

Sprint 01

Marathon C

September 3, 2020



 **code connect**

Contents

Engage	2
Investigate	3
Act: Task 00 > Hello world	5
Act: Task 01 > Say wake up	6
Act: Task 02 > Write knock, knock	7
Act: Task 03 > Matrix voice	8
Act: Task 04 > Print character	9
Act: Task 05 > Only printable	10
Act: Task 06 > Hexadecimal	11
Act: Task 07 > Print alphabet	12
Act: Task 08 > String length	13
Act: Task 09 > Print string	14
Share	15

Engage

DESCRIPTION

Hey, wazzup?

You just started learning programming. That's nice! Let's go further. During this challenge, you will learn the basics of writing code in `C`.

We invite you to start learning programming with `C` because:

- `C` is a great foundation for learning other programming languages.
- `C` is built on basic programming concepts and it is very simple to understand how to develop programs using it.
- When using `C`, you are always aware of how your program works under the hood. It doesn't hide anything from you. You've got the power.
- Last but not least, coding on `C` in accordance with the `Auditor` will help you to develop a mindset of a true programmer.

Welcome to `C`!

Please, follow me.

BIG IDEA

Develop a programmer mindset.

ESSENTIAL QUESTION

What are the components of the simplest C program?

CHALLENGE

Learn the basics of C.

Investigate

GUIDING QUESTIONS

We invite you to find answers to the following questions. By researching and answering them, you will gain the knowledge necessary to complete the challenge. To find answers, ask the students around you and search the internet. We encourage you to ask as many questions as possible. Note down your findings and discuss them with your peers.

- What do you remember the most from the previous **Sprint**?
- How was the **Sprint00**? How many tasks have you done?
- What topics were unclear to you?
- What programming languages do you know? What do you know about them?
- What do you know about the C language?
- How to write "Hello World!" in C?

GUIDING ACTIVITIES

Complete the following activities. Don't forget that you have a limited time to overcome the challenge. Use it wisely. Distribute tasks correctly.

- Discuss the weather with the other students. What are their favorite cookies? What we really mean here is: create a friendly work environment to ensure comfortable cooperation and effective Peer-to-Peer.
- Repeat the basics from the previous challenges. Create and remove some test files only using Unix-commands.
- Open test files using **Vim** or **Emacs**. Write something. Google some hotkeys and commands for text editors that can increase your productivity.
- **Man** unfamiliar words.
- Watch some C tutorials on **YouTube**. Type "learn c" in the search bar.
- Also watch tutorials that ucode connect has prepared for you:
 1. **c_basics_1** about how to write your first program
 2. **c_basics_2** about data types and work with variables
 3. **c_basics_3** about different operations
 4. **c_basics_4** about operators, conditions and cycles
 5. **c_basics_5** about functions
- Open **Terminal**. Create and open a new file **hello_world.c** using **Vim** or **Emacs**.
- Write a "Hello world" program using the **printf** function. Save the file.
- Compile this file as follows

```
clang -std=c11 -Wall -Wextra -Werror -Wpedantic hello_world.c -o hello_world.
```
- Run your program with **./hello_world**.
- Try to write the same program using the **write** function instead of **printf**.

- **Auditor** is a style guide for **C**. You can find the pdf of the Auditor [here](#) or in **LMS->Media**. Keep in mind that you must follow the Auditor rules for all the code you write in **C**.
- Clone your git repository that is issued on the challenge page in the LMS. Use **git clone** for this.
- You are ready to do the **t00**.
- Communicate with students and share information.

ANALYSIS

Analyze your findings. What conclusions have you made after completing guiding questions and activities? In addition to your thoughts and conclusions, here are some more analysis results.

- Be attentive to all statements of the story. Examine the given examples carefully. They may contain details that are not mentioned in the task.
- Analyze all information you have collected during the preparation stages.
- Perform only those tasks that are given in this document.
- Submit your files using the layout described in the story. Only useful files allowed, garbage shall not pass!
- Compile C-files with clang compiler and use these flags:
clang -std=c11 -Wall -Wextra -Werror -Wpedantic.
- Pay attention to what is allowed in a certain task. Use of forbidden stuff is considered a cheat and your tasks will be failed.
- Complete tasks according to the rules specified in the **Auditor**.
- The solution will be checked and graded by students like you. **Peer-to-Peer learning**.
- Also, the challenge will pass automatic evaluation which is called **Oracle**.
- If you have any questions or don't understand something, ask other students or just Google it.
- Use your brain and follow the white rabbit to prove that you are the Chosen one!

Act: Task 00

NAME

Hello world

DIRECTORY

```
t00/
```

SUBMIT

```
main.c
```

ALLOWED FUNCTIONS

```
printf
```

DESCRIPTION

Create a program that outputs the text below to the standard output followed by a newline.

CONSOLE OUTPUT

```
>clang -std=c11 -Wall -Wextra -Werror -Wpedantic -o hello_world main.c
>./hello_world | cat -e
Hello World$
>
```

FOLLOW THE WHITE RABBIT

```
man 3 printf
```

Act: Task 01

NAME

Say wake up

DIRECTORY

```
t01/
```

SUBMIT

```
mx_say_wake_up.c
```

ALLOWED FUNCTIONS

```
printf
```

DESCRIPTION

Create a function that outputs the text below to the standard output followed by a newline.

