Exercise 2: Logic With The LED

The LiFi-project

Prof. Dr. Nicolas Meseth

OVERVIEW

In the third exercise of the LiFi-project, you'll start programming using the LED and you'll deepen your understanding of code systems.

Here are the relevant lessons of the online script. Make sure you study them carefully in order to solve this exercise:

- 4 - Program's Anatomy
- 5 - The LED
- 6 - Code Systems

Good Luck!

YOUR TASKS

To pass this exercise, you must complete the following tasks and submit your results via ILIAS. You find the details on the form of submission below.

1. Colors By Menu

In the LiFi prototype, the LED will act as the sender of information. This requires frequent changes of the LED's color and turning it off at times. To practice this, create a program that changes the LED's color based on the user's keyboard input. More specifically:

- Write a program that initially turns off the LED and then waits for the user to enter a key on the keyboard and confirm the input with the enter key.
- The input can be any of the following letters: "r", "g", "b", "y", or one of the keywords "off" or "exit". All other input is invalid and should result in an error message: *"Invalid input. Please type either r, g, b, y, or off/exit."*

- Each of the four letters represents a color (r = red, g = green, b = blue, y = yellow) to which the LED should be set after the user confirms her choice with the enter key.
- The keyword "off" should turn off the LED. If it is already off, then nothing happens and a warning message is printed: *"LED is already off"*.
- The keyword "exit" should exit the program after it printed a short good bye message. As long as the user doesn't enter "exit", she should be asked for another input.

2. Hexadecimal Conversion

Write a program that prompts the user for an RGB code expressed in hexadecimal (e.g. `#00FF00`). On hitting enter, the program should convert the hexadecimal code and turn the LED on in the respective color.

Hint: You can use the `int()` function in Python to parse a hexadecimal number to its decimal equivalent:

```
hex = "FF"
dec = int(hex, 16)
print(dec)
# 255
```

3. SOS With Morse Code

Write a program that uses the LED to signal "SOS" in Morse code. Choose any color you like. Select a unit length that allows a trained human to decode the message.

SUBMISSION

For this exercise, please submit:

- Three Python programs, each in a separate file for the tasks above:
  - `colors_by_menu.py`
  - `hexadecimal_conversion.py`
  - `sos.py`

Submit all files via the corresponding exercise in ILIAS.

MORE EXERCISES TO PRACTICE

The following tasks are for you to practice your skills. They are optional and not part of the submission.

4. Morse Code Messages

Extend the third exercise "SOS With Morse Code" to allow the user to enter any text sequence from A-Z as well as the numbers 1-9. The program should translate the whole input sequence into Morse code and emit the corresponding Morse code using the LED.

5. Random Color Sequence

Write a program that randomly chooses a color from the RGB spectrum and sets the LED to that color for a constant duration d. The program should change the LED's color a configurable number of times until it exits.

What happens when all RGB values are very small by chance? How could you prevent this from happening and improve the program?

6. Traffic Light

Write a program that simulates the LED as a traffic light. The traffic light should start with a red-phase that lasts d_{red} seconds. The red-phase is followed by a transition-phase from red to green, where the LED turns orange for d_{orange} seconds before it turns green and enters the green-phase. The green-phase lasts d_{green} seconds and is yet again followed by a transition-phase, this time from green to red. The orange light again lasts for d_{orange} seconds. Choose useful values for the three duration parameters d_{red}, d_{green}, and d_{yellow}. Make sure that

$$d_{red} > d_{green} > d_{orange}$$

holds true.

7. The Flashlight Game

This exercise is inspired by chapter four of Petzold (2022) and has nothing to do with programming, it doesn't even involve a computer. It's a game that is designed to be played in the classroom. You'll only need:

- A flashlight with working batteries
- Ten paper cards and two pencils
- A printed version of the Morse code alphabet

Here are the game instructions:

- In this game, small teams of 2-4 students compete against each other in communicating short word via Morse code.
- For a better experience, turn off the lights and close the shades if available. The room should be as dark as possible. But not so dark we can't see anymore.
- On the paper cards, each team writes down 5 different terms from the LiFi project that we introduced so far.
- Decide which team starts. The starting team sends one person to the other side of the room. We call this person the *communicator*, and he or she must carry the flashlight.
- The rest of the teams gather and decide which of their terms they want to give to the opponent team's *communicator*. The communicator then has the task to transmit the term using only the flashlight and Morse code. No sounds or other signals are allowed. If the group guesses the correct term within 2 minutes, it gets 1 point.
- Alternatively, you can use a stop timer and record the time it took the team to guess the term. The group with the lowest total time to guess five terms wins.
- Now, the next team takes turn and sends a communicator for their first term. The steps repeat until each team attempted to guess five terms. The group with most points wins.

MORE QUESTIONS TO PRACTICE

Try to answer the following questions to practice your understanding of the topics around the LiFi-project. The questions are optional and not part of the submission.

6. Code Systems

- How many colors could we encode if we used only 7 bits instead of 8 for each of the base colors red, green, and blue?
- Would it make sense to create an extended version of the RGB code that adds an extra byte for each of the base colors? How many colors would we be able to encode then? Find arguments for and against this idea!
- We did not always enjoy the full 16,777,216 colors on our computer screens. Do some research on how color codes evolved over the history of digital computers. Can you find some screenshots from long gone times?
- What is a *variable bit-length* code?
- Is Morse code a binary code?
- Which character takes the longest to transmit with Morse code?
- How many bits does Braille code us to represent a symbol? Is Braille code a binary code?

REFERENCES

Petzold, Charles. 2022. *Code: The Hidden Language of Computer Hardware and Software*. 2nd ed. Hoboken: Microsoft Press.