

Intro to the Raspberry Pi (RPi) & The Internet of Things (IoT)

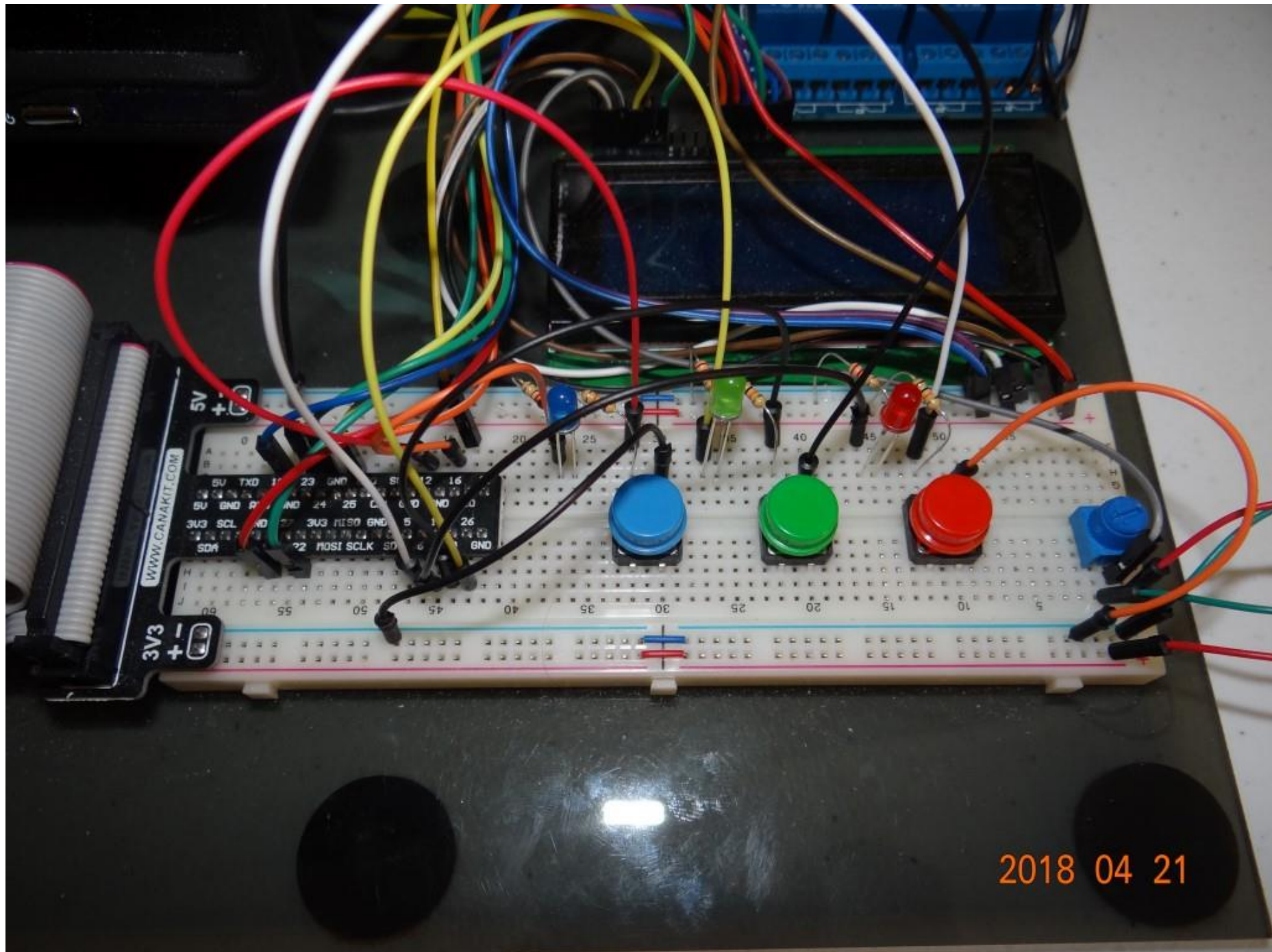
Class 6 – 2018 / 04 / 22

