LIST BASED PROGRAMMING QUESTIONS ¶

SET - 01

1. WAPP to sum all the items

Input:
[1,7,-10,34,2,-8]

Output:
Sum all the items = 26

2. WAPP to multiply all the items in a list

Input:
[3,4,5,4,7]

Output:
Multiply all the items in a list: 1680

3. WAPP to get the largest number from a list

Input:
[1,7,10,34,2,8]

Output:
Largest Number: 34

4. WAPP to get the smallest number from a list

Input:
[51,7,10,34,2,8]

Output:
Smallest Number: 2

5. WAPP to count the number of strings where the string length is 2 or more and the first and last character are same from a give list of strings

Input:
['abc','xyz','aba','1221']

Output:
Find and Last Character are same: 2

6. WAPP to remove duplicates from a list

Input:
[1,2,3,7,2,1,5,6,4,8,5,4]

Output:
[1,2,3,4,5,6,7,8]

7. WAPP to check a list is empty or not

Input:
[3,4,5,4,7]

Output:
List is Not Empty

8. WAPP to clone or copy a list

Input:
[3,4,5,4,7]

Output:
Clone or Copy a List: [10,22,44,23,4]

9. WAPP to find the list of words that are longer than n from a given list of words.

Input:
Find the List of Words that are Longer than n from a given List of Words Given value of n = 4

Output:
['Words', 'Longer', 'given', 'Words']

10. WAPP that get two lists as input and check if they have at least one common member.

Input:
[1,2,3,4,5] [5,6,7,8,9]

Output:
Lists have atleast one common member

SOLUTION -- SET 01

In [90]:
```
# 01
item=[1,7,-10,34,2,-8]
sum = 0
for i in item:
    sum += i
print(sum)
```

```
26
```

In [91]:
```
# 02
item=[3,4,5,4,7]
total = 1
for i in item:
    total *= i
print(total)
```

```
1680
```

In [92]:
```
# 03
a=[1,7,10,34,2,8]
print("Largest Number :",max(a))
```

```
Largest Number : 34
```

In [93]:
```
# 03 <- ALTERNATIVE SOLUTION
a=[1,7,10,34,2,8]
max_v = a[0]
for i in a:
    if i > max_v:
        max_v = i
print("Largest Number :",max_v)
```

```
Largest Number : 34
```

In [94]:
```
# 04
a=[51,7,10,34,2,8]
print("Smallest Number :",min(a))
```

```
Smallest Number : 2
```

In [95]:
```
# 04 <-- ALTERNATIVE SOLUTION
a=[51,7,10,34,2,8]
min_num = a[0]
for i in a:
    if i < min_num:
        min_num = i
print("Smallest Number :",min_num)
```

```
Smallest Number : 2
```

In [96]:
```
# 05
"""
Sample List: ['abc', 'xyz', 'aba', '1221']
Expected Result: 2
"""

word=["madem","3643","apple","3756"]
ch = 0
for w in word:
    if len(w) > 1 and w[0] == w[-1]:
        ch += 1
print(ch)
```

```
2
```

In [97]:
```
# 06
a = [1,2,3,7,2,1,5,6,4,8,5,4]
b=set(a)
print(list(b))
```

```
[1, 2, 3, 4, 5, 6, 7, 8]
```

In [98]:
```
# 06 <-- ALTERNATIVE SOLUTION
a = [1,2,3,7,2,1,5,6,4,8,5,4]
dup = set()
uniq = []
for x in a:
    if x not in dup:
        uniq.append(x)
        dup.add(x)
print(list(dup))
```

```
[1, 2, 3, 4, 5, 6, 7, 8]
```

In [99]:
```
# 07
a = []
if not a:
    print("List is Empty...")
else:
    print("List is Not Empty...")
```

```
List is Empty...
```

In [100]:
```
# 08
old_list = [10, 22, 44, 23, 4]
new_list = list(old_list)

print("Old List : ",old_list)
print("New List : ",new_list)
```

```
Old List :  [10, 22, 44, 23, 4]
New List :  [10, 22, 44, 23, 4]
```

In [101]:
```
# 09
n=4
str1 ="Find the List of Words that are Longer than n from a given List of Words"
new_list = []

text = str1.split(" ")

for x in text:
    if len(x) > n:
        new_list.append(x)
print(new_list)
```

```
['Words', 'Longer', 'given', 'Words']
```

