List sorting

```
# SORTING
my_list = [4, 2, 3, -1, -2, 0, 1]

for i in range(len(my_list)):
    for j in range(i + 1, len(my_list)):

        if my_list[i] > my_list[j]:
            my_list[i], my_list[j] = my_list[j], my_list[i]

print(my_list)
```

```
[-2, -1, 0, 1, 2, 3, 4]
```

Matrix multiplication

```
A = [[12, 7, 3],
    [4, 5, 6],
    [7, 8, 9]]

# take a 3x4 matrix
B = [[5, 8, 1, 2],
    [6, 7, 3, 0],
    [4, 5, 9, 1]]

# result will be 3x4
result = [[sum(a * b for a, b in zip(A_row, B_col))
                        for B_col in zip(*B)]
                                for A_row in A]

for r in result:
    print(r)
```

```
[114, 160, 60, 27]
[74, 97, 73, 14]
[119, 157, 112, 23]
```

List element multiply by 2

```
a1 = [1,2,3,4,5]
a2 = [2]
#print(map(lambda x,y:x*y,a1,a2))

multiplied_list = [element * 2 for element in a1]
print(multiplied_list)
```

```
[2, 4, 6, 8, 10]
```

Filter even no

```
lis1 = [1,2,3,4,5]
```

```
is_even = lambda x: x % 2 == 0

lis2 = list(filter(is_even, lis1))
print(lis2)

# ----------------------------------------

cvb = [i for i in lis1 if i%2 == 0]
cvb
```

```
[2, 4]
```

```
[2, 4]
```

Group and count silimar records

```
from collections import Counter

# initialize list
test_list = [('gfg', ), ('is', ), ('best', ), ('gfg', ),('is', ),
('for', ), ('geeks', )]
print("The original list : " + str(test_list))

# Group and count similar records
# using Counter() + list comprehension + items()
res = [(counter, ) + ele for ele, counter in
Counter(test_list).items()]

# printing result
print("Grouped and counted list is : " + str(res))
```

```
The original list : [('gfg',), ('is',), ('best',), ('gfg',), ('is',),
('for',), ('geeks',)]
Grouped and counted list is : [(2, 'gfg'), (2, 'is'), (1, 'best'), (1,
'for'), (1, 'geeks')]
```

Occurance of word in string

```
test_str = "GeeksforGeeks"
count = 0

for i in test_str:
    if i == 'e':
        count = count + 1
print(count)

# ----------------------------------------

ccbw = test_str.count('e')
print(ccbw)
```

```
4
4
```

Frequency of each words in string

```
from collections import Counter

# initializing string
test_str = "GeeksforGeeks"
res = Counter(test_str)
print(res)

# ------------------------------------

all_freq = {}
for i in test_str:
    if i in all_freq:
        all_freq[i] += 1
    else:
        all_freq[i] = 1
all_freq
```

```
Counter({'e': 4, 'G': 2, 'k': 2, 's': 2, 'f': 1, 'o': 1, 'r': 1})
```

```
{'G': 2, 'e': 4, 'k': 2, 's': 2, 'f': 1, 'o': 1, 'r': 1}
```

convert a list into a string

```
nums=['one','two','three','four','five','six','seven']
s= ' '.join(nums)
s
```

```
'one two three four five six seven'
```

ZIP

```
list(zip(['a','b','c'],[1,2,3]))
```

```
[('a', 1), ('b', 2), ('c', 3)]
```

Remove duplicate from list and nasted list

```
# set(list)

# using loop
test_list = [1, 3, 5, 6, 3, 5, 6, 1]
print ("The original list is : " +  str(test_list))

res = []
for i in test_list:
    if i not in res:
        res.append(i)
print(res)
```

```
######### Nested list
test_list1 = [[1, 0, -1], [-1, 0, 1], [-1, 0, 1],
                           [1, 2, 3], [3, 4, 1]]

print("The original list : " + str(test_list1))
res11 = list(set(tuple(sorted(sub)) for sub in test_list1))
res11
```

```
The original list is : [1, 3, 5, 6, 3, 5, 6, 1]
[1, 3, 5, 6]
The original list : [[1, 0, -1], [-1, 0, 1], [-1, 0, 1], [1, 2, 3],
[3, 4, 1]]

[(1, 2, 3), (-1, 0, 1), (1, 3, 4)]
```

