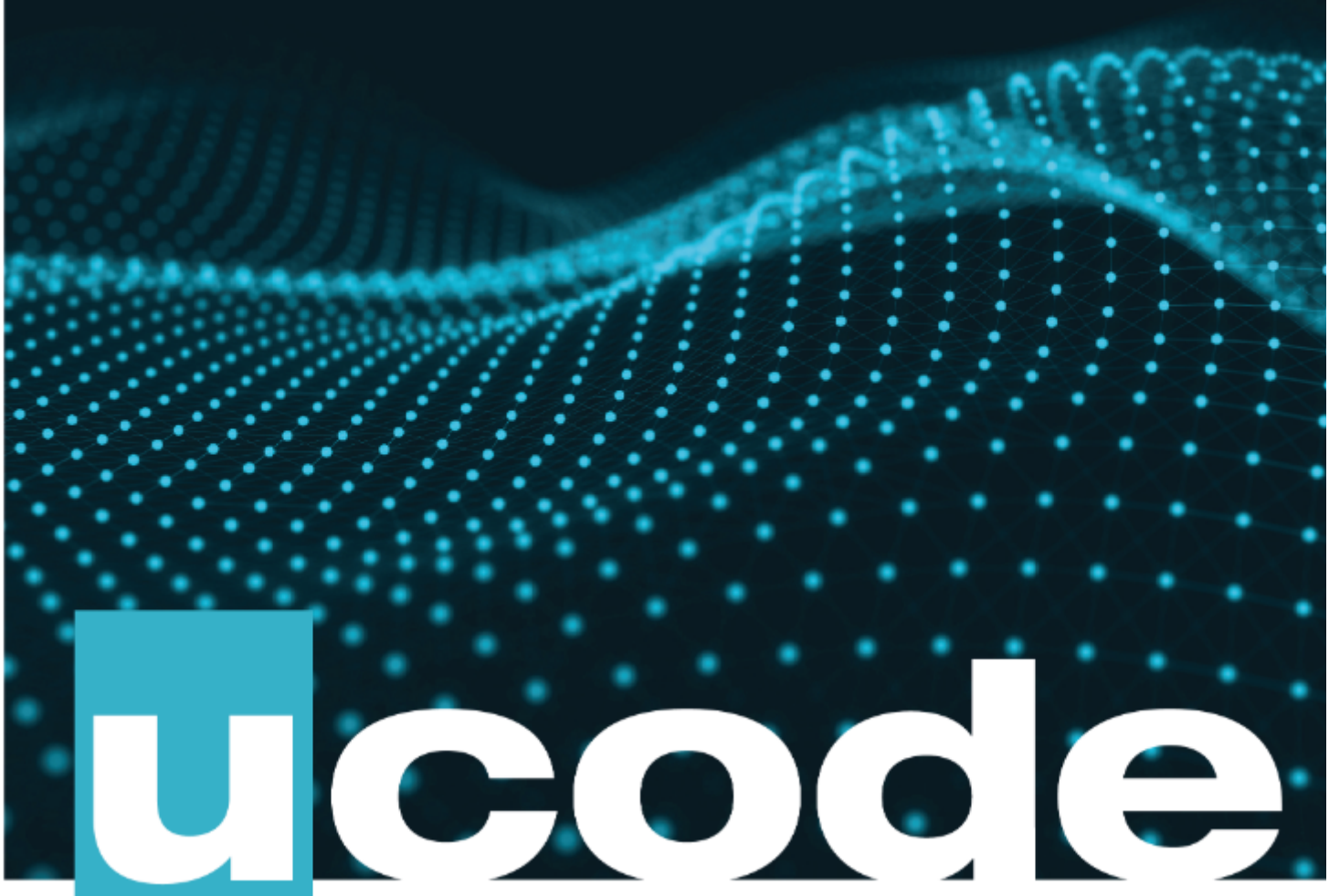
Race 05

Marathon C

April 30, 2020



ucode

Contents

ucode

Engage

DESCRIPTION

Ready to bring your code to a new, visibly higher, level?

You are starting the last Race of the Marathon C. You've accomplished a lot, and you have learned to implement rather difficult programs. However, everything you've done so far was made for simple console input and output. A CLI is great for some programming and maintenance tasks, but if you want to develop software that users enjoy and want to use, you need graphics. Besides, a graphical user interface has become a must-have for modern software.

To put it simply, graphics are used to present information visually by drawing shapes, coloring objects, creating the illusion of movement, etc. There is an always-increasing number of ways to create, modify, and display graphics.