In [102]:
```
# 10
list1=[1,2,3,4,5]
list2=[5,6,7,8,9]

result = False
for x in list1:
    for y in list2:
        if x == y:
            result = True
            print(result)
if result:
    print("Lists have at least one common member")
else:
    print("Lists do not have any common member")
```

```
True
Lists have at least one common member
```

SET - 02

11. WAPP to print a specified list after removing the 0th, 4th and 5th elements. (enumerate)

Input:
["Cat", "Dog", "Elephant", "Fox", "Tiger", "Lion", "Ponda"]

Output:
['Dog', 'Elephant', 'Fox', 'Ponda']

12. WAPP to print the numbers of a specified list after removing even numbers from it

Input:
[7,32,81,20,25,14,23,27]

Output:
[7, 81, 25, 23, 27]

13. WAPP to shuffle and print a specified list (shuffle)

Input:
["Cat", "Dog", "Elephant", "Fox", "Tiger", "Lion", "Ponda"]

Output:
['Fox', 'Cat', 'Tiger', 'Lion', 'Dog', 'Ponda', 'Elephant']

14. WAPP to generate and print a list of first and last 5 elements where the values are square of numbers between 1 and 30

Output:
First 5 elements : [1, 4, 9, 16, 25]
Last 5 elements : [625, 676, 729, 784, 841]`

15. WAPP to generate all permutations of a list in Python. (itertools)

Input:
[1,2,3]

Output:
[(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1)]

16. WAPP to convert a list of characters into a string

Input:
['D','P','S',' ', 'R','u','b','y']

Output:
DPS Ruby

17. WAPP to find the index of an item in a specified list

Input:
[20, 70, 30, 90, 10, 30, 90, 10, 80]

Output:
Item to find the index of 30

Index Number of Item = 2

18. WAPP to flatten a shallow list

Input:
[[20,30,70],[30,90,10], [30,20], [70,90,10,80]]

Output:
[20, 30, 70, 30, 90, 10, 30, 20, 70, 90, 10, 80]

19. WAPP to add a list to the second list

Input:
[10, 20, 30, 40]

["Cat", "Dog", "Lion", "Ponda"]

Output:
[10, 20, 30, 40, 'Cat', 'Dog', 'Lion', 'Ponda']

20. WAPP to select an item randomly from a list Using random.choice()

Input:
["Cat", "Dog", "Elephant", "Fox", "Tiger", "Lion", "Ponda"]

Output:
Item randomly from a list : Fox

SOLUTION -- SET 02

In [103]:
```
# 11
animal = ["Cat", "Dog", "Elephant", "Fox", "Tiger", "Lion", "Ponda"]
a = []
for i, x in enumerate(animal):
    if i not in (0,4,5):
        a.append(x)
print(a)
```

```
['Dog', 'Elephant', 'Fox', 'Ponda']
```

In [104]:
```
# 11 <-- ALTERNATE SOLUTION USING LIST COMPREHENSION
animal = ["Cat", "Dog", "Elephant", "Fox", "Tiger", "Lion", "Ponda"]
animal = [x for (i,x) in enumerate(animal) if i not in (0,4,5)]
print(animal)
```

```
['Dog', 'Elephant', 'Fox', 'Ponda']
```

In [105]:
```
# 12
n = [7,32,81,20,25,14,23,27]
n1 = []
for x in n:
    if x%2 != 0:
        n1.append(x)
print(n1)
```

```
[7, 81, 25, 23, 27]
```

In [106]:
```
# 12 <- ALTERNATE SOLUTION
n = [7,32,81,20,25,14,23,27]
n = [x for x in n if x%2!=0]
print(n)
```

```
[7, 81, 25, 23, 27]
```

In [107]:
```
# 13
from random import shuffle
animal = ["Cat", "Dog", "Elephant", "Fox", "Tiger", "Lion", "Ponda"]
shuffle(animal)
print(animal)
```

```
['Elephant', 'Tiger', 'Lion', 'Cat', 'Fox', 'Dog', 'Ponda']
```

In [108]:
```
# 14
l = list()
for i in range(1,25):
    l.append(i**2)
print(l[:5])
print(l[-5:])
```

```
[1, 4, 9, 16, 25]
[400, 441, 484, 529, 576]
```

In [109]:
```
# 15
import itertools
print(list(itertools.permutations([1,2,3])))
```

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

In [110]:
```
# 16
s = ['D','P','S',' ', 'R','u','b','y']
s1 = ''.join(s)
print(s1)
```

```
DPS Ruby
```

