Stats 21 - HW 2 - Due 4/22/2023 by 11:59PM

Blair Lee - 005721089

Homework is generally an opportunity to practice coding and to train your problem solving and critical thinking skills. Putting Python to use is where learning happens.

Copying and pasting another's solutions takes away your learning opportunities. It is also academic dishonesty.

ChatGPT is always allowed in this class, but do remember, it is not foolproof and if your solution looks too much like another submission, I am required to file a report

Please use this document as your homework template and submit both the modified .ipynb file and a PDF export.

Assignment

1. Data Types (2pts)

Please evaluate these expressions as if you were going to evaluate them using Python **BUT TRY NOT TO** evaluate instead, look at them first and identify the result by adding a comment inline. Of course you can evaluate them to check and correct, but do try it without Python. I will do the first one for you

In [ ]:
```
34 # int
34 / 2 # float
34 % 3 # int
True # bool
'True' # str
'True' * 4 # str
34 < 12 # bool
4**2 # int
4.**2 # float
'501288874827499166103492629211225388323720569439875448338896266889251097274
```

Out[ ]:
```
'50128887482749916610349262921122538832372056943987544833889626688925109727
46226260A'
```

If you got any of those wrong, please do stop and ask yourself why.

2. Working with Strings (3pts)

Please refer back to the Gettysburg Address (or any other comparably short work - like lyrics to your favorite song or some other work) and try the following questions

(a) Count the words (not individual letters) it's OK to leave the punctuation in (but if you know how to remove it or want to learn, go ahead)

In [ ]:
```python
## answer to 3a
import requests
import string

url = 'http://www.stat.ucla.edu/~vlew/datasets/gettysburg.txt'

response = requests.get(url)

gettysburg = response.text

no_punctuation = []
no_punctuation_gettysburg = []
for c in gettysburg:
    if c not in string.punctuation:
        no_punctuation.append(c)
        no_punctuation_gettysburg = "".join(no_punctuation)

gettysburg_words = no_punctuation_gettysburg.split()

print(len(gettysburg_words))
```

```
267
```

(b) Count the presence of any one specific word. For example "and"

In [ ]:
```python
## answer to 3b
gettysburg_lower = gettysburg.lower()
gettysburg_words = gettysburg_lower.split()
counter = 0
for word in gettysburg_words:
    if word == "and":
        counter += 1
print(counter)
```

```
6
```

(c) Please choose any word and demonstrate that you can replace it.

In [ ]:
```python
## answer to 3c
print(gettysburg, end = "\n\n")

print(gettysburg.replace("seven", "eight"))
```

```
Fourscore and seven years ago our fathers brought forth on this continent a n
ew nation, conceived in liberty and dedicated to the proposition that all men
are created equal. Now we are engaged in a great civil war, testing whether t
hat nation or any nation so conceived and so dedicated can long endure. We ar
e met on a great battlefield of that war. We have come to dedicate a portion
of that field as a final resting place for those who here gave their lives th
at that nation might live. It is altogether fitting and proper that we should
do this. But in a larger sense, we cannot dedicate, we cannot consecrate, we
cannot hallow this ground. The brave men, living and dead who struggled here
have consecrated it far above our poor power to add or detract. The world wil
l little note nor long remember what we say here, but it can never forget wha
t they did here. It is for us the living rather to be dedicated here to the u
nfinished work which they who fought here have thus far so nobly advanced. It
is rather for us to be here dedicated to the great task remaining before us t
hat from these honored dead we take increased devotion to that cause for whic
h they gave the last full measure of devotion that we here highly resolve tha
t these dead shall not have died in vain, that this nation under God shall ha
ve a new birth of freedom, and that government of the people, by the people,
for the people shall not perish from the earth

Fourscore and eight years ago our fathers brought forth on this continent a n
ew nation, conceived in liberty and dedicated to the proposition that all men
are created equal. Now we are engaged in a great civil war, testing whether t
hat nation or any nation so conceived and so dedicated can long endure. We ar
e met on a great battlefield of that war. We have come to dedicate a portion
of that field as a final resting place for those who here gave their lives th
at that nation might live. It is altogether fitting and proper that we should
do this. But in a larger sense, we cannot dedicate, we cannot consecrate, we
cannot hallow this ground. The brave men, living and dead who struggled here
have consecrated it far above our poor power to add or detract. The world wil
l little note nor long remember what we say here, but it can never forget wha
t they did here. It is for us the living rather to be dedicated here to the u
nfinished work which they who fought here have thus far so nobly advanced. It
is rather for us to be here dedicated to the great task remaining before us t
hat from these honored dead we take increased devotion to that cause for whic
h they gave the last full measure of devotion that we here highly resolve tha
t these dead shall not have died in vain, that this nation under God shall ha
ve a new birth of freedom, and that government of the people, by the people,
for the people shall not perish from the earth
```

4. Generating Sequences (2pts)

(a) Please show us how to generate a sequence of the first 10 positive even numbers

In [ ]:
```python
## answer to 4a
sequence = range(2, 21, 2)
for n in sequence:
    print(n, end = " ")
```

```
2 4 6 8 10 12 14 16 18 20
```

(b) Please show us how to generate a sequence of the first 'n' powers of 2 (2^0, 2^1, 2^2, ... 2^n), where 'n' is a positive integer.

```
In [ ]: ## answer to 4b
        n = 10 # can be equal to any positive integer
        for x in range(0, n + 1):
            print(2**x, end = " ")
```

```
1 2 4 8 16 32 64 128 256 512 1024
```

5. Conditionals (3pts)

Given an integer (0-100) representing a student's grade, please write a bit of code to classify it into one of the following categories:

A: 90-100 B: 80-89 C: 51-79 F: 0-50

```
In [ ]: ## please use following values to show us
        ## that your classifier works

        student_grade = [91.2, 83, 57.3, 12]

        letter_grade = []
        for grade in student_grade:
            if grade <= 50:
                letter_grade.append("F")
            if grade > 50 and grade <= 79:
                letter_grade.append("C")
            if grade > 79 and grade <= 89:
                letter_grade.append("B")
            if grade > 89 and grade <= 100:
                letter_grade.append("A")

        print(letter_grade)
```

```
['A', 'B', 'C', 'F']
```