Please send me your LCD displays!
I need to prepare them for next class

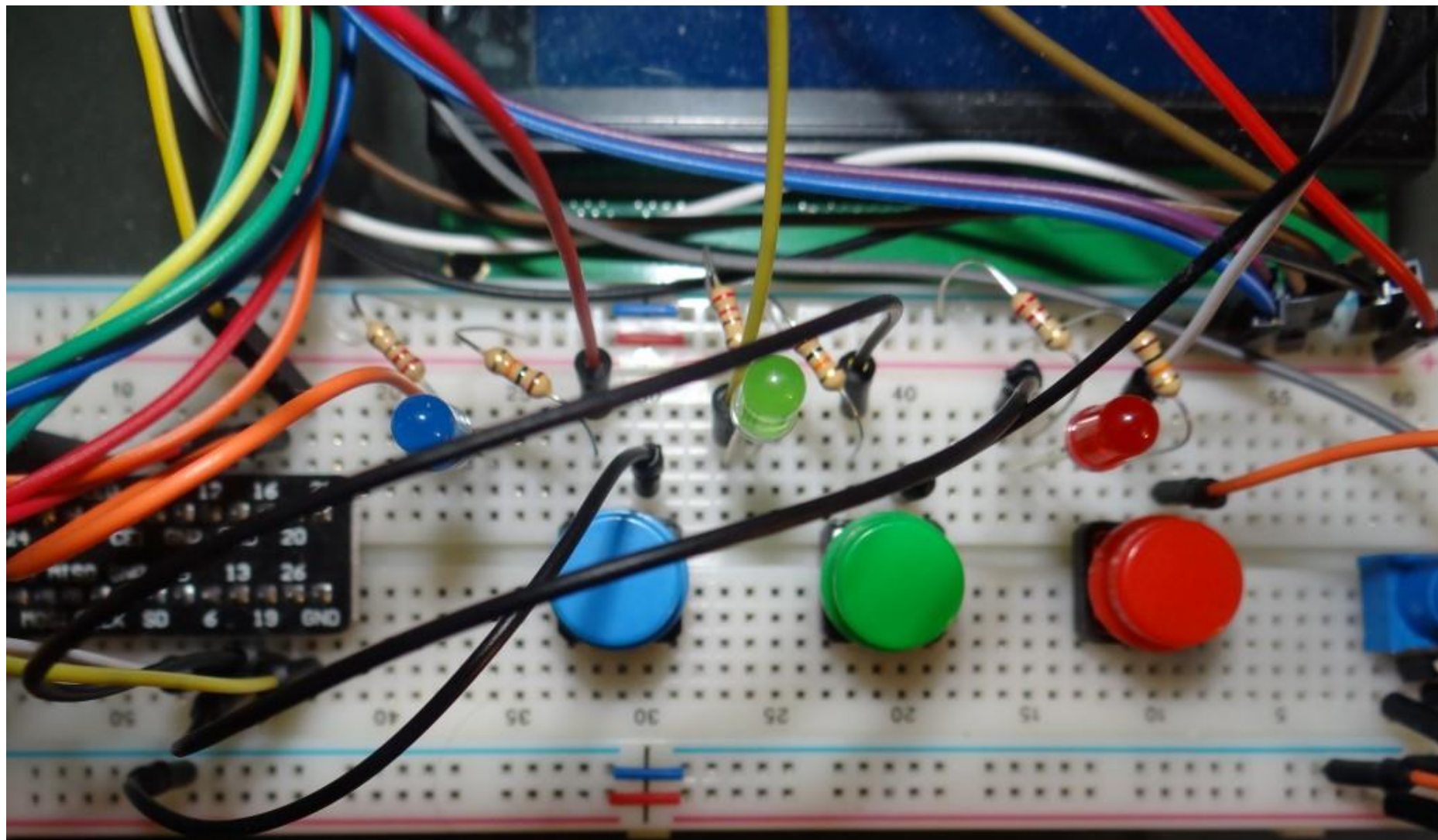


Let Make a Game!