In [111]:
```
# 17
num = [20, 70, 30, 90, 10, 30, 90, 10, 80]
print(num.index(30))
```

```
2
```

In [112]:
```
# 18
ori_list = [[20,30,70],[30,90,10], [30,20], [70,90,10,80]]
flatten = []
for i in ori_list:
    for j in i:
        flatten.append(j)
print(flatten)
```

```
[20, 30, 70, 30, 90, 10, 30, 20, 70, 90, 10, 80]
```

In [113]:
```
# 18 <-- ALTERNATE SOLUTION
import itertools

ori_list = [[20,30,70],[30,90,10], [30,20], [70,90,10,80]]
merged_list = list(itertools.chain(*ori_list))
print(merged_list)
```

```
[20, 30, 70, 30, 90, 10, 30, 20, 70, 90, 10, 80]
```

In [114]:
```
# 19
Number = [10, 20, 30, 40]
animal = ["Cat", "Dog", "Lion", "Ponda"]
final_list = Number + animal
print(final_list)
```

```
[10, 20, 30, 40, 'Cat', 'Dog', 'Lion', 'Ponda']
```

In [115]:
```
# 20
import random
animal = ["Cat", "Dog", "Elephant", "Fox", "Tiger", "Lion", "Panda"]
print(random.choice(animal))
```

```
Elephant
```

SET - 03

21. WAPP to check whether two lists are circularly identical

Input:
[8, 8, 12, 12, 8]

[8, 8, 8, 12, 12]

[1, 8, 8, 12, 12]

Output:
Compare List1 and List2 : True

Compare List1 and List3 : False

22. WAPP to find the second smallest number in a list

Input:
[2,4,56,78,4,34,5,8,9]

Output:
Second Smallest Number : 4

23. WAPP to find the second largest number in a list

Input:
[82,4,56,78,4,34,5,100,9]

Output:
Second Largest Number : 82

24. WAPP to get unique values from a list

Input:
[82, 4, 10, 56, 78, 4, 34, 5, 10, 9]

Output:

25. WAPP to get the frequency of the elements in a list.

Input:
[10, 30, 50, 10, 20, 60, 20, 60, 40, 40, 50, 50, 30]

Output:
{50: 3, 10: 2, 30: 2, 20: 2, 60: 2, 40: 2}

26. Create a list by concatenating a given list which range goes from 1 to n

Input:
['T', 'J']

N = 10

Output:
['T1', 'J1', 'T2', 'J2', 'T3', 'J3', 'T4', 'J4', 'T5', 'J5', 'T6', 'J6', 'T7', 'J7', 'T8', 'J8', 'T9', 'J9', 'T10', 'J10']

27. WAPP to get variable unique identification number or string

Input:
x = 30

s = "DPS Ruby"

Output:
Unique Identification Number : 7005f980

Unique Identification String : c24bb0

28. WAPP to find common items from two lists

Input:
[23,45,67,78,89,34]

[34,89,55,56,39,67]

Output:
Common items from two lists : {89, 34, 67}

29. WAPP to split a list based on first character of word

Input:
["cat", "dog", "cow", "tiger", "lion", "Fox", "Shark", "Snake", "turtle", "mouse", "monkey", "bear"]

Output:
F
Fox
S
Shark
Snake

b
bear
c
cat
cow
d
dog
l
lion
m
monkey
mouse
t
tiger
turtle

30. WAPP to select the odd number of a list

Input:
[1,2,4,3,6,7,5,8,9,7,8,9,10]

Output:
[1, 3, 7, 5, 9, 7, 9]

SOLUTION -- SET 03

In [116]:
```
# 21
list1 = [8, 8, 12, 12, 8]
list2 = [8, 8, 8, 12, 12]
list3 = [1, 8, 8, 12, 12]

print("Compare List1 and List2 : ",' '.join(map(str, list2)) in ' '.join(map(str, li
print("Compare List1 and List3 : ",' '.join(map(str, list3)) in ' '.join(map(str, li
```

```
Compare List1 and List2 :  True
Compare List1 and List3 :  False
```

In [117]:

```
# 22
num = [2,4,56,78,4,34,5,8,9]

if len(num)<2:
    print(num)

if (len(num)==2)  and (num[0] == num[1]):
    print(num)

dup_items = set()
uniq_items = []
for x in num:
    if x not in dup_items:
        uniq_items.append(x)
        dup_items.add(x)

uniq_items.sort()
print(uniq_items[1])
```

```
4
```

