Exercises – Part 1: Data Types, Control Structures, Loops and Functions

Do not use built-in functions except when stated otherwise! Name your functions exactly like in the specification. Upload your code as a .py file, e.g. **`exercises_part_1.py`**, on Moodle.

Do not copy and paste code from others or the internet! Find your own solution, there are many different correct ways to complete each exercise.

Exercise 1: Famous Quote

Find a quote from a famous person you admire. Write a function named **`famous_quote()`** that prints the quote and the name of its author.

Your output should look something like the following, including the quotation marks:

```
Albert Einstein once said, “A person who never made a mistake never tried anything new.”
```

Exercise 2: Number Eight

Write a function called **`number_eight()`**, that uses addition, subtraction, multiplication, division and exponentiation operations that each result in the number 8. Be sure to enclose your operations in **`print()`** calls to see the results. Your function should include five lines that look like this:

```
print(5+3)
```

Exercise 3: Formatting

Write a function **`name_age()`** that accepts the name and age of a person. It then prints it on the console in the following format:

```
Hello, NAME. You are AGE years old.
```

The function does not return anything. Use the **`input()`** function to read the name and age as a string and an integer. Use at least three different methods to format the output, e.g. concatenation, String **`format()`**, f-strings, … That is, print the greeting three times using different formatting methods which produce the same output.

Your output should look something like the following:

```
Hello, Georg. You are 46 years old.
Hello, Georg. You are 46 years old.
Hello, Georg. You are 46 years old.
```

Exercise 4: Swap

Write a function **`swap_integers()`** that reads two integers from the user and prints them on the console. Then the function swaps the integers in memory and prints the swapped integers again on the console. If, for example, the first integer is **`x=10`** and the second is **`y=20`**, **`x`** must have the value **`20`** after the swap (Hint: you can use a temporary variable). The function does not return anything.

Example output:

```
Please enter x: 10

Please enter y: 22

x=10

y=22

After swap:

x=22

y=10
```

Exercise 5: Modulo check

Write a function **`check_numbers(number1, number2)`** that accepts two arguments. The function checks if any of the numbers is divisible by 6 and if both are divisible by 10. The function does return true if both conditions are true. Hint: Use the modulo operator.

Example output:

```
Number 1: 6

Number 2: 10

One number is divisible by 6

Both numbers are not divisible by 10
```

Exercise 6: Summarizer

Write a function **`sum_up(number1, number2)`** that accepts two integers and sums up every integer between the two numbers including the given integers (inclusive). Check if the second number is greater than the first and display a message if it's not. The function returns the result as an integer.

Example: if the function is called with 3 and 9, that is, **`sum_up(3, 9)`** the result is 42 because:
3 + 4 + 5 + 6 + 7 + 8 + 9 = 42

Exercise 7: Sequencer

Write a function **`sequence(number)`** that accepts an integer as argument. It then checks if the given number is an integer between 0 and 9 (inclusive) and prints an error message if it's not. In case the given number is between 0 and 9, the function prints the sequence of number from 0 to 9 on the console without the given number. The function does not return anything.

For **`sequence(5)`** your output should look like the following:

```
0 1 2 3 4 6 7 8 9
```

Exercise 8: String check

Write a function **`check_string(text)`** that accepts a string and checks if it begins OR ends with the character "a". Use built-in string methods of python. A list with all methods can be found here: https://docs.python.org/3/library/stdtypes.html#string-methods

The function returns **`True`** if the string begins or ends with an "a". The function should work for lower and upper case strings.

Exercise 9: ASCII Art

Write a function **`triangle(rows)`** that accepts an integer and prints out a triangle of stars with spaces in between them with the height of the given integer.

Example: **`triangle(4)`** would result in a triangle with 4 rows:

The function does not return anything.