TIPS

Remember to split the problem in pieces so you know how to deal with the challengue

Let's start with some warm up problems

PRINT EVEN NUMBERS, ODD TEXT

First create a list and print all even numbers from 0 to 10 Then use a for loop to print the even numbers and return odd for the odd ones

In [4]:
```
list(range(0,11,2))
```

Out[4]:
```
[0, 2, 4, 6, 8, 10]
```

In [5]:
```
for i in list(range(1,11)):
    if i % 2 == 0:
        print(i)
    else:
        print('odd')
```

```
odd
2
odd
4
odd
6
odd
8
odd
10
```

FIZZ, BUZZ OR FIZZBUZZ

Write a program that prints the integers from 1 to 100. But for multiples of three print "Fizz" instead of the number, and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".

In [6]:
```
for i in range(1,21):
    if i % 3 == 0 and i % 5 == 0:
        print(f'{i} ----> FizzBuzz')
    elif i % 3 == 0:
        print(f'{i} ----> Fizz')
    elif i % 5 == 0:
        print(f'{i} ----> Buzz')
    else:
        print(i)
```

```
1
2
3 ----> Fizz
4
5 ----> Buzz
6 ----> Fizz
7
8
9 ----> Fizz
10 ----> Buzz
11
12 ----> Fizz
13
14
15 ----> FizzBuzz
16
17
18 ----> Fizz
19
20 ----> Buzz
```

PRINT FROM 1 TO N

Print a list of integers from 1 to n without spaces

'n' will be given by the user

Explanation

int(input('Enter a number')) -> here we use input() to be able to get the user number and that input has to be inside the int() because first we need to get the input and then convert it to integer.

Since the default parameter for a print statement is '\n' (which will print another result in another line), with ***end* = ''** we are able to print the desired result in just one line and without spaces

In [7]:
```
n = int(input('Enter a number'))
for i in range(1, n+1):
    print(i, end = '')
```

```
Enter a number10
12345678910
```

COMPREHENSION LISTS

Use List comprehension to create a list of all numbers between 1 and 50 that are divisible by 3.

Use a List Comprehension to create a list of the first letters of every word in the string below:

st = 'Create a list of the first letters of every word in this string'

In [8]:
```
[x for x in range(1,50) if x % 3 == 0]
```

Out[8]:
```
[3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48]
```

In [9]:
```
st = 'Create a list of the first letters of every word in this string'
```

In [10]:
```
[i[0] for i in st.split()]
```

Out[10]:
```
['C', 'a', 'l', 'o', 't', 'f', 'l', 'o', 'e', 'w', 'i', 't', 's']
```

DELETE DUPLICATES

Create a function that deletes duplicates from the list by using a for loop and 'set'. Sort the result

In [11]:
```
myList = [2, 1, 2, 3, 0, 6, 7, 6, 4, 8]
```

In [12]:
```
def deletes(values):
    numbers = []
    for i in myList:
        if i not in numbers:
            numbers.append(i)
    return sorted(numbers)

deletes(myList)
```

Out[12]:
```
[0, 1, 2, 3, 4, 6, 7, 8]
```

In [13]:
```
set(myList)
```

Out[13]:
```
{0, 1, 2, 3, 4, 6, 7, 8}
```

MULTIPLY

Write a Python function to multiply all the numbers in a list.

Sample List : [1, 2, 3, -4]

Expected Output : -24

In [14]:
```
def multiply(list):

    result = 1

    for i in list:
        result = result * i

    return result
```

In [15]:
```
multiply([1,2,3,-4])
```

Out[15]:
```
-24
```

COUNT THE WORDS

Create a function that counts the number of times a word takes place in a phrase.

Explanation:

First you need to prepare the text to lowercase every word so if the phrase contains Cat and cat, when counting, the function will do it properly, otherwise, since it's case sensitive, it wouldn't count it properly.

Then you have to replace the characters for spaces, if not, 'nada,' and 'imaginar.' would count as new words even if we already had 'nada' or 'imaginar'.

After doing all the necessary cleaning, we can go ahead and split the text. If we try to split the text before the previous steps, we'll get an error.

Finally, you just have to loop through the text and if the word isn't in the dictionary, set the count to 1, else, add 1 to the actual count.

