# Lab 4: Working with text strings

We want to get a program that presents a menu, that allows us to interact with the functions we wrote in Lab 3. The main menu should look something like this:

```
(S)et number of agents (currently 0)
(G)enerate agents
(P)rint list of agents
(Q)uit
```

If the user presses an s (or S), it should ask for the number of agents.
If the user presses a g (or G), it should generate a list of randomly generated agents (with values for $x$, $\varepsilon$, and $\tau$).
If the user presses a p (or P), it should print the list of agents.
If the user presses a q (or Q), it should exit the program.

## Part A: Writing an interactive menu

1. Write a function that creates a loop with a set of menu options. Whatever the user presses, it should print the menu again.
2. Implement the quit functionality: if the user presses q or Q, it should exit the menu loop (and thus the program).
3. Implement the set number of agents functionality, using a separate function to gather the user input. Ensure that the menu correctly states the number.
4. Implement the generate agents functionality. Use the functions from Lab 3.
5. Implement the printing of the list of agents. Write a separate function for this.

## Part B: A small experiment (bonus!)

The below is a bit more advanced, especially from the second question onwards. Use your group and Discord to get help, but make sure you learn from the process and do not just copy someone else's code. This is really good exercise if you can manage the below! But it can be treated as a bonus part of this lab, so don't worry if you can't manage question 2 onwards just yet!

1. Write a function that takes as input a list of agents and that returns the index of a random agent. Use randint() from the random library and the len() function to get an appropriate index value. Think very carefully about possible values for the first and last entry in the list.
2. Write a function that randomly selects two agents, and have them communicate to each other. You can calculate $\mu$ using the function you wrote in Lab 1. Calculate $a = |x_i - x_j|$, so $a$ is the absolute difference between the opinions of the two agents.Then, use an if-then-else statement based on Equation 1 in Chen and Lan (2021), where $i$ is the first agent, whose opinion we are going to update, and $j$ is the second agent. Equation 1 then states that the new value of $x_i$ for the first agent should be the same as the old value (not change) when $a$ is greater than $\varepsilon$ and less than or equal to $\tau$; that $x_i$ should become $x_i + \mu(x_j - x_i)$ when $a$ is less than or equal to $\varepsilon$; and that $x_i$ should become $x_i - \mu(x_j - x_i)(1 - |x_i|)$ when $a$ is greater than $\tau$. Make sure that the calculated new value for $x$ is stored in the list. ($||$ means to take the absolute value, using abs() in Python.)
3. Add an item to the menu (e.g. "(C)ommunicate") that calls this new function.
4. Play around by creating a list of just 5 agents, printing the list, have them communicate a bunch of times, and printing the list again. Are changes reasonable?