Computational Thinking and Programming – A.Y. 2019/2020

First partial written examination (02) – 11/11/2019

Given name: _____

Family name: _____

Matriculation number: _____

University e-mail: _____

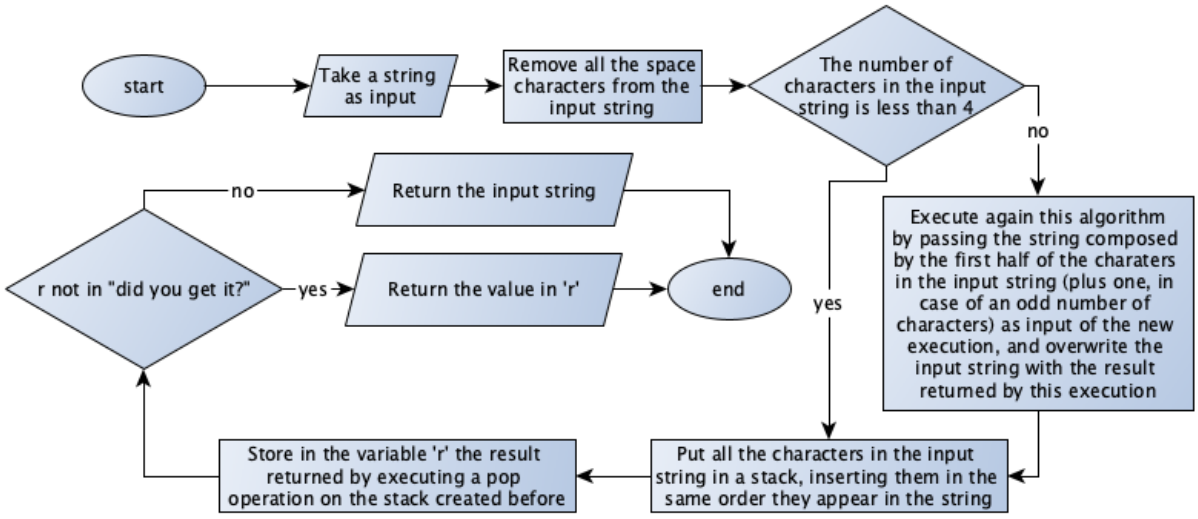
Please answer to the following 5 questions [40 minutes max, 1 point each, max score: 5 points]

1. Highlight whether the data structures in the table are compliant or not with the characteristics shown.

Data structure	Countability	Ordering	Repeatability
Dictionary	Yes / No	Yes / No	Yes / No
Queue	Yes / No	Yes / No	Yes / No
Stack	Yes / No	Yes / No	Yes / No

2. Describe what is the form of the production rules of a context-free grammar.

3. What is the output returned by executing the algorithm in the following flowchart if you use your family name as input string? Note: `"did you get it?"` in the second decision widget is a string, and remember that the input string can be modified during the execution of the algorithm.



4. The variables `my_gn_list` and `my_fn_list` store the lists of the characters, excluding any space, of your given name and family name in lowercase. Also, consider the last number (i.e. the rightmost) of your matriculation number as stored in the variable `my_number`. Write down the result of the execution of the following function passing `my_gn_list`, `my_fn_list` and `my_number` as input (i.e. `f(my_gn_list, my_fn_list, my_number)`).

```
def f(gn_list, fn_list, cur_number):
    all_numbers = list()
    for i in range(cur_number):
        if i < len(fn_list):
            all_numbers.append(i)

    idx = -1
    while True:
        idx = idx + 1
        if idx < len(gn_list):
            cur_char = gn_list[idx]
            for n in all_numbers:
                if cur_char == fn_list[n]:
                    return (cur_char, n)
```

5. Write the body of the Python function `def do_it(string, number)` that takes a *string* and a *number* in input, and returns the string `"Oh no!"` if the number of characters that are consonants (i.e. that are not “a”, “e”, “i”, “o”, “u”) in *string* is less than *number*, otherwise it returns a queue containing all the consonants in *string*. In this latter case, the consonants are inserted in the queue in the same order they appear in *string*. Example of execution:

```
my_string = "yes mama"
my_number = 3
do_it(my_string, my_number) returns deque(["y", "s", "m", "m"])
```