List comprehension

```
am_p = [i for i in range(1,11,2)]
print(am_p)

am_p1 = [i*2 for i in range(1,11,2)]
print(am_p1)
```

```
[1, 3, 5, 7, 9]
[2, 6, 10, 14, 18]
```

Sum of Elements in List

```
list_data = [1,2,3,-3]
n = 0
for i in list_data:
    n = n+i
print(n)
```

```
3
```

removing specified elements index=(0,4,5)

```
color = ['Red', 'Green', 'White', 'Black', 'Pink', 'Yellow']
color = [x for (i,x) in enumerate(color) if i not in (0,4,5)]
print(color)
```

```
['Green', 'White', 'Black']
```

Permutation

```
import itertools
kk_o = [1,2,3]
c = itertools.permutations(kk_o)
print(list(c))
```

```
[(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1)]
```

Change value in tuple

```
x = ("apple", "banana", "cherry")
y = list(x)
y[1] = "kiwi"
x = tuple(y)

print(x)
```

```
('apple', 'kiwi', 'cherry')
```

Difference between two list

```
def Diff(li1, li2):
    return list(set(li1) - set(li2)) + list(set(li2) - set(li1))

# Driver Code
li1 = [10, 15, 20, 25, 30, 35, 40]
li2 = [25, 40, 35]
print(Diff(li1, li2))
```

```
[10, 20, 30, 15]
```

List of multiple integers into a single integer

```
L = [11, 33, 50]
print("Original List: ",L)
x = int("".join(map(str, L)))
print("Single Integer: ",x)
```

```
Original List:  [11, 33, 50]
Single Integer:  113350
```

sum of elements is the highest

```
num = [[1,2,3], [4,5,6], [10,11,12], [7,8,9]]
print(max(num, key=sum))
```

```
[10, 11, 12]
```

Find the items start with specific character from a given list

```
def test(lst, char):
    result = [i for i in lst if i.startswith(char)]
    return result

text = ["abcd", "abc", "bcd", "bkie", "cder", "cdsw", "sdfsd",
"dagfa", "acjd"]
char = "ab"
test(text, char)
```

```
['abcd', 'abc']
```

Removing elements that have consecutive duplicates

```
L = [1,1,1,1,1,1,2,3,4,4,5,1,2]
from itertools import groupby
[x[0] for x in groupby(L)]
```

```
[1, 2, 3, 4, 5, 1, 2]
```

Zip two given lists of lists.

```
list1 = [[1, 3], [5, 7], [9, 11]]
list2 = [[2, 4], [6, 8], [10, 12, 14]]
result = list(map(list.__add__, list1, list2))
print("\nZipped list:\n" +  str(result))
```

```
Zipped list:
[[1, 3, 2, 4], [5, 7, 6, 8], [9, 11, 10, 12, 14]]
```

Find the difference between consecutive numbers in a given list

```
def diff_consecutive_nums(nums):
    result = [b-a for a, b in zip(nums[:-1], nums[1:])]
    return result

nums1 = [1, 1, 3, 4, 4, 5, 6, 7]
print(diff_consecutive_nums(nums1))
```

```
[0, 2, 1, 0, 1, 1, 1]
```

Remove all elements from a given list present in another list

```
list1 = [1,2,3,4,5,6,7,8,9,10]
list2 = [2,4,6,8]
[x for x in list1 if x not in list2]
```

```
[1, 3, 5, 7, 9, 10]
```

Remove all spaces in string

```
a = 'My name   is Kunal  My country name    is India  '
a = ' '.join(a.split())
print(a)
```

```
My name is Kunal My country name is India
```

Reverse string in list

```
colors_list = ["Red", "Green", "Blue", "White", "Black"]
print([x[::-1] for x in colors_list])

L = len(colors_list)
for i in range(int(L / 2)):
    n = colors_list[i]
    colors_list[i] = colors_list[L - i - 1]
```

```
    colors_list[L - i - 1] = n
print(colors_list)
```

```
['deR', 'neerG', 'eulB', 'etihW', 'kcalB']
['Black', 'White', 'Blue', 'Green', 'Red']
```

