SENECA COLLEGE OF APPLIED ARTS AND TECHNOLOGY
FACULTY OF CONTINUING EDUCATION
FINAL EXAMINATION-2214

SUBJECT NAME:	**Introduction to Python**
SUBJECT CODE:	**PRO675S1M**
INSTRUCTOR NAME:	**Morteza Kiadi**
DUE DATE:	**August 14, 2021-1:00 PM**
TOTAL MARKS:	**100**
WEIGHTING:	**40%**

SPECIAL INSTRUCTIONS:

Exam Books:	Required:	Not Required: X
Exam Aids:	Permitted: X	Not Permitted:
Exam Question Paper:	Returned:	Not Returned: X

Approved by:

Sheri Ladoucier

_____

Sheri Ladoucier, Academic Program Manager

Academic Policy Section 9:
Engaging in any form of academic dishonesty to obtain any type of advantage or credit is an offence and will not be tolerated by the College. Such offences under this policy include, but are not limited to, cheating, plagiarism, falsification, impersonation, misrepresentation and procurement.

NOTES:

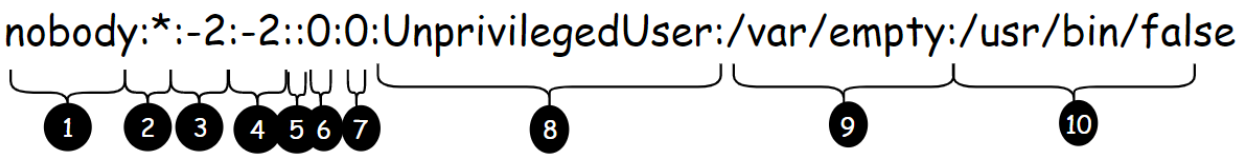
1) You need to submit answers to all questions in one Jupyter notebook file(10 marks). Make sure you use the relative paths in your code when you do any file operation (not absolute path) and always use the current directory of the notebook file for any file operation (10 marks).

2) Please document the comments inside the Jupyter notebook like any instruction that I need to follow to run your code. Lack of documentation and instruction about running your code may cause losing marks.

3) You can use any available Python library.

1- In this question, you read a text file programmatically (10 marks) that it has colons (:) as the delimiter between its data elements. You need to pick specific items in the file (10 marks) and write the result into a .csv file (10 marks). Ultimately, you will read that **.csv** file and show the result on the screen (5 marks). I have provided a text file in the Blackboard. That is a sample Linux password file. The name of the file is the **password.txt** file. Please use that file to test your code. Please consider that if there is no missing data, each row has 10 data elements, separated by colons like:

nobody:*:-2:-2::0:0:Unprivileged User:/var/empty:/usr/bin/false



The first item in each line of this file is the **user name** and the last item shows if that user can get a **shell** or not. If the user does not get shell access, you see **/false** at the end of that line like:

_assetcache:*:235:235::0:0:Asset Cache Service:/var/empty:/usr/bin/**false**

This file is not clean. Sometimes we have missing data elements in the file. For example, we have a line like the following line that does not have the first data element before the first colon is missing (that is the user name):

:*:236:236::0:0:Core Media IO Daemon:/var/empty:/usr/bin

Instance 1: Missing user name

Also, you may have rows that miss more than one data element. For example in the following lines, we do not even have placeholders for 10 elements (we are missing a few colons):

apple:launch:*:239:0:0:_launchservicesd:/var/empty
orange:launch:*:239:0:/var/empty:/usr/bin

Instance 2: Missing placeholder

We are interested to know about the users that have shell access (hence there is no **/false** in their corresponding line). In the output csv file, we will have just two columns, the user name and the shell program they use (like **sh** , **csh,** etc). Make sure your code filters (ignores) the lines if they do not have "both" of user name and shell (i.e we are only interested to show the lines that have both the user name and the shell. If only one of them exists, we ignore that line altogether, like the instance 1 above). Also, make sure you do not include the lines that do not have all the 10 elements or their placeholders (like instance 2 above) (5 marks). So you do not have to include the "orange" user in the output, in the same data set.
The code at the end needs to read the output .csv file to show the result in the notebook (5 marks) (**Total: 45 Marks).**

Sample Output after reading from the CSV file:

	User	Shell Program
0	root	sh
1	_uucp	ksh
2	_krb_changepw	csh

2- In this question, you write a code that creates a sub-folder (5 marks) in the current folder and it generates a random number of files, but not less than 10 files (at least 10 files) (5 marks). In each file, the code picks 20 random words from the dictionary file (5 marks) that is uploaded into the Blackboard. Each word must be in one line. Your code will search through all files and finds the longest word in each file (5 marks) and show a report like the name of the file and its longest word:
File_name: the longest word (5 marks)
Sample output:

```
The Report for the longest word in each file (file name: the longest word)
1.txt: accomplished
10.txt: accomplished
10.txt: professional
11.txt: announcements
12.txt: announcements
13.txt: announcements
14.txt: announcements
15.txt: announcements
16.txt: announcements
2.txt: announcements
3.txt: announcements
4.txt: announcements
5.txt: announcements
6.txt: announcements
7.txt: knowledgestorm
8.txt: knowledgestorm
9.txt: knowledgestorm
```

In the end, it prints the file name(s) that have the longest word(s) among all the generated files (10 marks) **(Total: 35%)**