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

Written examination – 11/01/2023

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 – Being the variable *test* a list, which of the following Python codes are valid instructions?

- `test.append(1)`
- `test.add(1)`
- `test.update([1, 2, 3])`
- `len(test)`
- `test["a"] = 1`
- `test["a"]`

2 – Consider the following Python function:

```
def f(a, b):
    r = ""
    for x in b:
        r = r + a
    return r
```

What is the value returned by `f("x", [3, 4, 5])`?

3 – Write down a small function in Python that takes in input a negative number and returns the related positive number (e.g. it returns `2` if the input is `-2`).

4 – Write down the name and describe the main steps of the algorithmic technique adopted in the implementation of the merge sort.

Section 2: understanding

Consider the following functions written in Python:

```
def sstr(family_name):
    if len(family_name) > 1:
        m = len(family_name) // 2
        l_name = family_name[0:m]
        r_name = family_name[m:len(family_name)]
        return sstr(l_name) + sstr(r_name)

    r = -1
    v = "aeiou"
    for idx, c in enumerate(v):
        if family_name == c:
            r = idx

    return r
```

Consider the variable `my_family_name` containing string of your family name all in lowercase with no spaces. What is the value returned by calling the function `sstr` as shown as follows:

```
sstr(my_family_name)
```

Section 3: development

Letter frequency is the number of times letters of the alphabet appear on average in written language. It is possible to have a *frequency sequence* of a language, i.e. the use of letters showing trends in related letter frequencies, by returning the sequence of letters from the most frequent one to least frequent one. For instance, considering the following simple text

> *Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do*

has the following letter frequencies:

```
'n': 10  'i': 9'   't': 9   'e': 8   'o': 7    'g': 6
'a': 5   's': 4    'r': 4   'h': 4   'b': 3    'd': 3
'v': 2   'y': 2    'f': 2   'l': 1   'c': 1    'w': 1   'k': 1
```

and the frequency sequence is represented by the string `"niteogasrhbdvyflcwk"`, where no punctuation and other non-letters are included. It is worth mentioning that, in the frequency sequence, letters having the same frequency are ordered according to their first occurrence in the input text – e.g. `'l'` comes before `'c'` because the first occurrence of the first letter happens before the first occurrence of the second one (in the word "A***l****i****c***e"). In addition, the input text is considered as lowercase when counting the frequencies.

Write an algorithm in Python – `def sequence(s)` – which takes in input a string `s` representing a text, and returns another string representing the fingerprint of such an input string.