Remove specific words from list

```
colors = ['red', 'green', 'blue', 'white', 'black', 'orange']
remove_colors = ['white', 'orange']
for word in colors:
        if word in remove_colors:
            colors.remove(word)
print(colors)
```

```
['red', 'green', 'blue', 'black']
```

to pair up the consecutive elements of a given list.

```
nums =  [1,2,3,4,5,6]
resp = [[nums[i], nums[i + 1]] for i in range(len(nums) - 1)]
print(resp)
```

```
[[1, 2], [2, 3], [3, 4], [4, 5], [5, 6]]
```

palindrom or not

```
fp = 'MADAM'
if fp == fp[:: - 1]:
    print('Palindrom')
else:
    print('not')
```

```
Palindrom
```

Swap using 2 variables

```
a = 5
b = 6
print(a,b)
a=a-b
b=a+b
a=b-a
print(a,b)
```

```
5 6
6 5
```

Check a given number is perfect or not in Python

```
# #6 is a positive number and its divisor is 1,2,3 and 6 itself.
# But we should not include 6 as by the definition of perfect number.
# Lets add its divisor excluding itself
# 1+2+3 = 6 which is equal to number itself.
# It means 6 is a Perfect Number.
```

```
num = 6
sum=0
for i in range(1,(num//2)+1):
    remainder = num % i
    if remainder == 0:
        sum = sum + i
if sum == num:
    print("given no. is perfect number")
else:
    print("given no. is not a perfect number")
```

```
given no. is perfect number
```

Fibonacci

```
n = 5
first,second=0,1

for i in range(0,n):
    if i<=1:
        result=i
    else:
        result = first + second;
        first = second;
        second = result;
    print(result)
```

```
0
1
1
2
3
```

given number is prime or not

```
i,temp=0,0
n = 3
for i in range(2,n//2):
    if n%i == 0:
        temp=1
        break
if temp == 1:
    print("given number is not prime")
else:
    print("given number is prime")
```

```
given number is prime
```

Reverse number

```
n = 4567

reverse = 0
```

```
while n!=0:
    reverse = reverse*10 + n%10
    n = (n//10)
print(reverse)
```

```
7654
```

Count the words frequency

```
from nltk import FreqDist
nlp_str = str('received capital one charge card offer xxxx applied was
accepted limit activated card and used for xxxx presents charge card
xxxx right after activating card capital one sent me another card with
same limit never activated never used that card first bill from above
card came due xxxx and minimum payment due was i sent in via uspmo and
sent in before due date with the xxxx non activated non used credit
card they also sent me bill for some yearly fees when never even
activated the card so called them up told them did not want the card
and sent back to them well get my next bill from the card above xxxx
they did not credit me for the payment and charged me outrageous over
the limit fees late fees etc and now payment due so i called up their
rep stated they accidentally applied my payment to wrong account
number and would be corrected so i sent in a payment via uspmo along
with note to make sure account corrected and both payments applied to
correct account number minimum due and wanted to keep that card and
also repair my credit from bankruptcy in xxxx or xxxx')
FreqDist(nlp_str).most_common(10)
```

```
[(' ', 199),
 ('e', 113),
 ('t', 84),
 ('a', 83),
 ('n', 58),
 ('d', 57),
 ('r', 54),
 ('c', 48),
 ('i', 47),
 ('o', 47)]
```

Count frequency of words

```
a = 'My name   is Kunal  My country name    is India  '
d = dict()
a = list(a)
count, char = [], []
for i in a:
    count.append(a.count(i))
    char.append(i)
print(dict(zip(char, count)))
```

```
{'M': 2, 'y': 3, ' ': 16, 'n': 5, 'a': 4, 'm': 2, 'e': 2, 'i': 3, 's':
2, 'K': 1, 'u': 2, 'l': 1, 'c': 1, 'o': 1, 't': 1, 'r': 1, 'I': 1,
'd': 1}
```

Ecludian distance

```
v1 = [3, 1, 5]
v2 = [2, 5, 2]
# ecludian_dist = squareRoot( square(v1[1]-v2[1]) + square(v1[2]-
v2[2]) + square(v3[1]-v3[1]) )
#               = squareRoot( square(3-2) + square(1-5) + square(5-
2) )
#               = squareRoot( 1 + 16 + 9)
#               squareRoot( 26 ) = 5.12
```

Sort the values pandas by ziping the list

```
import pandas as pd
Name = ['tom', 'krish', 'nick', 'juli']
Age = [25, 30, 26, 22]
df = pd.DataFrame(data=list(zip(Name, Age)), columns=['Name', 'Age'])
df.sort_values(by=['Age'])
```

```
    Name  Age
3   juli   22
0    tom   25
2   nick   26
1  krish   30
```