Hand-in nr.1

The hand-ins in the course will consist of two parts:

1) a set of short exercises to practice a particular part of the curriculum (e.g., loops), and

2) a project part. In the project part, we will consider a specific Data Science case, working on the same (or similar) data each week and gradually building up a complete analysis of the data.

Note that the tools required to solve the hand-ins will be generally covered in the lectures in the week where the assignment is given. Sometimes, you might have to wait until the Friday lecture before you have all the tools to complete the assignment.

Before we begin, please create a new project in Thonny for your hand-ins.

We recommend you use the same project for all hand-ins in the course since this will make it slightly easier for the project part (so you don't have to copy the data files each week).

Part 1

This week, we'll look into print statements, if-else statements and loops.

1. Create a new file called handin1\_1.py. Put the following piece of code in the file.
2. if 2<4

print "Hello, world!"

There are two errors in the code. Fix the errors so the code prints "Hello, world!" to the screen.

1. Create a new file called handin1\_2.py. In this program, you should create a variable called message and assign the string "Hello, world!" to this variable. The program should then print out the length of the string contained in the variable. Finally, it should open an output file called message.txt, and print the string to this file. (len(message)
2. Create a new file called handin1\_3.py. In this file, create a variable called number\_str, and assign the following string to this variable: "1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20\n". We will process this string in a sequence of steps:
   1. Use a string method to remove the newline character at the end of the string and save the result in a variable called number\_str\_cleaned. (replace (“”))
   2. Use another string methods to split the string into a list of small strings, where each element is a string containing a single number. Save the result in a variable called number\_str\_list.
   3. Write a for-loop that iterates over this list and prints out the numbers on the screen so each line contains a single number.

Part 2: Project

The project we will work on throughout the course will be around climate data. In particular, we will analyze the [Land\_and\_Ocean\_summary.txt](https://absalon.ku.dk/courses/59965/files/6623750?wrap=1)[Download Land\_and\_Ocean\_summary.txt](https://absalon.ku.dk/courses/59965/files/6623750/download?download_frd=1)data file. This file comes from the Berkeley Earth website ([http://berkeleyearth.org/data/ Links to an external site.](http://berkeleyearth.org/data/)). It contains global temperature anomalies, telling us how much colder or warmer it is compared to a reference value, which is chosen to be the average over the time period between 1951 and 1980. The header of the file contains further details.

In this first hand-in, we will start by reading the data into a Python list and then process the data in a few ways. Start by downloading the Land\_and\_Ocean\_summary.txt file and copying it to the directory in which your code resides (i.e., the project directory in PyCharm).

1. Create a file called handin1\_project.py. Inside this file, write code that reads the Land\_and\_Ocean\_summary.txt into a list of strings called list\_of\_lines, where each element in the list corresponds to a line in the file. This first exercise will include all lines (including comments).
2. In the same handin1\_project.py file, write a loop that iterates over the list\_of\_lines list. For each line, follow the steps in Q3 above:

1) remove the newline,

1. In the same handin1\_project.py file, write a loop that iterates over the list\_of\_lines list. For each line, follow the steps in Q3 above: 1) remove the newline, 2) split the string into small pieces. If you look at the first element in this list, you will see that it contains either a character that denotes a comment line, or--if actual data--the year in which this datapoint was recorded. Now print out the full line (in the original format, with white space at the beginning and end), but only if it is a data entry and if the year value is larger than or equal to 2000 (meaning you should skip the comment lines, which have no year value at all). Make sure that in your output, there are no empty lines. Hint: note that when you use print(), it automatically adds a new line (\n)character to the end of the line. If the line already contains a newline at the end, the two newlines will result in an empty line being printed after each line. You can tell print not to add a newline character by using the end="" option - e.g. print(line, end="")

When you are done, click on the "Load Handin1 in a new window" button below, which will take you to the CodeGrade server. Here, please submit the handin1\_1.py, handin1\_2.py, handin1\_3.py and handin1\_project.py files. CodeGrade will then automatically check the code for you and upgrade your grade for the assignment within Absalon. You can submit as many times as you want.

Øverst på formularen

This tool needs to be loaded in a new browser window

Nederst på formularen