Midterm Exam for Programming Tools for the Data Scientist: April 2021

Name: ____
Net ID ___________________

Instructions

There are two sections of this test. The first section consists of 8 multiple choice questions, worth 8 points each, for a total of 64 points points. The second section consists of two longer questions worth 18 points each, for a total of 36 points. Therefore, a perfect score on the test would be 100 points.
At the end of this file there are two blank pages, should you need additional space to add answers (see below). There is also a glossary of defining the Python and terminal commands that we used in this course. As per the instructions, you are permitted to consult this glossary during the exam.
This test booklet is a .pdf file. You should prepare a .pdf file with your answers to submit to Gradescope in the permitted time period. This .pdf can take one of the following forms:

- If you are able to edit .pdf files, you can revise this test booklet and submit that to GradeScope.
- You can print out the test, fill in your answers in pen or pencil (please make the writing dark enough so it is legible). Then you can scan this file and create a .pdf file. You can submit this. Very clear pictures taken with a cell phone camera are acceptable. However, poor quality photographs that the grader cannot read are not acceptable.
- You can create a .pdf file with a Word processor that contains all the answers in sequence, numbered as appropriate. For example, Microsoft Word has a “save as pdf or export to pdf or something like that” option.

This test has the following time restrictions:

- This test must be completed less than 24 hours after it is released.
- After downloading the test, you have 1 hour and 15 minutes to complete the test. In addition, there is a 15 minute window in case you have technical difficulties submitting the test.
- The above 1 hour and 15 minute window will be modified as a appropriate, if you submitted an accommodation letter through NYU’s Moses Center. For a 1.5 times accommodation, the window should be 1 hour and 55 minutes. For a 2 times accommodation, the window should be 2 hours and 30 minutes. The 15 minute window for technical difficulties will be the same.

This test is **partially** open book. The following are permitted:

- You can look at slides from class during the test.
- You can look at the glossary provided as part of the test. This will be the same glossary as is provided with the practice test.
- You can look at UNIX “man” pages, e.g, type “man ls” in Linux to find out about the “ls” command.
- You can use a simple calculator.
- For Part 2 questions, you can test your code in IDLE or similar IDE for Python. For shell scripts, you can test in a linux shell.

Since the test is partially open book, it is in part **closed** book as well. The following are not permitted:

- Do not communicate with others.

- Do not do websearches.
- Do not look at any materials not specifically listed above as being OK. For example, do not look at your notes.

Please (digitally or manually) sign in any way that you can the following statement:
I agree to complete this test, independently, in the appropriate amount of time as per the instructions:

____

Answer all questions on the test. There will be opportunities for partial credit on questions.

Section 1

Answer all 8 multiple choice questions. As per each question, indicate which answers are correct. You can circle the right answers with a pen, or put a dark X or asterisk next to the correct answers. For individual questions, it may be appropriate to either choose a single answer or to choose multiple answers. For example, in sample question **i**, there is a single correct answer, but for sample question **ii**, the correct answer entails choosing more than one of the possibilities enumerated as **a** through **e**. There also may be a different number of options for each question. The correct answers are indicated with blue Xs.

Sample Question I. Which of the following is the name of a month in the standard Gregorian Calendar?

(a) March **X**

(b) Earth Day

(c) Wednesday

(d) Noon

(e) Springtime

Sample Question II. Which of the following are names months in the standard Gregorian Calendar? Choose all that apply.

(a) March **X**

(b) December **X**

(c) Wednesday

(d) January **X**

(e) Springtime

Part I Questions 1-8

1. Given a file called ”blah.txt”, what does the following shell command do: **chmod 664 blah.txt**? Choose the single best or most complete answer.
   (a) Allow the the user to read and modify the file.
   (b) Allow the members of the user’s group to read and modify the file.
   (c) Allow anyone to read the file with access to the file system.
   (d) Make the file executable.
   (e) a,b and d
   (f) a, b and c **X**

2. Chose the single answer that most completely describes what the command ”git pull” does:
   (a) It downloads the current version of a repository
   (b) It checks to see if the current version of a repository is compatible with your version
   (c) It merges the current version of the program with your version, if they are compatible.
   (d) It indicates any parts of your program that are incompatible with the current version
   (e) a and b
   (f) c and d **X**

