Computational Thinking and Programming – A.Y. 2021/2022

Written examination – 28/01/2022

Given name: ____

Family name: ____

Matriculation number: ____

University e-mail: ____

Enrolment a. year: [ ] 2021/2022 [ ] 2020/2021 [ ] 2019/2020 [ ] 2018/2019 [ ] 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 the following numbers is a possible result returned by the Fibonacci function?

- 2
- 4
- 6
- 8
- 10
- 12

2 – Consider the following snippet of Python code:

```
def f(s):
    if len(s) > 3:
        return s[3:]
    else:
        return s[:3]
```

Which value is returned by calling the function above as follows: `f("bod")`?

3 – Write down a small function in Python that takes in input two positive integers and returns the result of the division of the smaller one over the greater one.

4 – Describe, at a general level, the approach used by the merge sort for addressing the related computational problem.

Section 2: understanding

Consider the following functions written in Python:

```
def c_rec(chars, mat_list):
    result = ["a", "e", "i", "o", "u"]

    if len(mat_list) == 0:
        result = sorted(list(chars))
        return "".join(result)
    elif mat_list[0] % 2 == 0:
        idx = mat_list[0] % len(result)
        chars.add(result[idx])

    return c_rec(chars, mat_list[1:])
```

Consider the variable `my_mat_list` containing a list of integers where each number is a digit of your matriculation number (e.g. `[0, 0, 0, 0, 1, 2, 3, 4, 5, 6]`), and the variable `my_chars` containing the set with all the alphabetic characters in lower case included in your full name (i.e. for the name John Doe we have the set `{"j", "o", "n", "h", "d", "o", "e"}`). What is the value returned by calling the function `c_rec` as shown as follows:

```
c_rec(my_chars, my_mat_list)
```

Section 3: development

The **Quick Gestalt Pattern Matching** is a statistic used for comparing the similarity of two strings. In particular, given two strings *S* and *T*, the measure is calculated by considering twice the number K_m of all used characters in *S* which occur in *T* divided by the sum of the number of characters in *S* and *T*:

$$\frac{2 * K_m}{|S| + |T|}$$

Write an algorithm in Python – `def qgpm(s, t)` – which takes in input two strings and returns the Gestalt Pattern Matching statistic.