In [118]:

```
# 23
num = [ 82,4,56,78,4,34,5,100,9]

if (len(num)<2):
    print(num)

if ((len(num)==2)  and (num[0] == num[1]) ):
    print(num)

dup_items = set()
uniq_items = []
for x in num:
    if x not in dup_items:
        uniq_items.append(x)
        dup_items.add(x)
uniq_items.sort()
print(uniq_items[-2])
```

```
82
```

In [119]:

```
# 24
l = [ 82,4,10,56,78,4,34,5,10,9]

print("Original List : ",l)

s = set(l)
new_list = list(s)

print("Unique Numbers : ",new_list)
```

```
Original List :  [82, 4, 10, 56, 78, 4, 34, 5, 10, 9]
Unique Numbers :  [34, 4, 5, 9, 10, 78, 82, 56]
```

In [120]:

```
# 25
import collections
l = [10,30,50,10,20,60,20,60,40,40,50,50,30]

print("Original List : ",l)

f = collections.Counter(l)

print("Frequency of the Elements: ",f)
```

```
Original List :  [10, 30, 50, 10, 20, 60, 20, 60, 40, 40, 50, 50, 30]
Frequency of the Elements:  Counter({50: 3, 10: 2, 30: 2, 20: 2, 60: 2, 40: 2})
```

In [121]:

```
# 26
ch = ['T', 'J']
n = 10
new_list = ['{}{}'.format(a, b) for b in range(1, n+1) for a in ch]
print(new_list)
```

```
['T1', 'J1', 'T2', 'J2', 'T3', 'J3', 'T4', 'J4', 'T5', 'J5', 'T6', 'J6', 'T7', 'J7', 'T
8', 'J8', 'T9', 'J9', 'T10', 'J10']
```

In [122]:

```
# 27
x = 30
print(format(id(x), 'x'))
s = "DPS Ruby"
print(format(id(s), 'x'))
```

```
7ffcb4a0d6c8
2136278fdb0
```

In [123]:

```
# 28
num1 = [23,45,67,78,89,34]
num2 = [34,89,55,56,39,67]
print(set(num1) & set(num2))
```

```
{89, 34, 67}
```

In [124]:

```
# 29
from itertools import groupby
from operator import itemgetter

a = ["cat","dog","cow","tiger","lion","Fox","Shark","Snake","turtle","mouse","monkey

for ltr, wds in groupby(sorted(a), key=itemgetter(0)):
    print(ltr)
    for w in wds:
        print(" ", w)
        print("")
```

```
F
  Fox

S
  Shark

  Snake

b
  bear

c
  cat

  cow

d
  dog

l
  lion

m
  monkey

  mouse

t
  tiger

  turtle
```

In [125]:

```
# 30
a=[1,2,4,3,6,7,5,8,9,7,8,9,10]
odd_num=[]

for i in a:
    if(i%2==1):
        #print(i)
        odd_num.append(i)

print(odd_num)
```

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

SET - 04

31. WAPP to count unique values inside a list

Input:
[10, 20, 30, 50, 80, 70, 70, 80, 10]

Output:
No of Unique Items in List : 6

32. WAPP to List product excluding duplicates

Input:
[2, 1, 2, 4, 6, 4, 3, 2, 1]

Output:
Duplication removal list product : 144

33. WAPP to Extract elements with Frequency greater than K

Input:
[4, 6, 4, 3, 3, 4, 3, 7, 8, 8]

Output:

34. WAPP to Test if List contains elements in Range

Input:
[4, 5, 6, 7, 3, 9]

Output:
Does list contain all elements in range : True

35. WAPP to check if the list contains three consecutive common numbers in Python

Input:
[18, 18, 18, 6, 3, 4, 9, 9, 9]

Output:
Three Consecutive common numbers = 18, 9

36. WAPP to find the Strongest Neighbour

Input:
[10,20,30,20,30,400]

Output:
20 30 30 30 400

37. WAPP to print all Possible Combinations from the three Digits

Input:
[1, 2, 3]

Output:
1 2 3

1 3 2

2 1 3

2 3 1

3 1 2

3 2 1

38. WAPP to find all the Combinations in the list with the given condition

Input:
['DPS Ruby', ['Software', 'Computer'], ['Solution', 'Education']]

