CPSC 1301, Computer Science I Quiz

Quiz 9b

This is a timed test. You have thirty minutes to complete the test. Your deliverable will be a plain text file, that is, an ASCII file with a `.txt` file extension. When you finish the test, upload your deliverable to Canvas. Do not publish your answer to your `git` repository.

You may work with your study partner for this quiz. In fact, working together is strongly encouraged. If you work with a partner, you must each make a seperate submission for credit, but you must also include the names of both authors in your submission.

1 Instructions

You will find in the template below two data structures, `presidentsJson` and `month`. I have reproduced them in the output below.[1] The requirement is to take the JSON file and print it to an output file named "presidents.tsv." Note that the items in this file are separated by tabs (hence *tsv*), which is a very common data format. Also note that the file contains a header and that the strings are surrounded by double quotes ("). Here is the header:

```
"order"\t"party"\t"name"\t"date of birth"\t"date inaugurated"\t"date of death"
```

You will write a short script that will write the JSON file given to you to an output file named `presidents.tsv`. *This is not a difficult task!* Here are the steps:

1. process the input data, handling each list item off to a processing function that processes lines
2. for each line, break the line into all the parts, understanding how to use nested lists and dictionaries
3. reassemble the parts into a string that meets the requirements
4. write the string to the output file

Finally, you can use multiple methods for opening and reading files. I would like for you to use the File Context Manager, as shown on page 294 of your book. I have copied the code below. I have a very specific reason for this requirement, which I will explain to you if you ask. (Don't worry, it's a real reason.)

```
1 with open('data.csv') as myfile: # See Chapter 34 for details
2     for line in myfile:
3         ...use line here...
```

For those of you who have been reading the book, you will notice that we have been covering CSV and JSON formats. The XML format is coming up. These topics are advanced for this class, but at least you will gain experience with these formats even though you many not have been formally introduced to them.

2 Template

You should not be needing a template at this point. Here is a minimal template to get you started. I have included only the months, the presidents JSON data, an empty list, and the write-to-file statement. The remainder is up to you.

[1] The imput JSON data is identical to the output in Quiz 09a. You can use that output to save typing.

```
# quiz09b.py

# initialize month dictionary
month = {
        1:'January',
        2:'February',
        3:'March',
        4:'April',
        5:'May',
        6:'June',
        7:'July',
        8:'August',
        9:'September',
        10:'October',
        11:'November',
        12:'December'
        }

#initialize presidents list
presidentsJson = [
        {'order': 1, 'name': {'firstName': 'george', 'middleName': '', 'lastName': '
            washington'}, 'dates': {'birth': [1722, 2, 22], 'inaugurated': [1789, 4, 30], '
            death': [1799, 12, 13]}, 'facts': {'state': 'virginia', 'party': '', 'religion':
             'deist'}},
        {'order': 2, 'name': {'firstName': 'john', 'middleName': '', 'lastName': 'adams'}, '
            dates': {'birth': [1735, 10, 30], 'inaugurated': [1797, 3, 4], 'death': [1826,
            7, 4]}, 'facts': {'state': 'massachusetts', 'party': 'federalist', 'religion': '
            unitarian'}},
        {'order': 3, 'name': {'firstName': 'thomas', 'middleName': '', 'lastName': '
            jefferson'}, 'dates': {'birth': [1743, 4, 13], 'inaugurated': [1801, 3, 4], '
            death': [1826, 7, 4]}, 'facts': {'state': 'virginia', 'party': 'democratic-
            republican', 'religion': 'deist'}},
        {'order': 4, 'name': {'firstName': 'james', 'middleName': '', 'lastName': 'madison'
            }, 'dates': {'birth': [1751, 3, 16], 'inaugurated': [1809, 3, 4], 'death':
            [1826, 6, 28]}, 'facts': {'state': 'virginia', 'party': 'democratic-republican',
             'religion': 'episcopalian'}},
        {'order': 5, 'name': {'firstName': 'james', 'middleName': '', 'lastName': 'monroe'},
             'dates': {'birth': [1758, 4, 28], 'inaugurated': [1817, 3, 4], 'death': [1831,
            7, 4]}, 'facts': {'state': 'virginia', 'party': 'democratic-republican', '
            religion': 'episcopalian'}},
        {'order': 6, 'name': {'firstName': 'john', 'middleName': 'quincy', 'lastName': '
            adams'}, 'dates': {'birth': [1767, 11, 7], 'inaugurated': [1825, 3, 4], 'death':
             [1848, 2, 23]}, 'facts': {'state': 'massachusetts', 'party': 'democratic-
            republican', 'religion': 'unitarian'}}
]

presidentsTsv = []

with (open("presidents.tsv", "w")) as f:
    for pres in presidentsTsv:
        f.write(pres)
        f.write('\n')
```

3 Output

You should produce an output file named `presidents.tsv` that looks like this. Please note that the empty spaces used as item separators are actually tabs (`\t`), that integers do not have quote delimiters, and that all strings have quote delimiters.

```
"order" "party" "name" "date of birth" "date inaugurated" "date of death"
1 "washington, george " "" "February 22, 1722" "April 30, 1789" "December 13, 1799"
2 "adams, john " "federalist" "October 30, 1735" "March 4, 1797" "July 4, 1826"
3 "jefferson, thomas " "democratic-republican" "April 13, 1743" "March 4, 1801" "July 4, 1826"
4 "madison, james " "democratic-republican" "March 16, 1751" "March 4, 1809" "June 28, 1826"
5 "monroe, james " "democratic-republican" "April 28, 1758" "March 4, 1817" "July 4, 1831"
6 "adams, john quincy" "democratic-republican" "November 7, 1767" "March 4, 1825" "February 23, 1848"
```