3. What are the two most common values for os.linesep
   (a) ’\n’ and ’\r’
   (b) ’\n’ and ’\r \n’ **X**
   (c) ’\n’ and ’\r \r’
   (d) ’\r’ and ’\r \n’

4. Given two files:

   ”three_numbers.csv”– A file consisting of the following lines:

   ```
   41,522,6
   100,200,300
   75,81,91
   ```

   and an (executable) shell script called ”**pick_2.sh**” which consists of the following single line:

   ```
   sort $1 | cut -f1 -d, > $1.result1
   ```

   If you executed the command ”pick_2.sh three_number.csv”, which of he following would occur (indicate all that apply):
   (a) A file called three_numbers.result1 would be produced
   (b) A file called three_numbers.csv.result1 would be produced **X**
   (c) The output file would contain three numbers per line
   (d) The first line of the output file would contain the number ”41”
   (e) The last line of the output file would contain the number ”75” **X**

5. You are trying to predict whether a person is likely to have brown eyes or blue eyes from other traits. To achieve this goal you look for patterns in a medical database. The database lists 1000s of different traits of 1000 people, 750 who have brown eyes and 250 who have blue eyes. After searching through the database, you observe that there are 100 people whose names begin with ”B”, 65 of whom have brown eyes and 35 whom have blue eyes. There are 20 people whose names begin with ”Z”, 16 who have brown eyes and 4 of them have blue eyes. Consider these two hypotheses:

   Hypothesis I: People whose name begins with ”B” have a higher than normal probability of having blue eyes

   Hypothesis II: People whose name begins with ”Z” have a higher than normal probability of having brown eyes.

   What is wrong with accepting both of these hypotheses without further research (choose the best single answer)?

   (a) One cannot test hypotheses on the same data used to motivate those hypotheses. Otherwise, it is like our dice rolling homework. **X**

   (b) The indicators are not strong enough. For these hypotheses to be valid, it would be better if 40/100 of the people whose names begin with ”B” had blue eyes, and it would be better if 17/20 of the people whose name begin with ”Z” have brown eyes.

   (c) You are not an expert in genetics and so you are not qualified to make these hypotheses.

6. Which statements are true about a symbolic link to a file, but not about a copy you made of the same file in your own directory. Mark one or more of these choices.

   (a) If there is a change made to the original file, the symbolic link will reflect that change **X**

   (b) If there is a change made to the original file, the copy will reflect that change

   (c) If you do not own the original file or do not have “write” permission, you can still change the contents of the copy of the file **X**

   (d) If you do not own the original file or do not have “write” permission, you can still change the contents of the symbolic link

7. Two Part Question.

   I. Let’s say there was a new English verb that you never heard of before: ”bliffy”. If you use a correct version of the homework 4 verbal inflection program to generate this past tense form, what form should it produce:

      (a) bliffyed

      (b) bliffied **X**

   II. If it turned out that ”bliffy” is irregular and that the real past tense form should be ”did-bliffy”, what would be the best way of incorporating it into your homework program?

      (a) Add a rule that adds the prefix ”did-” at the beginning of words ending in ”ffy”.

      (b) Add an entry to verb_irreg.tsv file for ”bliffy” with “did-bliffy” in the PAST column**X**

      (c) Add a rule to your Python program that specifically applied to the past-tense of ”bliffy”.

8. What type of scale are five star ratings on movies and other products? In these rating systems customers assign a rating of 1 to products they don't like, 5 to products they like a lot and intermediate scores to products they like in varying degrees:

   (a) Nominal
   (b) Ordinal **X**
   (c) Interval
   (d) Ratio

Section 2

Answer both of the following two questions. Depending on the question, the answer can be a shell script, a Python program, a regular expression or some other programmatic solution to some problem. You can test the program on a linux terminal or in an IDE like Idle. However, it does not need to be a fully debugged program to get a good score.

Question 9: Write Python code that takes a .csv file as input and produces a .csv file as output. Assume that the input .csv file is in the following format. The first line has headings separated by commas and all subsequent lines are like the second and third line. You can assume that the fields in this .csv file contain no commas or newlines. An example input file follows.

