Fall 2020
Computer Science I — Maíra Marques Samary PhD
Section 5 – Tuesday and Thursdays – 12:00PM – Fulton 250

TA's Office Hours – online

James Monahan - monahajm@bc.edu - https://bccte.zoom.us/j/2822792652
Tuesdays 7:00 PM – 8:00 PM
Wednesdays 4:00 PM – 5:00 PM

Jennifer Joseph - - josephjz@bc.edu - https://bccte.zoom.us/j/5882755193
Wednesdays 11:00 AM – 12:00 PM
Thursdays 3:00 PM - 4:00 PM

Liam Murphy- - murpaue@bc.edu - https://bccte.zoom.us/j/3085424208
Tuesdays 2PM-4PM

Discussion Groups:
CSCI100701 - Tuesday 6:00 PM – 6:50 PM- Fulton Hall 220 (James Monahan)
CSCI100702 - Thursday 5:00 PM – 5:50 PM – Fulton Hall 220 (Jennifer Joseph)
CSCI100703 - Wednesday 4:00 PM – 4:50 PM – Gasson Hall 203 (Liam Murphy)

Assignment 6

Due date – 11/19/20 11:59 PM

General Instructions

Create a folder named ***LASTNAME_FIRSTNAME***. You will populate the folder with **ALL** of the .py files you write for this homework. To submit the homework, verify the folder includes all your .py files, compress (zip) the folder then upload to Canvas. Remember to include the following comments at the **top of each** of your .py files:

```
# author:
# assignment:
# description:
```

What to submit in Canvas?

Make sure all your files are saved in the folder LASTNAME_FIRSTNAME, then compress (zip) the folder and upload to Canvas.

If you encounter any problems in completing the assignment or in the submission process, please don't hesitate to ask for help. The sooner, the better!

Problem 1

Create a **recursive** function that must be called num_nines, that receives a number as a parameter (y). The function must return the number of times the number 9 appears as a digit on y.

Examples of prints:

print(num_nines(3)) → 0
print(num_nines(7)) → 0
print(num_nines(9)) → 1
print(num_nines(99999)) → 5
print(num_nines(345678)) → 0
print(num_nines(397896599)) → 4

Problem 2

Write a **recursive** function called missing_digits that takes a number x that is sorted in increasing order (for example, 12289 is valid but 15362 and 98764 are not). Your function should return the number of missing digits in x.
A missing digit is defined in this case as a number between the first and last digit of x, that is not in x.
In this problem, you can assume that the input/parameter will be always correct, meaning in increasing order.

Examples of prints:

print(missing_digits(12289)) → 5
(answer is 5, because it is missing 3,4,5,6,7)
print(missing_digits(3459)) → 3
(answer is 3, because it is missing 6,7,8)

Problem 3

Write the **recursive** function called partials, that receives two parameters.

Partials is a part of the original number (not necessarily contiguous). For example, the numbers 12345 has partials that include: 123, 234, 124, 245, etc. Your function is to get the **maximum** number of partials that exists below a certain length.

The parameters of this partials function are the number (n) and the length (l).

Examples of prints:

print(partials(20125,3)) → 225
print(partials(20125,5)) → 20125
print(partials(20125,6)) → 20125
print(partials(12345,0)) →0
print(partials(12345,1)) → 5

One detailed example:
For example, for n = 20125 and l = 3, the partials are

2
0
1
2
5
20
21
22
25
01
02
05
12
15
25
201
202
205
212
215
225
012
015
025
125

and of these, the maximum number is 225, so our answer is 225.

Problem 4

Consider information about Marvel characters in the file called marvel.csv. This file contains 7 pieces of information related to the Marvel characters:

name,

ID (if this name is public, secret, or dual)

ALIGN (good or bad)

ALIVE

APPEARANCES (number of appearances of the character in comics)

FIRST APPEARANCE (day and month of first appearance)

Year (year of first appearance)

The information is separated by a comma.

Write a function that will read the file marve.txt, and that it will Create a dictionary where the alignment of the character name is the key (good, bad or neutral) and all the other information are list of values for its name.

Resulting dictionary:

{'Good Characters': [['Spider-Man (Peter Parker)', 'Secret Identity', 'Living Characters', '4043', 'Aug-62', '1962'], ['Captain America (Steven Rogers)', 'Public Identity', 'Living Characters', '3360', 'Mar-41', '1941'], ['"Iron Man (Anthony ""Tony"" Stark)"', 'Public Identity', 'Living Characters', '2961', 'Mar-63', '1963'], ['Thor (Thor Odinson)', 'No Dual Identity', 'Living Characters', '2258', 'Nov-50', '1950'], ['Benjamin Grimm', 'Public Identity', 'Living Characters', '2255', 'Nov-61', '1961'], ['Reed Richards', 'Public Identity', 'Living Characters', '2072', 'Nov-61', '1961'], ['Hulk (Robert Bruce Banner)', 'Public Identity', 'Living Characters', '2017', 'May-62', '1962'], ['Jonathan Storm', 'Public Identity', 'Living Characters', '1934', 'Nov-61', '1961'], ['Henry McCoy', 'Public Identity', 'Living Characters', '1825', 'Sep-63', '1963'], ['Susan Storm', 'Public Identity', 'Living Characters', '1713', 'Nov-61', '1961']],
'Neutral Characters': [['"Wolverine (James ""Logan"" Howlett)"', 'Public Identity', 'Living Characters', '3061', 'Oct-74', '1974'], ['Scott Summers', 'Public Identity', 'Living Characters', '1955', 'Sep-63', '1963'], ['Namor McKenzie', 'No Dual Identity', 'Living Characters', '1528', '', '']]}

In the file that you are receiving you only have 2 characters' alignment, but we will test with files with more than just these 2 character alignments. So, your code must not create the key in a hard-coded way (you cannot do something like == "Good Characters" – this is hard code).

RUBRIC

Problem Number	Items to be evaluated	Excellent (100% of points))	Average (60% of points)	A Little below average (40%)	Needs Improving (30% of points)	Possible Points
1	A. Recursive function B. Function created according to what was asked (name and number of parameters) C. The function return exactly what was supposed to return D. The function gives the expected results for small and big numbers.	All items correct	If you failed 1 item in the list between the options B, C or D	You failed 2 items between the options B, C or D	You have the options B, C and D right but you don't have a recursive function (failed item A)	**2.5**
2	A. Recursive function B. Function created according to what was asked (name and number of parameters) C. The function return exactly what was supposed to return D. The function gives the expected results for small and big numbers.	All items correct	If you failed 1 item in the list between the options B, C or D	You failed 2 items between the options B, C or D	You have the options B, C and D right but you don't have a recursive function (failed item A)	**2.5**
3	A. Recursive function B. Function created according to	All items correct	If you failed 1 item in the list between the options B, C or D	You failed 2 items between the options B, C or D	You have the options B, C and D right but you don't have a recursive	**2.5**

	what was asked (name and number of parameters) C. The function return exactly what was supposed to return D. The function gives the expected results for small and big numbers.				function (failed item A)	
4)	A. Function created according to what was asked (name and number of B. Correct dictionary	All items correct	Minor mistakes in the dictionary creation Or partial item A		Item A correct, but only created a bunch of lists of lists with the character alignment.	**2.5**
FINAL SCORE						**10**