- „Catch the LED“ Game
 - A reaction game with 3 LEDs and 3 buttons
 - There is one button per LED
 - After a random time between 0.2 and 5 seconds, a random LED lights up
 - Use has to push the corresponding button *fast* (0.2 seconds?) to get a point
 - Don't allow to cheat!
- After 20 attempts, report points
- Game should accelerate (get faster)



2018 04 21



2018 04 21

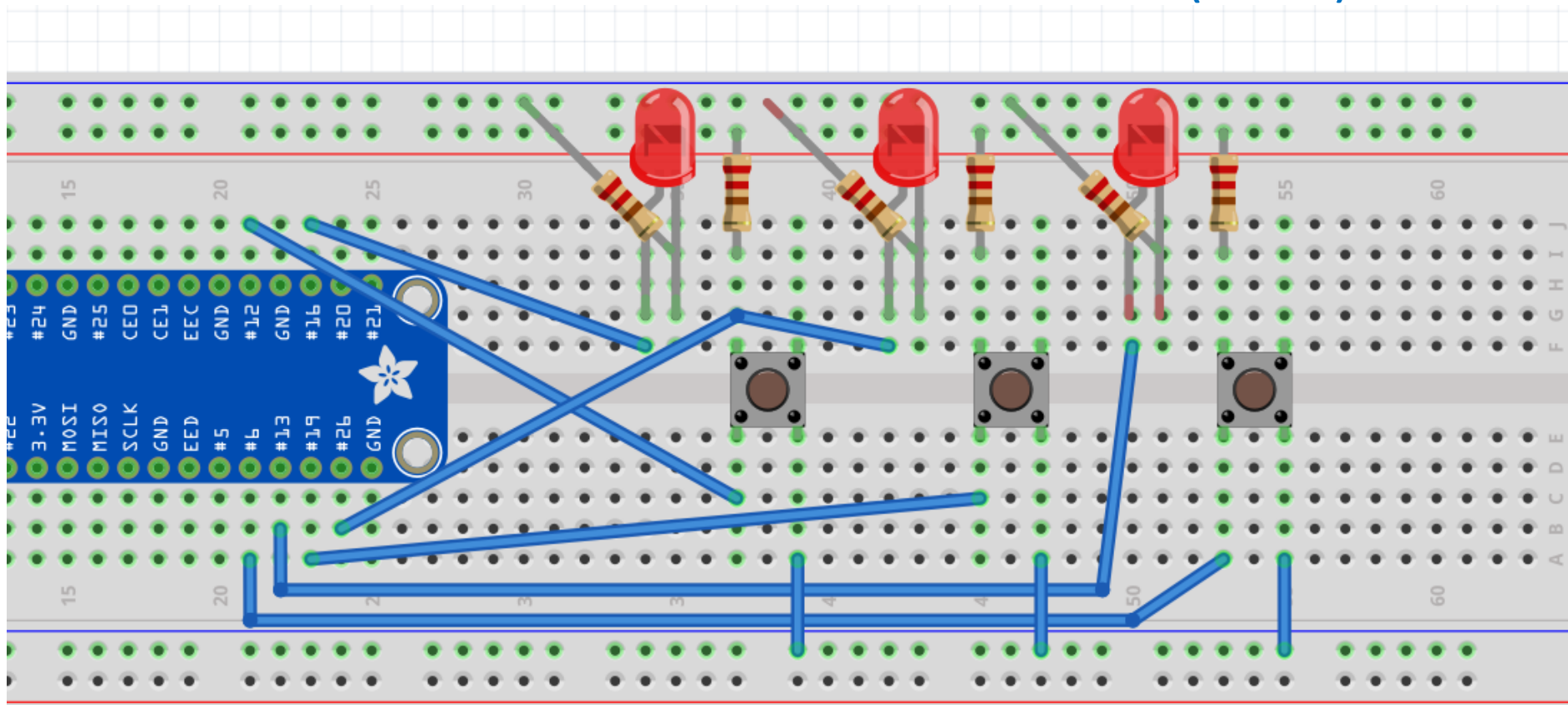
LEDs: 16 26 13 via cables to long leads of LEDs

220 Ohm Resistors from SHORT LEAD to GND (BLUE)

BUTTONS: 12 19 6 via cables to left side pins of buttons

10 kOhm PULL-UP RESISTORS TO PLUS 5 V (RED)

BUTTON RIGHT SIDE PINS TO GND (BLUE)



Step 1

- Test your setup: run <https://www.michael-wessel.info/class6/step1.py>
- Do the LEDs light up when you press the buttons?
- Task 1: write a function „button_pressed(button)“ and replace the
if GPIO.input(button) == GPIO.LOW:
with a call to your “button_pressed” function
- Task 2: what do we need to change if we want LEDs to go OFF when then button is pressed instead of ON?