Sample Input File

```
Student,Shoesize,Phone Number
Mary,6,212-333-4444
John,12,384-111-9797
```

The output file should be a two column .csv file as in the example below. The first line has headings separated by commas and all subsequent lines are like the second and third line. To calculate the value of the "code" column, do the following: (a) split the phone number by hyphens (-); (b) convert each of these pieces of the phone number to floats; (c) convert the shoe size to a float; and (d) multiply the resulting 4 numbers together. For example, $6 \times 212 \times 333 \times 444 = 1882371744.0$.

Sample Ouput File

```
Student,Code
Mary,1882371744.0
John,5011047936.0
```

Sample Answer is on the next page

Sample Answer

```
def get_code(shoesize,phone_number):
    score = float(shoesize)
    phonelist = phone_number.split('-')
    for num in phonelist:
        score = score *float(num)
    output = str(score)
    return(output)

def shoephone(infile,outfile):
    import os
    with open(infile) as \
      instream,open(outfile,'w') as outstream:
        title_found = False
        for line in instream:
            line = line.strip(os.linesep)
            student,shoesize,phone_number = line.split(',')
            if title_found ==False:
                outstream.write('Student,Code\n')
                title_found = True
            else:
                outstream.write(student+','\
                     +get_code(shoesize,phone_number)+'\n')
```

10. Write a regular expression that matches a string of alternating letters and digits (starting with either a digit or a letter). For example, the expression should match the entire strings in the first column, but should not match the entire strings in the second column because there are some letters followed by other letters; some of the digits are followed by other digits; some the strings contain punctuation and/or some strings do not contain any digit; and/or some strings do not contain any letters. Note that it is OK if your regular expression matches substrings of the second group of strings. For example, when searching 'aa34h8', your regexp pattern can match "a3" or "4h8", it just shouldn't match the whole 'aa34h8'.

Matching string examples	**Non-matching string examples**
`a1b2c3`	`aa34h8`
`A5b9Z7q4`	`a5n8;9`
`C7u8D0`	`AAAAbbb`
`e1e1e0`	`123456`
`5p6w`	

Sample Answer

```
(([a-zA-Z][0-9])+)|(([0-9][a-zA-Z])+)
```

Glossary of Terms

1. Python's **os** module includes global variables like *os.linesep* (end of line strings: '$\backslash n$' or '$\backslash r \backslash n$') and *os.sep* (path separators – forward slash '/' or backward slash '\'). The os module also includes functions that interact with the operating system. *os.getcwd()* returns the current working directory. *os.listdir(PATH)* returns a list of files in PATH; *os.path.isdir(PATH)* returns True if PATH is a directory and False otherwise; *os.path.isfile(PATH)* returns True if PATH is the name of an existing file and False otherwise.

2. File Object Streams – Python objects used for reading files and writing to files.

   - *instream = open('my_file.txt','r')* sets the variable *instream* to the contents of the file *'my_file.txt'*. *for* loops will treat *instream* as a list of strings, each ending with *os.linesep*. For most applications, it makes sense to remove these.
   - *outstream = open('my_file.txt','w')* sets the variable outstream to an object that will ultimately be saved as the file *my_file.txt*. The method *outstream.write(string)* will write a string to that file. It is a good idea to include $\backslash n$ anywhere you would like a line break in the file as end of lines are not automatic. $\backslash n$ should be used, rather than os.linesep, even in Windows.
   - *stream.close()* will close an opened stream. This ends the connection between Python and a file. In the case of output streams (like *outstream*), the content of the stream is written to the file.
   - *with open(file,'r') as instream:* or *with open(file,'w') as outstream:* starts a block in which a stream is opened. The body of code indented under these statements can read from or write to the stream. After the block ends, the stream is closed.

3. **requests.get(url)** is part of the **requests** library. It produces a input stream containing the content of the url (web address). **requests.get(url).text** (the .text value of that stream object) is the html text from that website. It is possible to obtain text between <p> and </p> or paragraphs and remove all html from that text using other filters. For the class exercises, I wrote such filters using regular expressions and Pythons **re** package.

