
6   }
7
8 ✓ int main(){
9   /* Do something */
10
11   return 0;
12 }
13
```

Declaration

```
1 void max();
2 void average();
3
4 ✓ int main(){
5   /* Do something */
6
7   return 0;
8 }
9
10 ✓ void max() {
11   }
12
13 ✓ void average() {
14   }
15
```

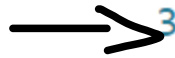
FUNCTION DECLARATION

```
1  
2 int my_function(int parameter1, int parameter2);  
3
```

```
1  
2 int my_function(int parameter1, int parameter2);  
3
```

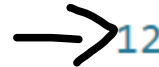
CALLING A FUNCTION

Declaration



```
2  /* function declaration */  
3  int max(int num1, int num2);  
4
```

Call



```
5  int main () {  
6      /* local variable definition */  
7      int a = 100;  
8      int b = 200;  
9      int result;  
10  
11     /* calling a function to get max value */  
12     result = max(a, b);  
13  
14     return 0;  
15 }  
16
```

Definition



```
17  /* function returning the max between two numbers */  
18  int max(int num1, int num2) {  
19      /* do something */  
20  }
```

FUNCTION ARGUMENTS BY VALUE

The scope of num1, num2 and sum is within the sum function.

```
1
2 int sum(int num1, int num2) {
3     int sum = num1 + num2;
4
5     return sum;
6 }
7
```

The scope of a, b and result is within the Main function.

```
8 int main () {
9     int a = 100;
10    int b = 200;
11    int result;
12
13    result = sum(a, b);
14
15    return 0;
16 }
17
```

Memory

num1 = 100

num2 = 200

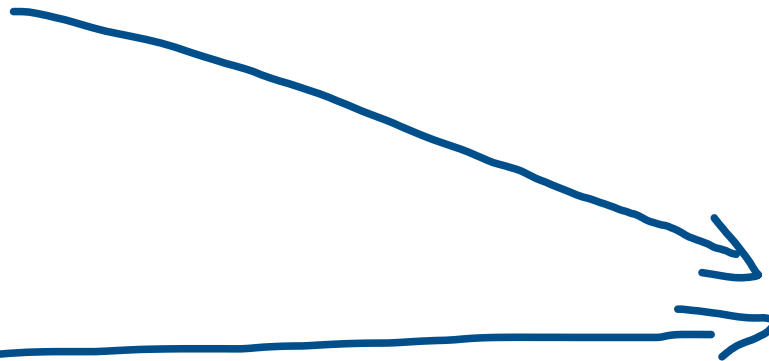
a = 100

b = 200

FUNCTION ARGUMENTS BY REFERENCE

```
1  #define N 4
2
3  void my_func(int array, int len){
4      /* Do something */
5  }
6
7  int main(){
8      int a[N] = {0};
9      my_func(a, N);
10
11     return 0;
12 }
13
```

Memory
a[0] = 0
a[1] = 0
a[2] = 0
a[3] = 0



QUESTION

```
1
2 ~ int sum(int num1, int num2) {
3     int a = 500;
4     int sum = num1 + num2;
5     printf("%d", a);
6     return sum;
7 }
8
9 ~ int main () {
10     int a = 100;
11     int b = 200;
12     int result;
13
14     result = sum(a, b);
15     printf("%d", a);
16
17     return 0;
18 }
19
```

Output ???

QUESTION

```
1
2 ✓ int sum(int num1, int num2) {
3     int a = 500;
4     int sum = num1 + num2;
5     printf("%d", a);
6     return sum;
7 }
8
9 ✓ int main () {
10     int a = 100;
11     int b = 200;
12     int result;
13
14     result = sum(a, b);
15     printf("%d", a);
16
17     return 0;
18 }
19
```

Output ???

a = 500

a = 100

CHECKPOINTS GROUP A

ID	Week 2	Week 3	Week 4	Total
2839067/1				0
2816787/1		check	check	2
2817566/1	check	check	Failed	2
2825056/1	check	check late	check	3
2835267/1	check	check	check	3
2823680/1	check	check	check	3
2811801/1	check	check	check	3
2836012/1	check	check	check	3
2810713/1	check	check	check	3

Checkpoint current lesson

Checkpoint one week late

Failed

CHECKPOINTS GROUP B

ID	Week 2	Week 3	Week 4	Total
2928413/2				0
2823735/1	check	to recover	Failed	1
2813060/2				0
2710797/1	check late	check	Failed	2
2944806/1	check			1
2823106/1	Failed			0
2814919/1	check	check	Failed	2
2839798/2	check late	Failed	Failed	1
2716869/1				0

Checkpoint current lesson

Checkpoint one week late

Failed

Any Questions?

Thank you