Step 2

- Solution:
<https://www.michael-wessel.info/class6/step2.py>
- Take some time to understand the program!
- Task 3: extend step2.py to turn on (and off) a random LED
 - Add
 - `import random`
 - `import time`at the top of the program
 - Use
 - `choice = random.randint(0, 2)`to get a random number from 0 to 2
 - Get the LED using
 - `led = leds[choice]`
 - Wait a bit using
 - `time.sleep(0.3)`

Solution to Step3

- **Solution:**
<https://www.michael-wessel.info/class6/step3.py>
- **Take some time to understand the program!**

How to do something while waiting

- In the game, we want to wait for a while for the user to press the button after the LED turned on
 - DURING that time, the program needs to notice when a button is pressed!
 - We CAN NOT do that if we simply wait using `time.sleep(wait_time)`
 - We need to say what happens WHILE we are waiting

(Possible) Solution – „Waiting Loop“

```
ts = time.clock()  
while time.clock() - ts < wait_time :  
    print("waiting")
```

OR, if we don't want to do anything, use „pass“:

```
ts = time.clock()  
while time.clock() - ts < wait_time :  
    pass
```

Now, within that loop, we can check if buttons are being pressed etc.

Step 4

- Task 4: extend step3.py
 - For the given „choice“, determine the corresponding button
 - Hint: there is a list variable „buttons“
Look how „choice“ is used to determine the „led“ from „leds“
Use the same to determine the „button“ from the „buttons“ !
 - Within the „while“ time waiting loop, check if the right button is pressed, for that LED
 - If it gets pressed, then
`print(“Good job!”)`
and exit the while time waiting loop using
`break`
Hint: use the function `button_pressed` you wrote earlier

Step 5

- Solution:
<https://www.michael-wessel.info/class6/step4.py>
- Take some time to understand the program!
- Task 5: extend step4.py such that points are given each time you caught the LED on time
 - Use a variable
points
to keep track of the points, add +1 for each hit
 - Print the current points in the main while loop using
print("Points: ", points)

Step 6

- Solution:
<https://www.michael-wessel.info/class6/step5.py>
- Take some time to understand the program!
- Task 6: extend step5.py such that 1 is SUBTRACTED from **points** variable each time a WRONG BUTTON is pressed while waiting for the user to press the right button
 - given button, determine the other 2 buttons which SHOULD NOT be pressed
 - Use 2 variables: **other_button1, other_button2**
 - Determine these based on **button**
 - Make sure that the **points** (score) cannot get negative!

Step 7

- **Solution:**
<https://www.michael-wessel.info/class6/step6.py>
- **Take some time to understand the program!**
- **Task 7: extend step6.py such that it is impossible to cheat!**
Cheating is easy currently:
 - Just hold down all buttons all the time
 - When it is checked if the right button is pressed down, an all time pushed down button will get you a point
 - **Q: How can we detect cheating like this?**
A: Make sure button is NOT already pushed down before going into the while waiting loop
 - **Subtract 5 points whenever cheating is detected!**

Step 8

- **Solution:**
<https://www.michael-wessel.info/class6/step7.py>
- **Take some time to understand the program!**
- **Task 8: add SOUND EFFECTS when**
 - A: Cheating is detected**
 - B: The right button is pressed**
 - C: A wrong button is pressed**
 - Take a look at and run
<https://www.michael-wessel.info/class6/step8.py>
 - **EXTEND** that program to add sounds **for case B, C**
 - **Sound for A is already implemented**

Download Sound Effects

- <https://www.michael-wessel.info/class6/wav/>
- Create a folder on your Desktop called WAV
- /home/pi/Desktop/WAV/
- Copy all the *.wav file to there

Step 9

- Solution:
<https://www.michael-wessel.info/class6/step9.py>
- Take some time to understand the program!
- Task 9: make the game more interesting
Instead of an endless loop
 while True:
 use a
 for step in range(0, tries):

 Game stops after **tries** rounds (variable, e.g., 20).
 After that, GAME OVER, and achieved SCORE is reported.
- Task 10:
let the game ACCELERATE!
Use the **step** variable to reduce **the time.sleep(wait)** time.

Step 10 – Final Solution for Today

- **Solution:**
<https://www.michael-wessel.info/class6/step10.py>
- **Take some time to understand the program!**
- **Next time: add the LCD to show the score!**