This Race is a chance to make your first steps in graphics. You will use a library that wraps the Terminal's functionality and produces an appealing user interface. This way, you will be able to practice the key ideas of graphics, within a familiar environment.

The key role of this race to prepare for the final challenge. After completing this challenge, you will know the basics of working with a graphics library, and have the skills and knowledge that are vital for the Endgame.

Have fun, and may the odds be ever in your favor!

BIG IDEA

Dive into graphics.

ESSENTIAL QUESTION

What to learn first?

CHALLENGE

Create a screensaver in the Terminal using `ncurses`.

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 in the Slack and search the internet. We encourage you to ask as many questions as possible. Note down your findings and discuss them with your peers.

- How were your Sprints? Did you like all of them?
- What is `unicode`?
- What are `graphics`?
- What is a `graphic library`?
- What is `ncurses`?
- What is the `x-axis` and the `y-axis`?
- How to print something in the Terminal using `ncurses`?
- How to handle key events using `ncurses`?

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.

- Meet with your team in the Zoom. Discuss how you will organize teamwork. Create a channel for a team communication.
- Learn information about `ncurses`.
- Create a standard "Hello world!" program using `ncurses`.
  - Make sure that `ncurses` is installed on your computer
  - Include the `ncurses.h` library in your program
  - Google `ncurses hello world example`
  - Compile Hello World !!! as follows: `clang test_program.c -lncurses`
- Add color to your program from the first paragraph. Output "Hello world!" in blue color on a black background, for example. Google it.
- Make the phrase symbols green, red, cyan, etc. So that each symbol has a different color.
- Create a small animation using some guides from Google. Like that.
- Read the story, taking everyone's ideas on board.
- Clone your git repository that is issued on the challenge page in the LMS.
- Distribute tasks among all team members.
- Start to develop the solution. Offer improvements. Test your code.
- 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.

- Challenge has to be carried out by an entire team.
- Each team member must understand the challenge and realization, and be able to reproduce it individually.
- It is your responsibility to assemble the whole team. Phone calls, SMS, messengers are good ways to stay in touch.
- You can proceed to `Act: Creative` only after you have completed all requirements in `Act: Basic`. But before you begin to complete the challenge, pay attention to the program's architecture. Take into account the fact that many features indicated in the `Act: Creative` require special architecture. And in order not to rewrite all the code somewhen, we recommend you initially determine what exactly you will do in the future.
- Be attentive to all statements of the story.
- Analyze all information you have collected during the preparation stages. Try to define the order of your actions.
- 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`.
- Your program must manage memory allocations correctly. Memory which is no longer needed must be released otherwise the task is considered as incomplete.
- Pay attention to what is allowed. Use of forbidden stuff is considered a cheat and your challenge 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: Basic

NAME

Matrix rain

DIRECTORY

`./`

SUBMIT

`Makefile, inc/*.[h], src/*.[c]`

ALLOWED FUNCTIONS

write, malloc, free, exit, rand, srand, usleep, time, setlocale, all the functions defined in the ncurses library

BINARY

race05

LEGEND

Matrix code, also known as Matrix digital rain or sometimes green rain, is the computer code related to the Matrix. The falling green code is a way of representing the activity of the virtual reality environment of the Matrix on screen.

"You get used to it, I don't even see the code. All I see is a blond, brunette, redhead."
– Cypher

DESCRIPTION

Create a program that simulates the screensaver from `The Matrix`.
The program must show text running from top to bottom of the `Terminal`. It must scroll through all the lines with the same speed.
The illusion of `falling` columns with different heights filled with random characters along the width of your Terminal must be created.
`The first` (lowest) character of each column must be `white` and changing continuously, and the remaining characters of the column must be `green` by default. Background of the Terminal must be `black` by default.

The program must:

- use all printable `ASCII` characters
- exit by pressing on `q` on the keyboard
- not handle window resize
- display usage in case of invalid input

SEE ALSO

See `resources/basic.gif` (this is just the example)

Act: Creative

DESCRIPTION

Just a few ideas of features which you can add to the program:

- using Japanese or any other multibyte characters instead of `ASCII`
- handling `-s` flag to run the program in the screensaver mode
- the speed of the animation can be changed during runtime or program startup
- the color of the characters or the background can be changed during runtime or program startup
- printing messages from the Neo's Terminal like in the example `resources/creative.gif`
- other cool features. Be creative and use your imagination, Neo!