Stats 21 - Homework 09 {-}

Blair Lee - 005721089

Due June 10, 2023 by 11:59pm {-}

Problems have been adapted from the exercises in Think Python 2nd Ed by Allen B. Downey. https://greenteapress.com/thinkpython2/html/thinkpython2017.html

The questions have been entered into this document. You will modify the document by entering your code.

Make sure you run the cell so the requested output is visible.

In [ ]:
```
from IPython.display import display, Math, Latex
```

Problem 1. Convert integers to Roman number {-}

Please write define a class (we got you started wtih some useful attributes) that will convert integers to Roman numbers. So for example the number 2023 should convert to 'MMXXIII'

If a float is received, it should convert the float to integer without error.

You should be able to program this one without importing any libraries/modules.

In [ ]:
```
class IntegerToRoman:
    def __init__(self):
        self.values = [1000, 900, 500, 400, 100, 90, 50, 40, 10, 9, 5, 4, 1]
        self.symbols = ["M", "CM", "D", "CD", "C", "XC", "L", "XL", "X", "IX

    def int_to_roman(self, number):
        if type(number) not in [int, float]:
            print("WARNING: the inputted value is not a number! Returning em
            return ""
        number = round(number)
        i = 0
        roman = ""
        while number:
            division = number // self.values[i]
            number %= self.values[i]
            while division:
                roman += self.symbols[i]
                division -= 1
            i += 1
        return roman
```

In [ ]:
```
converter = IntegerToRoman()
```

In [ ]:
```
# Uncomment the next line and run
converter.int_to_roman(2023)
```

Out[ ]:
```
'MMXXIII'
```

In [ ]:
```
# Uncomment the next lines and run
x = [4, 9, 19.7, 53, 102, 507, 2022, True, False]
for i in x:
    print(IntegerToRoman().int_to_roman(i))
```

```
IV
IX
XX
LIII
CII
DVII
MMXXII
WARNING: the inputted value is not a number! Returning empty string.

WARNING: the inputted value is not a number! Returning empty string.
```

Problem 2. String Cleaning {-}

One of the reasons we use Object Oriented Programming is to reuse code in the current and possibly different applications.

Strings, with only a few exceptions, are unclean.

Please construct a class named StringCleaner that will accept a test string and perform the following actions found below.

You may import any libraries/modules that you have seen presented in this class and any of the modules from the Python Standard library that you may not have seen.

In [ ]:
```
import string

class StringCleaner:
    def __init__(self, string):
        self.string = string

    def clean_string_tolower(self):
        return self.string.lower()

    def clean_string_toupper(self):
        return self.string.upper()

    def remove_punctuation(self):
        no_punc = ""
        punc = '''!()-[]{};:'"\,<>./?@#$%^&*_~'''
        for i in self.string:
```

```
            if i not in punc:
                no_punc += i
        return no_punc

    def remove_capitalization(self):
        return self.string.lower()
```

In [ ]:
```
test_string = "I'm findin' ways to articulate the feelin' I'm goin' through
```

In [ ]:
```
# uncomment the next line and run
test = StringCleaner(test_string)
```

In [ ]:
```
# uncomment the next line and run
test.clean_string_tolower()
```

Out[ ]:
```
"i'm findin' ways to articulate the feelin' i'm goin' through i just can't
say i don't love you (yeah) 'cause i love you, yeah it's hard for me to com
municate the thoughts that i hold but tonight, i'm gon' let you know let me
tell the truth baby, let me tell the truth, yeah"
```

In [ ]:
```
# uncomment the next line and run
test.clean_string_toupper()
```

Out[ ]:
```
"I'M FINDIN' WAYS TO ARTICULATE THE FEELIN' I'M GOIN' THROUGH I JUST CAN'T
SAY I DON'T LOVE YOU (YEAH) 'CAUSE I LOVE YOU, YEAH IT'S HARD FOR ME TO COM
MUNICATE THE THOUGHTS THAT I HOLD BUT TONIGHT, I'M GON' LET YOU KNOW LET ME
TELL THE TRUTH BABY, LET ME TELL THE TRUTH, YEAH"
```

In [ ]:
```
# uncomment the next line and run
test.remove_punctuation()
```

Out[ ]:
```
'Im findin ways to articulate the feelin Im goin through I just cant say I
dont love you Yeah Cause I love you yeah Its hard for me to communicate the
thoughts that I hold But tonight Im gon let you know Let me tell the truth
Baby let me tell the truth yeah'
```

In [ ]:
```
# uncomment the next line and run
test.remove_capitalization()
```

Out[ ]:
```
"i'm findin' ways to articulate the feelin' i'm goin' through i just can't
say i don't love you (yeah) 'cause i love you, yeah it's hard for me to com
municate the thoughts that i hold but tonight, i'm gon' let you know let me
tell the truth baby, let me tell the truth, yeah"
```

In [ ]:
```
# uncomment the next line and run
test_string.lower()
```

Out[ ]:
```
"i'm findin' ways to articulate the feelin' i'm goin' through i just can't
say i don't love you (yeah) 'cause i love you, yeah it's hard for me to com
municate the thoughts that i hold but tonight, i'm gon' let you know let me
tell the truth baby, let me tell the truth, yeah"
```

Problem 3. Rolling Dice {-}

Please construct a class (called DiceRoll below, but name yours whatever) with a method named roll that allows the user to specify the number of die rolled, and the number of sides.

