Computational Thinking and Programming – A.Y. 2020/2021

Written examination – 15/03/2021

Given name: ______________________________

Family name: ______________________________

Matriculation number: ______________________________

University e-mail: ______________________________

Group name: ______________________________

Is it your first try? Yes | No

The examination is organised in three different sections:

- Section 1: basic questions [max. score: 8]. 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 either 2 points (full answer) or 1 point (partial answer).
- Section 2: understanding [max. score 4]. 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 4] 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 sentences are true statements?

- Noam Chomsky designed the Bombe in 1940
- The Fibonacci number can be computed using a dynamic programming approach
- The Fibonacci number can be computed using a non-recursive approach
- Divide and conquer is an algorithmic technique that abandons partial candidates once it is clear that they will not be able to provide a solution to the problem
- A set is a countable sequence of ordered and repeatable elements

2 – Consider the following Python implementation of the mergesort algorithm:

```
def merge_sort(input_list):
    input_list_len = len(input_list)
    if input_list_len == 0:
        return input_list
    else:
        mid = input_list_len // 2
        left = merge_sort(input_list[input_list_len:mid])
        right = merge_sort(input_list[mid:input_list_len])
        return merge(left, right)
```

The code above contains two mistakes. Please, identify them and propose corrections.

3 – Write down a small function in Python that takes in input two boolean values and implements the *xor* operation, which returns `True` only when one of the input boolean is `True` and the other is `False`, and returns `False` otherwise.

4 – Which machines developed Charles Babbage and what were their main differences.

Section 2: understanding

Consider the following functions written in Python:

```
from collections import deque


def cv(full_name, mat):
    result = dict()

    nq = deque()
    for d in mat:
        n = int(d)
        if n > 0:
            nq.append(int(n))

    vs = deque()
    idx = 0
    for c in full_name:
        if c in "aeiou":
            idx = idx + 1
            result[idx] = c
            if c not in vs:
                vs.append(c)

    while len(vs) != 0 and len(nq) != 0:
        e = vs.pop()
        i = nq.popleft()

        if i in result:
            result[i] = result[i] + e

    return result
```

Consider the variable `my_mat_string` containing the string of your matriculation number and the variable `my_full_name` containing the string of your full name (i.e. given name and family name separated by a space) but in lower case. What is the value returned by calling the function `cv` as shown as follows:

```
cv(my_full_name, my_mat_string)
```

Section 3: development

The Erdős number describes the collaborative distance between mathematician Paul Erdős and another person, as measured by authorship of scholarly articles. In practice, having the collaboration network of some people described as an undirected graph, where each node represent a person and an edge between two people states that they have coauthored some article together, the goal is to find the minimal distance, computed as number of edges to traverse, between the node representing Paul Erdős and another input person.

An **Erdős number by research group (ENG)** is the average number computed by dividing the Erdős numbers of each member of the research group by the number of people in that group.

Write an algorithm in Python – `def eng(coauthor_graph, research_group)` – which takes in input an undirected graph `coauthor_graph` (i.e. an object of the class `networkx.Graph`) describing a collaboration network, in which each node is defined by the string of the name of a person and that includes the node `"Paul Erdős"`, and the list of strings `research_group`, which contains the strings of the names of people that are member of a research group. The Python function should return the related Erdős number by research group.