In [16]:
```
text = 'Creo que a veces es la gente de la que nadie espera nada, la que ha
```

In [17]:
```
def conteo(text):

    d = {}
    text_clean = text.lower().replace(',','').replace('.','').split()

    for i in text_clean:
        if i not in d:
            d[i] = 1
        else:
            d[i] += 1
    return d

conteo(text)
```

Out[17]:
```
{'creo': 1,
 'que': 4,
 'a': 1,
 'veces': 1,
 'es': 1,
 'la': 3,
 'gente': 1,
 'de': 1,
 'nadie': 2,
 'espera': 1,
 'nada': 1,
 'hace': 1,
 'cosas': 1,
 'puede': 1,
 'imaginar': 1}
```

COUNT THE WORDS THAT START WITH 'S'

Create a function 'phrase' that will print out words that start with 's':

In [18]:
```
st = 'Print only the words that start with s in this sentence'
```

In [19]:
```
def phrase(text):

    for i in st.lower().split():
        if i[0] == 's':
            print(i)
```

In [20]:
```
phrase(st)
```

```
start
s
sentence
```

MINSUM, MAX SUM

Given five positive integers, find the minimum and maximum values that can be calculated by summing exactly four of the five integers.

Then print the respective minimum and maximum values as a single line of two space-separated long integers.

Sample Input 1 2 3 4 5

Sample Output 10 14

In [21]:
```
numbers = [1,2,3,4,5]
```

In [22]:
```
def min_max(numbers):

    minn = min(numbers)
    maxx = max(numbers)

    min_sum = sum(numbers) - maxx
    max_sum = sum(numbers) - minn

    return min_sum, max_sum
```

In [23]:
```
min_max(numbers)
```

Out[23]:
```
(10, 14)
```

EASY PROBLEMS

LESSER - EVEN, ODD

Write a function that returns the lesser of two given numbers if both numbers are even, but returns the greater if one or both numbers are odd.

lesser_of_two_evens(2,4) --> 2

lesser_of_two_evens(2,5) --> 5

In [24]:
```
def check(n1,n2):

    if n1 % 2 == 0 and n2 % 2 == 0:
        print(min(n1,n2))
    else:
        print(max(n1,n2))
```

In [25]:
```
check(2,4)
```

```
2
```

In [26]:
```
check(2,5)
```

```
5
```

ANIMAL CRACKERS

Write a function that takes a two-word string and returns True if both words begin with same letter, false if not.

cracker('Levelheaded Llama') --> True

cracker('Crazy Kangaroo') --> False

In [27]:
```
def cracker(word):
    word_split = word.lower().split()
    return word_split[0][0] == word_split[1][0]
```

In [28]:
```
cracker('Crazy Kangaroo')
```

Out[28]:
```
False
```

In [29]:
```
cracker('Levelheaded Llama')
```

Out[29]:
```
True
```

MAKES TWENTY

Create a function where given two integers, return True if the sum of the integers is 20 or if one of the integers is 20. If not, return False

makes_twenty(20,10) --> True

makes_twenty(12,8) --> True

makes_twenty(2,3) --> False

In [30]:
```
def makes_twenty(num1, num2):
    return num1 + num2 == 20 or num1 == 20 or num2 == 20
```

In [31]:
```
makes_twenty(20,10)
```

Out[31]:
```
True
```

In [32]:
```
makes_twenty(12,8)
```

Out[32]:
```
True
```

In [33]:
```
makes_twenty(2,3)
```

Out[33]:
```
False
```

PALINDROME

Write a Python function that checks whether a word or phrase is palindrome or not.

Note: A palindrome is word, phrase, or sequence that reads the same backward as forward, e.g., madam,kayak,racecar, or a phrase "nurses run". Hint: You may want to check out the .replace() method in a string to help out with dealing with spaces. Also google search how to reverse a string in Python, there are some clever ways to do it with slicing notation.

In [34]:
```
def pal(word):

    word = word.replace(' ','')
    return word == word[::-1]
```

In [35]:
```
pal('madam')
```

Out[35]: True

In [36]:
```
pal('nurses run')
```

Out[36]: True

In [37]:
```
pal('taco')
```

Out[37]: False

LEVEL 1 PROBLEMS

PANGRAM

Write a Python function to check whether a string is pangram or not. (Assume the string passed in does not have any punctuation)

Note : Pangrams are words or sentences containing every letter of the alphabet at least once.

For example : "The quick brown fox jumps over the lazy dog"

Hint: You may want to use .replace() method to get rid of spaces.