Output:
[['DPS Ruby', 'Software', 'Solution'], ['DPS Ruby', 'Computer', 'Education']]

39. WAPP to get all unique combinations of two Lists

Input:
['A','B','C']

[1,2,3]

Output:
[[('A', 1), ('B', 2), ('C', 3)], [('A', 1), ('C', 2), ('B', 3)], [('B', 1), ('A', 2), ('C', 3)], [('B', 1), ('C', 2), ('A', 3)], [('C', 1), ('A', 2), ('B', 3)], [('C', 1), ('B', 2), ('A', 3)]]

40. WAPP to remove all the occurrences of an element from a list

Input:
[1, 3, 4, 6, 5, 1]

Output:
[3, 4, 6, 5]

SOLUTION -- SET 04

In [126]:
```
# 31
#First Method

a = [10, 20, 30, 50, 80, 70, 70, 80, 10]
s = set(a)
print("No of Unique Items in List :", len(s))
```

```
No of Unique Items in List : 6
```

In [127]:
```
# 31
# Second Method
a = [10, 20, 30, 50, 80, 70, 70, 80, 10]
l = []
count = 0
for i in a:
    if i not in l:
        count += 1
        l.append(i)

print("No of Unique Items in List :", count)
```

```
No of Unique Items in List : 6
```

In [128]:
```
# 32
a = [2,1,2,4,6,4,3,2,1]
print ("Original list : " + str(a))

b=list(set(a))
p=1
for i in b:
    p*=i

print("Duplication removal list product : " + str(p))
```

```
Original list : [2, 1, 2, 4, 6, 4, 3, 2, 1]
Duplication removal list product : 144
```

In [129]:
```
# 33
a = [4, 6, 4, 3, 3, 4, 3, 7, 8, 8]
print("Original list : " + str(a))
K = 2

res = []
for i in a:
    freq = a.count(i)
    if freq > K and i not in res:
        res.append(i)

print("The Required Elements : " + str(res))
```

```
Original list : [4, 6, 4, 3, 3, 4, 3, 7, 8, 8]
The Required Elements : [4, 3]
```

In [130]:
```
# 34
a = [4, 5, 6, 7, 3, 9]

print("Original list is : " + str(a))

i, j = 3, 10

res = True
for e in a:
    if e < i or e >= j :
        res = False
        break

print ("Does list contain all elements in range : " + str(res))
```

```
Original list is : [4, 5, 6, 7, 3, 9]
Does list contain all elements in range : True
```

In [131]:
```
# 35
# creating the aay
a = [18, 18, 18, 6, 3, 4, 9, 9, 9]
l = len(a)
for i in range(l - 2):
    if a[i] == a[i + 1] and a[i + 1] == a[i + 2]:
        print(a[i])
```

```
18
9
```

In [132]:
```
# 36
n = 6
a1 = [10,20,30,20,30,400]
a2 = []
for i in range(1, n):
    r = max(a1[i], a1[i-1])
    a2.append(r)
for i in a2 :
    print(i,end=" ")
```

```
20 30 30 30 400
```

In [133]:
```
# 37
a = [1, 2, 3]
for i in range(3):
    for j in range(3):
        for k in range(3):
            if (i!=j and j!=k and i!=k):
                print(a[i], a[j], a[k])
```

```
1 2 3
1 3 2
2 1 3
2 3 1
3 1 2
3 2 1
```

In [134]:
```python
# 38
val = ["DPS Ruby",["Software","Computer"], ["Solution", "Education"]]
print("Original List : " + str(val))
a = 2
l = []
c = 0

while c <= a - 1:
    t = []
    for i in val:
        if not isinstance(i, list):
            t.append(i)
        else:
            t.append(i[c])
    c += 1
    l.append(t)

print("\nIndex Combinations : " + str(l))
```

```
Original List : ['DPS Ruby', ['Software', 'Computer'], ['Solution', 'Education']]

Index Combinations : [['DPS Ruby', 'Software', 'Solution'], ['DPS Ruby', 'Computer', 'E
ducation']]
```

In [135]:
```python
# 39
import itertools
from itertools import permutations
l1 = ['A','B','C']
l2 = [1,2,3]
unique = []
permut = itertools.permutations(l1, len(l2))
for comb in permut:
    zipped = zip(comb, l2)
    unique.append(list(zipped))

print(unique)
```

```
[[('A', 1), ('B', 2), ('C', 3)], [('A', 1), ('C', 2), ('B', 3)], [('B', 1), ('A', 2),
('C', 3)], [('B', 1), ('C', 2), ('A', 3)], [('C', 1), ('A', 2), ('B', 3)], [('C', 1),
('B', 2), ('A', 3)]]
```