In [ ]:
```
import random
import seaborn as sns

random.seed(1)

class DiceRoll:
    def __init__(self, sides, num_dice):
        self.sides = sides
        self.num_dice = num_dice

    def random_roll(self):
        results = []
        rolls = self.num_dice
        while rolls:
            results.append(random.randint(1, self.sides))
            rolls -= 1
        return results
```

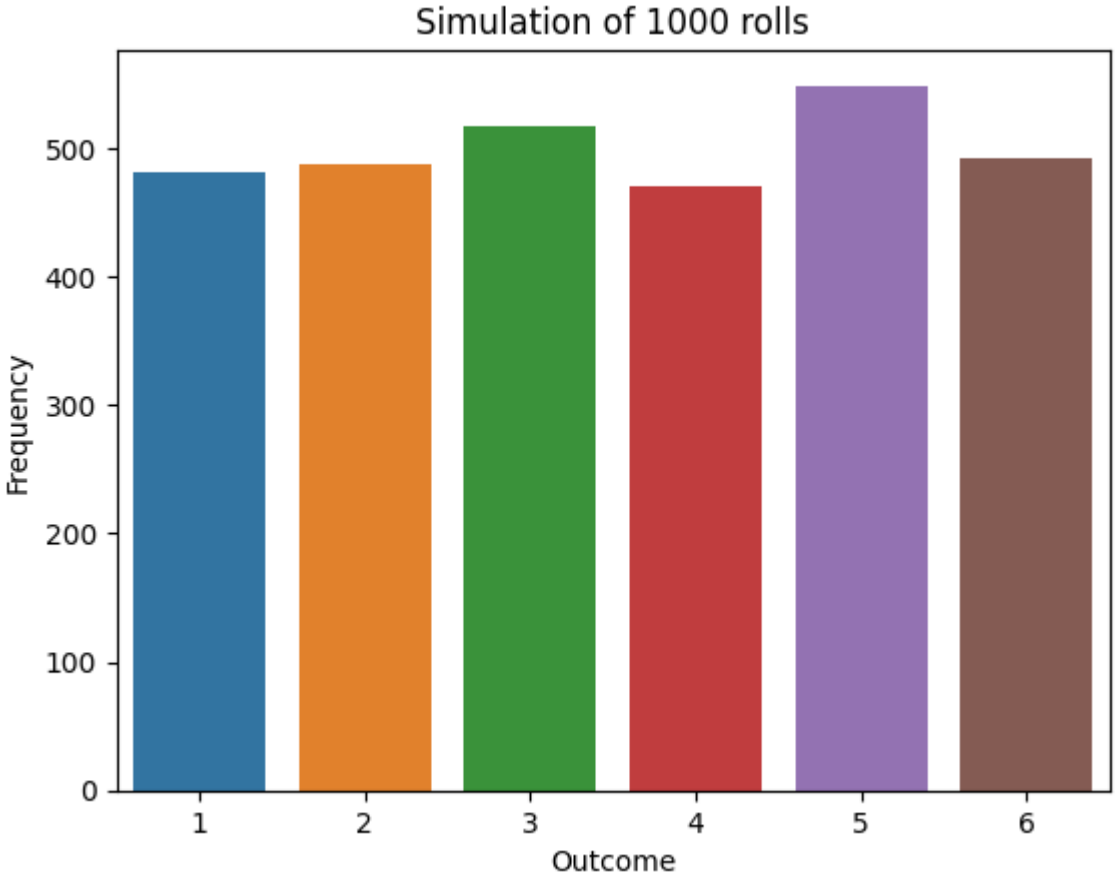
In [ ]:
```
# Roll two 6-sided dice
two_six = DiceRoll(6, 2)
two_six.random_roll()
```

Out[ ]:
```
[2, 5]
```

In [ ]:
```
# Roll three 10-sided dice
three_ten = DiceRoll(10, 3)
three_ten.random_roll()
```

Out[ ]:
```
[2, 5, 2]
```

In [ ]:
```
# Roll three 6 sided dice 1000 times and plot the results
all_rolls = []
many = 1000
while many:
    three_six = DiceRoll(6, 3)
    temp = three_six.random_roll()
    for i in temp:
        all_rolls.append(i)
    many -= 1
sns.countplot(x = all_rolls)
plt.xlabel("Outcome")
plt.ylabel("Frequency")
plt.title("Simulation of 1000 rolls")
plt.show()
```


Simulation of 1000 rolls
Frequency
0
100
200
300
400
500
1
2
3
4
5
6
Outcome