1 Assignment

In this assignment, you are only allowed to use loops, counters, if-statements and print statements (comments are allowed). All the below exercises require you to create some Python code that will match the output pattern shown in the particular question. **Please pay attention to the following:**

1. If in doubt how the pattern looks, check with the pdf version of the assignment (which should be in your assignment directory, otherwise you can find it on Quercus).

2. The goal is to create these patterns with one or more loop statements (including loops that use the range function, and while loops) - if in doubt, go back to the previous chapters.

3. You are not allowed to use string manipulation.

4. **Your print statements can only be selected verbatim from the following three options:**

```python
print(f"X", end="") # this will print a single X without a linefeed
print(f" ", end="") # this will print a single blank space without a linefeed
print("") # this will print a linefeed
```

5. **You need to use these statements exactly as shown e.g., so you cannot modify them**

```python
print(f"X", end="")
print("X")  # not ok
print("XX") # not ok
print("XX X") # not ok
print("   ") # not ok
```

6. **You cannot put three print statements if the instructions state that your code cannot contain more than two!** However, you can have a print statement within a loop, and it does not matter how often you execute the loop. It will still count as a single statement.

1.1 Learning outcomes

- Hone your problem-solving skills in an environment with strict rules and a limited set of tools
- Practice the application of the previously introduced problem-solving techniques.
- Reflect on each question and what tools you used to solve them. This will help you to solve the next problem. In other words each task contains the clue to the next problem.

2 Tasks

1. Write a program that uses only one print statement (out of the allowed 3) to create this output:

   ```
   XXXXX
   ```

   ```
   #
   ```

2. Write a program that uses only two print statements out of the allowed 3) to create the output below.

   ```
   XXXXX
   XXXXX
   XXXXX
   XXXXX
   XXXXX
   ```

   ```
   #
   ```

3. Write a program that uses only two print statements (out of the allowed 3) to create the output below. If you can't see the immediate solution go to problem #2, which is the same question but stated more simply.

   ```
   XXXXX
   XXXX
   ```

```
XXX
XX
X
```

```
#
```

4. Write a program that uses only two print statements (out of the allowed 3)to create the output below. Initially, this may look puzzling. But you probably recognize that you solved an analogous problem in #3.

```
X
XX
XXX
XXXX
XXX
XX
X
```

```
#
```

5. Write a program that uses no more than the three print statements mentioned above to create the following pattern.

```
XXXXX
 XXX
  X
```

```
#
```

6. Similar to 5, write a program that creates the following pattern with no more than 3 print statements

```
  X
 XXX
XXXXX
 XXX
  X
```

```
#
```

7. Similar to 6, write a program that creates the following pattern. Your code can contain up to five print statements. Think of it as a puzzle involving loops and counters. While it looks complicated at first, try and decompose it into pieces you have solved before (in fact, most of the code for this piece has been used before). Once you have some ideas, create a game plan, and solve it step by step (say. e.g., start with the left side of the figure. . .).

```
X            X
 XX        XX
  XXX    XXX
   XXXXXXXX
   XXXXXXXX
  XXX    XXX
 XX        XX
X            X
```

2.1 Marking Scheme

Total points = 28 points.

- 4 points for each solution that follows the rules and produces the requested output. 2 points for each solution, which is at least 50% correct. (28 pt)

2.2 Submission Instructions

Create a new (or copy and existing) notebook in your `submissions` folder before editing it. Otherwise, your edits may be overwritten the next time you log into syzygy. Please name your copy `A#-assignment-name-firstname-lastname`

- Where `A#` is something like A1, A2 A3, etc.
- Replace the `assignment-name` with the name of the assignment (i.e., the filename of the respective Jupyter Notebook). This can be shortened as long as it remains clear which assignment it is.

- `firstname-lastname` with your own name.

Note: If the notebook contains images, you must also copy the image files! Your notebook/pdf must start with the following lines :

ESS245: Assignment Title

Date:

First Name:

Last Name:

Student: Id

Before submitting your assignment:

- Check the marking scheme and ensure you have covered all requirements.
- Re-read the learning outcomes and verify that you are comfortable with each concept. If not, please speak up on the discussion board and ask for further clarification. I can guarantee that if you feel uncertain about a concept, at least half the class will be in the same boat. So don't be shy!

To submit your assignment, you need to download it as `ipynb` notebook format **and** `pdf` format:

- **Do not use File -> Save and Export as -> pdf!!!**
- You **must** use your **browser's print function** (`Ctrl-P`, or `Cmd-P` on a Mac) and then select `Save as pdf`. In the past, this worked best with Chrome or Firefox.

Please submit **both files** on Quercus. Note that the pdf export can fail if your file contains invalid markup/python code. So you need to check that the pdf export is complete and does not miss any sections. If you have export problems, don't hesitate to contact the course instructor directly.

Notebooks typically have empty code cells in which you must enter python code. Please use the respective cell below each question, or create a python cell where necessary. Add text cells to enter your answers where appropriate. Your responses will only count if the code executes without error. It is thus recommended to run your solutions before submitting the assignment.

Note: Unless specifically requested, do not type your answers by hand. Instead, write code that produces the answer. Your pdf file* should show the code and the results of the code execution.*