In [136]:
```python
# 40
val = [1, 3, 4, 6, 5, 1]
a = 1
print ("Original list :" ,val)

c = val.count(a)
for i in range(c):
    val.remove(a)

print ("Remove operation :" , val)
```

```
Original list : [1, 3, 4, 6, 5, 1]
Remove operation : [3, 4, 6, 5]
```

SET - 05

41. WAPP to Remove Consecutive K element records

Input:
[('A', 'B', 'C', 'D'), ('B', 'C', 'C', 'I'), ('H', 'D', 'B', 'C'), ('C', 'C', 'G', 'F')]

Output:
[('A', 'B', 'C', 'D'), ('H', 'D', 'B', 'C')]

42. WAPP to Replace index elements with elements in Other List

Input:
[['DPS Ruby', 'Computer', 'Education']

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

Output:
['Education', 'Computer', 'DPS Ruby', 'Computer', 'DPS Ruby', 'Education', 'Education', 'DPS Ruby', 'Computer', 'DPS Ruby', 'Computer', 'Education']

43. WAPP to Retain records with N occurrences of K

Input:
[(4, 5, 6, 5, 4), (4, 5, 3), (5, 5, 2), (3, 4, 9)]

K = 5

N = 2

Output:
[(4, 5, 6, 5, 4), (5, 5, 2)]

44. WAPP to Swap elements in String list

Input:
['DPS', 'Ruby', 'Computer', 'Education']

Output:
['DPS', "Ruby", 'Software', 'Solutions']

45. WAPP to reverse All Strings in String List

Input:
Original list : ['DPS', 'ruby', 'Computer', 'Education']

Output:
Reversed list : ['SPD', 'ybur', 'retupmoC', 'noitacudE']

46. WAPP to find the character position of Kth word from a list of strings

Input:
['DPS', 'ruby', 'Computer', 'Education']

K = 20

Output:
Index of character at Kth position word : 5

47. WAPP to Prefix frequency in string List

Input:
['TjC', 'TjCpp', 'TjPython', 'Java']

Prefix = 'Tj'

Output:
Strings count with matching frequency : 3

48. WAPP to Split Strings on Prefix Occurrence

Input:
['TjC', 'TjCpp', 'TjPython', 'Java', 'tj']

Prefix = 'Tj'

Output:
[['TjC'], ['TjCpp'], ['TjPython', 'Java', 'tj']]

49. WAPP to Replace all Characters of a List Except the given character

Input:
['P', 'Y', 'T', 'H', 'O', 'N']

Output:
['@', '@', 'T', '@', '@', '@']

50. WAPP to Add Space between Potential Words

Input:
['DPSRuby', 'ComputerEducations']

Output:
[' D P S Ruby', ' Computer Educations']

SOLUTION -- SET 05

In [137]:
```
# 41
val = [('A', 'B', 'C', 'D'), ('B', 'C', 'C', 'I'), ('H', 'D', 'B', 'C'), ('C', 'C',
print("Original List : " + str(val))
K = 'C'
res = [i for i in val if not any(i[j] == K and i[j + 1] == K for j in range(len(i) -

print("After Removal : " , res)
```

```
Original List : [('A', 'B', 'C', 'D'), ('B', 'C', 'C', 'I'), ('H', 'D', 'B', 'C'),
('C', 'C', 'G', 'F')]
After Removal :  [('A', 'B', 'C', 'D'), ('H', 'D', 'B', 'C')]
```

In [138]:
```
# 42
a = ['DPS', 'Computer', 'Education']
b = [2 ,1 ,0 ,1 ,0 ,2 ,2 ,0 ,1 ,0 ,1 ,2]
print("List 1 : " , a)
print("List 2 : " , b)
res = [a[i] for i in b]

print ("After Index Elements Replacements is : " ,res)
```

```
List 1 :  ['DPS', 'Computer', 'Education']
List 2 :  [2, 1, 0, 1, 0, 2, 2, 0, 1, 0, 1, 2]
After Index Elements Replacements is :  ['Education', 'Computer', 'DPS', 'Computer', 'D
PS', 'Education', 'Education', 'DPS', 'Computer', 'DPS', 'Computer', 'Education']
```