Hint: Look at the string module

Hint: In case you want to use set comparisons

In [38]:
```
import string
alphabet=string.ascii_lowercase
alphabet
```

Out[38]: 'abcdefghijklmnopqrstuvwxyz'

In [39]:
```
str1 = "The quick brown fox jumps over the lazy dog"
str1 = str1.lower().replace(' ','')
str1
```

Out[39]: 'thequickbrownfoxjumpsoverthelazydog'

In [40]:
```
import string

def ispangram(str1, alphabet=string.ascii_lowercase):

    alphaset = set(alphabet)
    str1 = str1.lower().replace(' ','')

    return set(str1) == alphaset
```

In [41]:
```
str1 = "The quick brown fox jumps over the lazy dog"
ispangram(str1)
```

Out[41]:
```
True
```

OLD MACDONALD

Write a function that capitalizes the **first** and **fourth** letters of a name

old_macdonald('macdonald') --> MacDonald

In [42]:
```
word = 'macdonald'
```

In [43]:
```
def old_macdonald(word):

    if len(word) > 3:
        return word[:3].capitalize() + word[3:].capitalize()
    else:
        print('The word is to short')
```

In [44]:
```
old_macdonald(word)
```

Out[44]:
```
'MacDonald'
```

MASTER YODA

Given a sentence, return a sentence with the words reversed

master_yoda('I am home') --> 'home am I'

master_yoda('We are ready') --> 'ready are We'

In [45]:
```
sentence = 'I am home'
```

In [46]:
```
def master_yoda(sentence):

    new_sentence = sentence.split()

    return ' '.join(new_sentence[::-1])
```

In [47]:
```
master_yoda(sentence)
```

Out[47]:
```
'home am I'
```

ALMOST THERE

Given an integer n, return True if n is within 10 of either 100 or 200

almost_there(90) --> True

almost_there(104) --> True

almost_there(150) --> False

almost_there(209) --> True

NOTE: abs(num) returns the absolute value of a number

In [48]:
```
def almost_there(number):

    return ((abs(100 - number)) <= 10 or (abs(200 - number)) <= 10)
```

In [49]:
```
almost_there(90)
```

Out[49]:
```
True
```

In [50]:
```
almost_there(104)
```

Out[50]:
```
True
```

In [51]:
```
almost_there(150)
```

Out[51]:
```
False
```

In [52]:
```
almost_there(209)
```

Out[52]:
```
True
```

INSIDE OR NOT

Write a function that checks whether a number is in a given range (inclusive of high and low)

ran_check(5,2,7) -> 5 is in the range between 2 and 7

In [53]:
```
def ran_check(num, low, high):

    if num >= low and num <= high:
        print(f'{num} is in the range between {low} and {high}')
    else:
        print('The number is outside the range')
```

In [54]:
```
ran_check(5,2,7)
```

```
5 is in the range between 2 and 7
```

In [55]:
```
ran_check(15,2,7)
```

```
The number is outside the range
```

UPPER AND LOWER

Write a Python function that accepts a string and calculates the number of upper case letters and lower case letters.

Sample String : 'Hello Mr. Rogers, how are you this fine Tuesday?'

Expected Output :

No. of Upper case characters : 4

No. of Lower case Characters : 33

In [56]:
```
def up_low(string):

    up_count = 0
    low_count = 0

    for i in string:
        if i.isupper():
            up_count += 1
        elif i.islower():
            low_count += 1

    print(f'No of Upper case characters: {up_count}')
    print(f'No of Lower case Characters: {low_count}')
```

In [57]:
```
s = 'Hello Mr. Rogers, how are you this fine Tuesday?'
```

In [58]:
```
up_low(s)
```

```
No of Upper case characters: 4
No of Lower case Characters: 33
```

LEVEL 2 PROBLEMS

FIND 33

Given a list of ints, return True if the array contains a 3 next to a 3 somewhere.

has_33([1, 3, 3]) → True

has_33([1, 3, 1, 3]) → False

has_33([3, 1, 3]) → False

In [59]:
```
mynums = [1,3,3]
```

In [60]:
```
def has_33(numbers):

    for i in range(len(numbers)-1):
        if numbers[i] == 3 and numbers[i+1] == 3:
            return True

    return False
```

In [61]:
```
has_33([1,3,3])
```

Out[61]: True

In [62]:
```
has_33([1,3,1,3])
```

Out[62]: False

STRING X3

Given a string, return a string where for every character in the original there are three characters

paper_doll('Hello') --> 'HHHeeellllllooo'

paper_doll('Mississippi') --> 'MMMiiissssssiiipppppppiii'

In [63]:
```
def paper_doll(string):

    paper_doll = ''
    for i in string:
        paper_doll += i * 3

    return paper_doll
```

In [64]:
```
paper_doll('hello')
```

Out[64]: 'hhheeellllllooo'

BLACKJACK

Given three integers between 1 and 11

If their sum is less than or equal to 21, return their sum.

