Computational Thinking and Programming – A.Y. 2022/2023

Written examination – 07/09/2023

Given name: _____

Family name: _____

Matriculation number: _____

University e-mail: _____

Enrolment a. year: [ ] 2022/2023 [ ] 2021/2022 [ ] 2020/2021 [ ] 2019/2020 [ ] other

Is it your first try? Yes | No

The examination is organised in three different sections:

- Section 1: basic questions [max. score: 16]. It contains four simple questions about the topics of the whole course. Each question requires a short answer. Each question answered correctly will give you 4 points (or less for partial answers).
- Section 2: understanding [max. score 8]. It contains an algorithm in Python, and you have to report the particular results of some of its executions according to specific input values.
- Section 3: development [max. score 8] It describes a particular computational problem to solve, and you are asked to write an algorithm in Python for addressing it.

You have 1 hour and 30 minutes for completing the examination. By the final deadline, you should deliver only the original text (i.e. this document) with the definitive answers to the various exercises that must to be written with a pen – pencils are not permitted. You can keep all the draft papers that you may use during the examination for your convenience – blank sheets will be provided to you on request.

Section 1: basic questions

1 – Which of items in the following lists are **not** high-level programming languages?

- Python
- Assembly
- Javascript
- Fortran
- Machine language

2 – Consider the following Python function:

```
def f(x, y):
    if y != 0:
        return x / y
    else:
        return x / (y + 1)
```

What is the value returned by `f(4, 0)`?

3 – Write down a small function in Python that takes in input a string returns `None` if the string contains less than three characters, otherwise it returns `True` if the third character in the string is a vowel (and `False` otherwise).

4 – Introduce the *greedy algorithm* approach and explain which main characteristics a computational problem must have for using a greedy algorithm to solve it successfully.

Section 2: understanding

Consider the following functions written in Python:

```
def t(given_name, mat_string):
    res = 0

    mat_len = len(mat_string)
    for i in range(mat_len):
        sx = int(mat_string[i])
        dx = int(mat_string[mat_len - i - 1])

        if sx < dx:
            n = dx - sx
        else:
            n = sx - dx

        res = res + n

    res_s = given_name[res % len(given_name)]
    res_b = res_s in "aeiou"

    return res_s, res_b
```

Consider the variable `my_mat_string` containing the string of all the ten numbers in your matriculation number (e.g. `"0000123456"`), and the variable `my_given_name` containing string of your given name all in lowercase. What is the value returned by calling the function `gcs` as shown as follows:

```
t(my__given_name, my_mat_string)
```

Section 3: development

The **Sørensen–Dice coefficient** is a statistic used to gauge the similarity of two samples, that was intended to be applied to discrete data. Given two sets, A and B, it is defined as twice the number of elements common to both sets divided by the sum of the number of elements in each set, as defined in the following formula:

$$\frac{2*|A \cap B|}{|A|+|B|}$$

Write an algorithm in Python – `def sd_coeff(s1, s2)` – which takes in input two sets and returns the number defining the Sørensen–Dice coefficient for those sets.