Module 6 Assignment

Essential Python Packages

# Instructions

Please make a notebook called ‘module6\_assignment’ and put ONLY the following code in it. We will try and consolidate some information from previous lectures. Note that you do not need to perform any input validation in the functions.

## Problem 1

Create a function called the\_next\_15\_days. The function will accept one argument which is a date string in the form of %Y%m%d. Some examples of date strings in this format are ‘20200101’, ‘20210204’, and ‘19831201’. The function will convert this string into a datetime object and then return a list of 15 datetime objects that start on the date that was passed to the function and extend 15 days into the future. Remember to test your function with different date strings.

### Hints

* You will want to use a timedelta object.
* You can use list comprehension, although you do not have to.

Here is an example of how the function should work:

### Example 1

|  |
| --- |
| ***my\_date\_str = "20170817"  result = days\_15(my\_date\_str)  print(result)  # The following should print '[datetime.datetime(2017, 8, 17, 0, 0), datetime.datetime(2017, 8, 18, 0, 0), datetime.datetime(2017, 8, 19, 0, 0), datetime.datetime(2017, 8, 20, 0, 0), datetime.datetime(2017, 8, 21, 0, 0), datetime.datetime(2017, 8, 22, 0, 0),  datetime.datetime(2017, 8, 23, 0, 0), datetime.datetime(2017, 8, 24, 0, 0), datetime.datetime(2017, 8, 25, 0, 0), datetime.datetime(2017, 8, 26, 0, 0), datetime.datetime(2017, 8, 27, 0, 0),  datetime.datetime(2017, 8, 28, 0, 0), datetime.datetime(2017, 8, 29, 0, 0), datetime.datetime(2017, 8, 30, 0, 0), datetime.datetime(2017, 8, 31, 0, 0)]'*** |

## 

## Problem 2

Create a function called count\_files. The function will accept a string that is in the format of ‘glob pattern’ (review the lectures for example). The function will then use glob to determine how many files match the pattern. The function will return the count of the number of files that match. For example, if 5 files match the pattern, the function will return 5.

Here is an example of how the function should work. Note that the file path I use for the glob\_pattern variable will not work on your computer, this is just an example.

### Example 1

|  |
| --- |
| glob\_pattern = os.path.join('path', 'to', 'data', 'logData\_\*.txt')  num\_files = count\_files(glob\_pattern) |

## 

## Problem 3

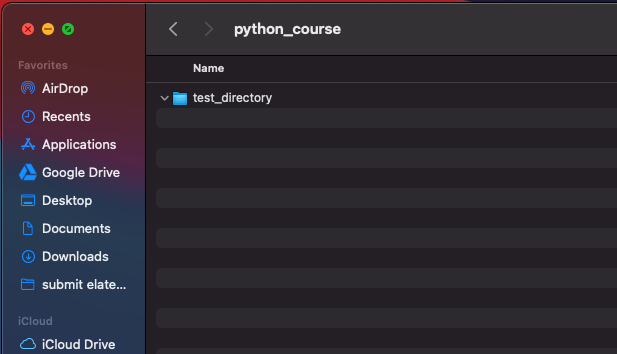
Create a function called make\_dirs. This function will accept two parameters. The first will be a path to a directory that exists (the function will check if it exists). The second will be a list of directory names that you want created inside the directory that exists. The function should check if the directory given in the first argument exists and is a directory. If the directory does not exist, the function will return False. If the directory does exist, the function will create new directories inside of the directory. The second argument is a list that contains the names of the new directories. The function will also return True if the directory, given in the first argument, exists.

Here is an example of how the function should work

### Example 1

|  |
| --- |
| directory\_path = os.path.join('Users', 'williamhenry', 'Documents', 'python\_course', 'test\_directory')  new\_directory\_names = ['dir\_a', 'dir\_b', 'dir\_c']  success = make\_dirs(directory\_path, new\_directory\_names)  print(success) # should print True if the directory\_path exists and is a directory, otherwise it should print False |

### Screenshots of ‘test\_directory’ before and after running the function above:





## Problem 4

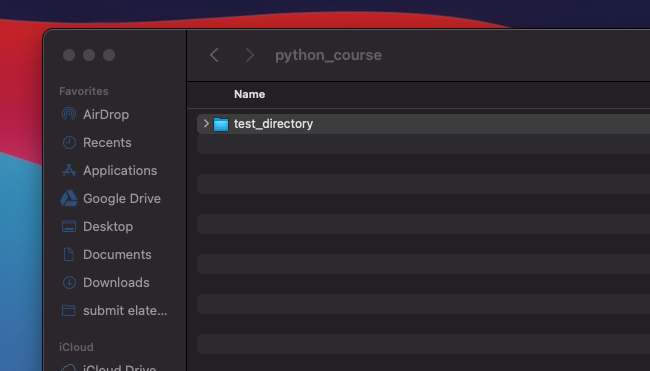
Create a function called create\_dir\_with\_timestamp. The function will accept one argument. The argument is the path to a new directory that you want to create. The function will create the directory; however, it will add the current datetime to the directory name. The datetime format should be like so ‘%Y%m%dT%H%M%S.