SYNOPSIS

```
void mx_say_wake_up(void);
```

CONSOLE OUTPUT

```
>./mx_say_wake_up | cat -e
Wake up, NEO \ ( ^_^ ) / ...$
The Matrix has you ...$
>
```

FOLLOW THE WHITE RABBIT

```
man 3 printf
```


Act: Task 02

NAME

Write knock, knock

DIRECTORY

```
t02/
```

SUBMIT

```
mx_write_knock_knock.c
```

ALLOWED FUNCTIONS

```
write, strlen
```

DESCRIPTION

Create a function that outputs the text below to the standard output followed by a newline.

SYNOPSIS

```
void mx_write_knock_knock(void);
```

CONSOLE OUTPUT

```
>./mx_write_knock_knock | cat -e
Follow the white rabbit.$
Knock, knock, Neo.$
>
```

FOLLOW THE WHITE RABBIT

```
man 2 write
man strlen
```


Act: Task 03

NAME

Matrix voice

DIRECTORY

```
t03/
```

SUBMIT

```
mx_matrix_voice.c
```

ALLOWED FUNCTIONS

```
write
```

DESCRIPTION

Create a function that outputs the smallest unit of matrix voice - beep (sound signal).

SYNOPSIS

```
void mx_matrix_voice(void);
```

CONSOLE OUTPUT

```
>./mx_matrix_voice | cat -e
^G%
>
```

FOLLOW THE WHITE RABBIT

```
man 2 write
man ascii
```

SEE ALSO

Matrix voice

Act: Task 04

NAME

Print character

DIRECTORY

```
t04/
```

SUBMIT

```
mx_printchar.c
```

ALLOWED FUNCTIONS

```
write
```

DESCRIPTION

Create a function that outputs a single character to the standard output.

SYNOPSIS

```
void mx_printchar(char c);
```

FOLLOW THE WHITE RABBIT

```
man 2 write  
man ascii
```

Act: Task 05

NAME

Only printable

DIRECTORY

```
t05/
```

SUBMIT

```
mx_only_printable.c, mx_printchar.c
```

ALLOWED FUNCTIONS

```
write
```

DESCRIPTION

Create a function that outputs all printable characters in reverse order to the standard output followed by a newline.

Hint: Space is a printable character.

SYNOPSIS

```
void mx_only_printable(void);
```

FOLLOW THE WHITE RABBIT

```
man ascii
```

Act: Task 06

NAME

Hexadecimal

DIRECTORY

```
t06/
```

SUBMIT

```
mx_hexadecimal.c, mx_printchar.c
```

ALLOWED FUNCTIONS

```
write
```

DESCRIPTION

Create a function that outputs characters representing hexadecimal numerals in ascending order to the standard output followed by a newline. Characters must be in uppercase.

SYNOPSIS

```
void mx_hexadecimal(void);
```

EXAMPLE

```
void mx_hexadecimal(void); //prints 0...F ; there must be all characters instead of ...
```

FOLLOW THE WHITE RABBIT

```
man ascii
```


Act: Task 07

NAME

Print alphabet

DIRECTORY

```
t07/
```

SUBMIT

```
mx_print_alphabet.c, mx_printchar.c
```

ALLOWED FUNCTIONS

```
write
```

DESCRIPTION

Create a function that outputs the alphabet, alternating upper and lower case characters in ascending order to the standard output followed by a newline. See the output in the **EXAMPLE** below.

SYNOPSIS

```
void mx_print_alphabet(void);
```

EXAMPLE

```
mx_print_alphabet(); //prints AbC... ; there must be full alphabet instead of ...
```

FOLLOW THE WHITE RABBIT

```
man ascii
```

Act: Task 08

NAME

String length

DIRECTORY

```
t08/
```

SUBMIT

```
mx_strlen.c
```

ALLOWED FUNCTIONS

None

DESCRIPTION

Create a function that has the same behaviour as the corresponding standard libc function `strlen`.

SYNOPSIS

```
int mx_strlen(const char *s);
```

FOLLOW THE WHITE RABBIT

```
man 3 strlen
```

Act: Task 09

NAME

Print string

DIRECTORY

```
t09/
```

SUBMIT

```
mx_printstr.c, mx_strlen.c
```

ALLOWED FUNCTIONS

```
write
```

DESCRIPTION

Create a function that outputs a string of characters to the standard output.

SYNOPSIS

```
void mx_printstr(const char *s);
```

FOLLOW THE WHITE RABBIT

```
man 2 write
```

Share

PUBLISHING

Last but not least, the final stage of your work is to publish it. This allows you to share your challenges, solutions, and reflections with local and global audiences. During this stage, you will discover ways of getting external evaluation and feedback on your work. As a result, you will get the most out of the challenge, and get a better understanding of both your achievements and missteps.

To share your work, you can create:

- a text post, as a summary of your reflection
- charts, infographics or other ways to visualize your information
- a video, either of your work, or a reflection video
- an audio podcast. Record a story about your experience
- a photo report with a small post

Helpful tools:

- [Canva](#) - a good way to visualize your data
- [QuickTime](#) - an easy way to capture your screen, record video or audio

Examples of ways to share your experience:

- [Facebook](#) - create and share a post that will inspire your friends
- [YouTube](#) - upload an exciting video
- [GitHub](#) - share and describe your solution
- [Telegraph](#) - create a post that you can easily share on Telegram
- [Instagram](#) - share photos and stories from ucode. Don't forget to tag us :)

Share what you've learned and accomplished with your local community and the world. Use [#ucode](#) and [#CBLWorld](#) on social media.