QP03: Program flow, conditionals & iteration

Bachelor in Informatics and Computing Engineering
Programming Fundamentals
Instance: 2024/2025

***Goals**: write programs using conditionals and iteration with for, while, break and continue*

***Rules**: you may work with colleagues, however each student must write and submit their own work*

***Assignment**: read the questions and start answering using VS Code and the Python interpreter; for each finished answer, go through the public tests in Moodle using "Pre-check" ("Verificar"); to see the results of the private tests use "Check" ("Submeter") but beware as there's a penalty regime of "0, 0, 0, 0, 10, 20, 30, ...*

***Submission**: Note that you are entitled to ONLY ONE attempt*

***Review**: Later in the semester, in addition to the submitted answers and the marks, the private tests and the code used to correct the answers will also be visible*

1. Multiplication table

Write a Python program that takes an integer **`num`**, provided by the user, and outputs the multiplication table with lines: **`num x index = res`**, where **`res = num * index`** and **`index`** is a number that goes from 1 to a maximum of 10.

When the number is multiplied by itself, the displayed format should be **`num ^ 2 = res`** and the table is finished immediately.

For example for **`num = 5`** the output is:

```
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 ^ 2 = 25
```

2. Hour converter (24h to 12h)

Write a Python program that reads two integers, representing the hours (in the interval [0, 24[) and minutes (in the interval [0, 60[), using the 24-hour format and prints the equivalent hour in the 12-hour format.

The 12-hour clock is a time convention in which the 24 hours of the day are divided into two periods, "am" and "pm", each of 12 hours, numbered: 12 (acting as 0), 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 and 11 (en.wikipedia.org/wiki/12-hour_clock).

The minutes are always represented with two digits and, if the minutes' value is zero, it should not appear in the output (see the examples available in the public tests).

If the hours and minutes values are incorrect, the program must print the message "INVALID DATE FORMAT".

Hint: you can try to use formatted string literals (f-strings) to correctly print leading zeros.

3. Reverse integers

Write a Python program that, given a positive integer **`num`** by user input, computes its reverse (the number with the digits by the reverse order) and prints it.

You are not allowed to use string manipulation (for example string concatenation).

For example:

- for num=`766`, the output is `667`
- for num=`789`, the output is `987`
- for num=`45654`, the output is `45654`

4. Draw a square

Write a Python program that prints a square filled with the `#` character for a given dimension greater or equal to 3. The square centre must be filled with either one or four zeros (`0`), depending on whether the dimension is odd or even.

For example for dimension 4 the output is:

```
####
#00#
#00#
####
```

and for dimension 5 the output is:

```
#####
#####
##0##
#####
#####
```

5. Join digits two numbers

Write a Python program that, given two numbers `n1` and `n2` provided by the user, produces a new number `result` by joining the last digit of `n1` (i.e. the least significant) with the last digit of `n2`, and so on for the other digits.

The number of digits of `n1` and `n2` is the same.

For example:

- if the numbers given are `n1=123` and `n2=456`, the resulting number is `362514`.