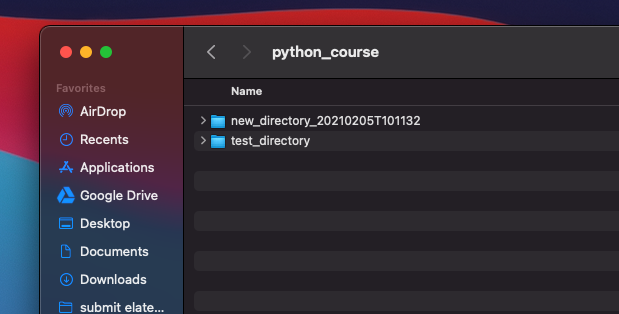
Here is an example of how the function should work:

### Example 1

|  |
| --- |
| directory\_path = os.path.join('Users', 'williamhenry', 'Documents', 'python\_course', 'new\_directory')  create\_dir\_with\_timestamp(directory\_path) |

#### Screenshots of the ‘python\_course’ directory before and after running the function above:





## Problem 5

Create a function called print\_environment. This is a simpler function than the problems above; however, the subject may be less familiar. This function will print all of the environment variables. You can use the os.environ attribute to access the current environment variables. os.environ is a dictionary of the current environment variables, this is the dictionary you want to print. The function will simply loop through the keys in the dictionary and print each key, value pair.

Here is an example of how the function should work:

### Example 1

|  |
| --- |
| print\_environment()  Something like the following should print out:  environ{'\_\_CFBundleIdentifier': 'com.apple.Terminal',  'TMPDIR': '/var/folders/0w/j\_2c9mf91zl2p9qzndt81ncw0000gn/T/',  'XPC\_FLAGS': '0x0',  'LaunchInstanceID': '3D4CF8BC-78DA-413C-9E41-0DE23C3FCAFA',  'TERM': 'xterm-256color',  'SSH\_AUTH\_SOCK': '/private/tmp/com.apple.launchd.nBi7HtUzV4/Listeners',  'SECURITYSESSIONID': '186aa',  'XPC\_SERVICE\_NAME': '0',  'TERM\_PROGRAM': 'Apple\_Terminal',  'TERM\_PROGRAM\_VERSION': '440',  'TERM\_SESSION\_ID': 'A844D004-78B0-46BB-8772-B581D1A51DBA',  'SHELL': '/bin/zsh',  'HOME': '/Users/williamhenry',  'LOGNAME': 'williamhenry',  'USER': 'williamhenry',  'PATH': '/usr/local/opt/mysql-client/bin:/Users/williamhenry/.local/bin:/Applications/google-cloud-sdk/bin:/Users/williamhenry/opt/anaconda3/bin:/Users/williamhenry/opt/anaconda3/condabin:/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin:/Library/TeX/texbin:/Library/Apple/usr/bin',  'SHLVL': '1',  'PWD': '/Users/williamhenry',  'OLDPWD': '/Users/williamhenry/Documents/2\_Areas\_Of\_Responsibility/UCI\_Courses/UCI\_Introduction\_To\_Python\_For\_Data\_Analysis/nbgrader',  'CONDA\_EXE': '/Users/williamhenry/opt/anaconda3/bin/conda',  '\_CE\_M': '',  '\_CE\_CONDA': '',  'CONDA\_PYTHON\_EXE': '/Users/williamhenry/opt/anaconda3/bin/python',  'CONDA\_SHLVL': '1',  'CONDA\_PREFIX': '/Users/williamhenry/opt/anaconda3',  'CONDA\_DEFAULT\_ENV': 'base',  'CONDA\_PROMPT\_MODIFIER': '(base) ',  'LANG': 'en\_US.UTF-8',  '\_': '/Users/williamhenry/opt/anaconda3/bin/ipython'} |

## How To Turn In The Assignment

First, remove any testing/ scripting code so that only the functions definitions remain in your notebook. If you want to make a copy of the notebook to keep a copy of any scripting or testing code that is fine. But, otherwise - ONLY turn in the functions definitions.

Then, please download the notebooks as a Python file (Go to the file menu, in Jupyter notebooks, and choose "Download as…", then choose python to download as a python file) and submit the assignment.