4. **urllib.request.urlopen** is a command in the **urllib.request** library. It is for obtaining text from websites that we used in conjunction with **BeautifulSoup**.

5. **bs4.BeautifulSoup** (or BeautifulSoup.BeautifulSoup) is a html (and xml) parser that takes two arguments: the input stream produced by **urllib.request.urlopen** ; and 'lxml' is the name of a library that BeautifulSoup uses to process the html. The soup variable in: **soup = bs4.BeautifulSoup(input_stream)** is an object that contains the website in a sorted form, so it is possible to lookup html fields. We used it to find all the paragraphs, by means of the command **paragraphs = soup.find_all('p')**. This

produced a list of paragraph structures and assigns them to the variable **paragraphs**. The text from each paragraph is accessible using **.text**, e.g., **paragrpahs[0].text** refers to the text of the first paragraph.

6. **Python re package** includes a variety of ways to use regular expressions. The most basic use is with the command **re.search(REGEXP,TEXT_TO_SEARCH)**. This produces a **re.Match object**. For example, the command **abc = re.search('(ABC)+','123ABCA**
returns a regexp object representing the substring **ABCABCABC**. The object has several values including abc.group(0) (the whole match) and abc.group(1) (the first instance of ABC). The group number N after 0 refers to a piece of the matching regep, matching the part of the regexp starting at the Nth parenthesis. So the 1st parenthesis is around '(ABC) and thus, it corresponds to the first match of the string "ABC". In addition to **MATCH.group(number)**, there is also **MATCH.start(N)** and **MATCH.end(N)**, referring to the beginning and end of the block of text matching group N. In addition to re.search, other re functions include **re.findall**, **re.finditer** and **re.sub**. **re.findall** and **re.finditer** identify multiple instances of a regexp inside a string. **re.sub** substitutes a regexp with a replacement string.

7. Python **csv** package is a package for processing comma separated value files (and tab separated value files). **csv.reader(instream)** returns a list of lists from a **csv** file. Unlike **string.split(',')**, it accounts for more complex **csv** structure that is used for spreadsheets, e.g., where columns are surrounded by quotes and a field can contain a comma. There are other commands such as **csv.writer.writerow** that we did not cover in class.

8. Python **sys.exit** and **sys.argv**. **sys.exit** exits Python. **sys.argv** is a list of arguments (strings) taken from command line use of Python. sys.argv[0] is the name of the Python File you are executing and the remaining items in the list are the command line parameters. For example, if you executed the command "python3 do_stuff.py 57 100, sys.argv would be equal to ['do_stuff.py', '57', '100'].

9. Linux shell commands:

    - **ls** lists the files in a directory. There are numerous flags (e.g., -l) that you can use to provide additional detail or present the results in different ways.
    - **cp**, **mkdir**, **ln -s** are for copying files, making directories and making symbolic links.
    - **chmod** changes the file permissions on a file or directory.
    - **ˆC, kill, fg, bg, ˆZ, &, top, ps, free** are commands that have to do with manipulating jobs (computer processes) or finding out information about jobs.
    - command for printing things to the screen or printing parts of files: echo, cat, grep, cut, sort, uniq

- > and | are operators used in shell commands. | directs the output of the process on its left to be input for the process on its right. > sends the output of the process on its left to the filename on its write, creating a file in the process.
- Shell scripts are executable files that contain shell commands. Very simple shell scripts can be created which are just sequences of shell commands. In addition, command line parameters can be referred to. $1 is the first parameter on the command line; $2 is the second argument and so on.

10. Git commands: git push, git pull, git add, git commit, git clone

11. Short guide to regular expresions

    (a) Repeat operators: *, +, {number}, {minimum,maxium}

    (b) Optional and disjunction: ? and |

    (c) Parentheses () mark scope of repeat, optional and disjunction operators

    (d) Beginning and End of line: ˆand $

    (e) Disjunction (and ranges) for single characters: [a-z], [A-Z], [0-9], [a-zA-Z,0-9], [123abc]

    (f) Negation for single characters [ˆa-z]

    (g) The period . represents any character, e.g., ".*" represents any string. Thus the expression ' ˆA.*Z$' matches any line that starts with A and ends with Z.