If their sum exceeds 21 and there's an eleven, reduce the total sum by 10.

If the sum (even after adjustment) exceeds 21, return 'BUST'

blackjack(5,6,7) --> 18

blackjack(9,9,9) --> 'BUST'

blackjack(9,9,11) --> 19

In [65]:
```
def blackjack(n1,n2,n3):

    if sum([n1,n2,n3]) <= 21:
        print(sum([n1,n2,n3]))

    elif sum([n1,n2,n3]) > 21 and 11 in (n1,n2,n3):
        print(sum([n1,n2,n3])-10)

    else:
        print('BUST')
```

In [66]:
```
blackjack(5,6,7)
```

```
18
```

In [67]:
```
blackjack(9,9,9)
```

```
BUST
```

In [68]:
```
blackjack(9,9,11)
```

```
19
```

SUMMER OF '69

Return the sum of the numbers in the array, except ignore sections of numbers starting with a 6 and extending to the next 9 (every 6 will be followed by at least one 9). Return 0 for no numbers.

summer_69([1, 3, 5]) --> 9

summer_69([4, 5, 6, 7, 8, 9]) --> 9

summer_69([2, 1, 6, 9, 11]) --> 14

In [69]:
```
def summer_69(arr):

    total = 0
    add = True

    for i in arr:

        while add:
            if i != 6:
                total += i
                break
            else:
                add = False

        while not add:
            if i != 9:
                break
            else:
                add = True
                break

    return total
```

In [70]:
```
summer_69([1, 3, 5])
```

Out[70]: 9

In [71]:
```
summer_69([4, 5, 6, 7, 8, 9])
```

Out[71]: 9

In [72]:
```
summer_69([2, 1, 6, 9, 11])
```

Out[72]: 14

CHALLENGING PROBLEMS

SECRET SPY

SPY GAME: Write a function that takes in a list of integers and returns True if it contains 007 in order

spy_game([1,2,4,0,0,7,5]) --> True

spy_game([1,0,2,4,0,5,7]) --> True

spy_game([1,7,2,0,4,5,0]) --> False

Explanation

We'll create the a code list that will contain the secret numbers '007' and also a fourth item.

When looping through the given numbers, we'll pop the item if it matches THE CURRENT first item of our code list. So what this does is that we'll be looking exactly for the 007 numbers in order.

Once the items match, they'll will be popped out leaving the code list without that item.

If the code was found, it means we popped out out first 3 items of the code list leaving it with just one item. So we returning the length of code if it is equal to 1, it mean we did find the secret code.

In [73]:
```
def spy_game(numbers):

    code = [0,0,7,'#']

    for i in numbers:
        if i == code[0]:
            code.pop(0)

    return len(code) == 1
```

In [74]:
```
spy_game([1,2,4,0,0,7,5])
```

Out[74]:
```
True
```

In [75]:
```
spy_game([1,0,2,4,0,5,7])
```

Out[75]:
```
True
```

In [76]:
```
spy_game([1,7,2,0,4,5,0])
```

Out[76]:
```
False
```

COUNT PRIMES

Write a function that returns the number of prime numbers that exist up to and including a given number

count_primes(100) --> 25

By convention, 0 and 1 are not prime.

In [77]:
```
def count_primes(num):

    primes = [2]
    x = 3

    if num < 2:  # for the case of num = 0 or 1
        return 0

    while x <= num:

        for y in range(3,x,2):  # test all odd factors up to x-1
            if x%y == 0:
                x += 2
                break
        else:
            primes.append(x)
            x += 2
    print(primes)

    return len(primes)
```

In [78]:
```
count_primes(100)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67,
71, 73, 79, 83, 89, 97]
```

Out[78]:
```
25
```