In [139]:
```
# 43
val = [(4, 5, 6, 5, 4), (4, 5, 3), (5, 5, 2), (3, 4, 9)]
print(val)
K = 5
N = 2
res = [e for e in val if e.count(K) == N]

print(res)
```

```
[(4, 5, 6, 5, 4), (4, 5, 3), (5, 5, 2), (3, 4, 9)]
[(4, 5, 6, 5, 4), (5, 5, 2)]
```

In [140]:
```
# 44
s = ["DPS","ruby","Computer","Education"]

print("Before Swap :",s)

res = [sub.replace("ruby","Ruby").replace("Computer", "Software").replace("Education

print ("After Swap : ",res)
```

```
Before Swap : ['DPS', 'ruby', 'Computer', 'Education']
After Swap :  ['DPS', 'Ruby', 'Software', 'Solutions']
```

In [141]:
```
# 44 <-- Alternate Solution
s = ["DPS","ruby","Computer","Education"]

print("Before Swap :",s)

res = []
for i in s:
    res.append(i.replace("ruby","Ruby").replace("Computer", "Software").replace("Edu

print ("After Swap : ",res)
```

```
Before Swap : ['DPS', 'ruby', 'Computer', 'Education']
After Swap :  ['DPS', 'Ruby', 'Software', 'Solutions']
```

In [142]:
```python
# 45
val = ["DPS","ruby","Computer","Education"]
print ("Original list : ", val)

#First Methods

res = [i[::-1] for i in val]
print ("Reversed list : " , res)
```

```
Original list :  ['DPS', 'ruby', 'Computer', 'Education']
Reversed list :  ['SPD', 'ybur', 'retupmoC', 'noitacudE']
```

In [143]:
```python
# 45 <- ALTERNATE SOLUTION
# Second Methods
val = ["DPS","ruby","Computer","Education"]
print ("Original list : ", val)
print("Reversed list : " ,val[::-1])
```

```
Original list :  ['DPS', 'ruby', 'Computer', 'Education']
Reversed list :  ['Education', 'Computer', 'ruby', 'DPS']
```

In [144]:
```python
# 46
val = ["DPS","ruby","Computer","Education"]
print("The original list is : " ,val)
K = 20
res = [i[0] for sub in enumerate(val) for i in enumerate(sub[1])]
res = res[K]
print("Index of character at Kth position word : " + str(res))
```

```
The original list is :  ['DPS', 'ruby', 'Computer', 'Education']
Index of character at Kth position word : 5
```

In [145]:
```python
# 47
val = ["TjC","TjCpp","TjPython","Java"]
print("Original List : ",val)
sub = 'Tj'
res = 0
for e in val:
    if e.startswith(sub):
        res = res + 1
print ("Strings count with matching frequency : ",res)
```

```
Original List :  ['TjC', 'TjCpp', 'TjPython', 'Java']
Strings count with matching frequency :  3
```

In [146]:
```python
# 49
l = ["TjC","TjCpp","TjPython","Java","tj"]
print("The original list is : ",l)
pref = "Tj"

res = []
for val in l:
    if val.startswith(pref):
        res.append([val])
    else:
        res[-1].append(val)
print("Prefix Split List : " + str(res))
```

```
The original list is :  ['TjC', 'TjCpp', 'TjPython', 'Java', 'tj']
Prefix Split List : [['TjC'], ['TjCpp'], ['TjPython', 'Java', 'tj']]
```

In [147]:
```python
# 49
val = ['P', 'Y', 'T', 'H', 'O', 'N']
print("The original list : " + str(val))

res = [i if i == 'T' else '@' for i in val]
print("List after replacement : " + str(res))
```

```
The original list : ['P', 'Y', 'T', 'H', 'O', 'N']
List after replacement : ['@', '@', 'T', '@', '@', '@']
```

In [148]:
```python
# 50
val = ["DPSRuby", "ComputerEducations"]
print("Original list : " ,val)

res = []
for i in val:
    t = [[]]
    for ch in i:
        if ch.isupper():
            t.append([])
        t[-1].append(ch)
    res.append(' '.join(''.join(i) for i in t))

print("The space added list of strings : " , res)
```

```
Original list :  ['DPSRuby', 'ComputerEducations']
The space added list of strings :  [' D P S Ruby', ' Computer Educations']
```

THANK YOU FOR SOLVING 50 QUESTIONS 😁