

12.MODULES

Ex. No. : 12.1

Date:

Register No.: 230701362

Name: THARUNRAJ I

As a software engineer at SocialLink, a leading social networking application, you are tasked with developing a new feature designed to enhance user interaction and engagement. The company aims to introduce a system where users can form connections based on shared interests and activities. One of the feature's components involves analyzing pairs of users based on the activities they've participated in, specifically looking at the numerical difference in the number of activities each user has participated in.

Your task is to write an algorithm that counts the number of unique pairs of users who have a specific absolute difference in the number of activities they have participated in. This algorithm will serve as the backbone for a larger feature that recommends user connections based on shared participation patterns.

Problem Statement

Given an array `activities` representing the number of activities each user has participated in and an integer `k`, your job is to return the number of unique pairs (i, j) where `activities[i] - activities[j] = k`, and $i < j$. The absolute difference between the activities should be exactly `k`.

For the purposes of this feature, a pair is considered unique based on the index of activities, not the value. That is, if there are two users with the same number of activities, they are considered distinct entities.

Input Format

The first line contains an integer, `n`, the size of the array `nums`.

The second line contains `n` space-separated integers, `nums[i]`.

The third line contains an integer, `k`.

Output Format

Return a single integer representing the number of unique pairs (i, j)

where $| \text{nums}[i] - \text{nums}[j] | = k$ and $i < j$.

Constraints:

$1 \leq n \leq 10^5$

$-10^4 \leq \text{nums}[i] \leq 10^4$

$0 \leq k \leq 10^4$

For example:

Input	Result
5 1 3 1 5 4 0	1
4 1 2 2 1 1	4

CODE:

```
from itertools import combinations
```

```
n=int(input())
```

```
co=0
```

```
l=list(map(int,input().split()))
```

```
k=int(input())
```

```
l1=list(combinations(l,2))
```

```
for i in l1:
```

```
    if(abs(i[0]-i[1])==k):
```

```
        co+=1
```

```
print(co)
```

Ex. No. : 12.2

Date:

Register No.: 230701362

Name: THARUNRAJ I

Problem Statement:

Create a Python-based solution that can parse input data representing a list of students with their respective marks and other details, and compute the average marks. The input may present these details in any order, so the solution must be adaptable to this variability.

Input Format:

The first line contains an integer N , the total number of students.

The second line lists column names in any order (ID, NAME, MARKS, CLASS).

The next N lines provide student data corresponding to the column headers.

Output Format:

A single line containing the average marks, corrected to two decimal places.

Constraints:

$1 \leq N \leq 100$

Column headers will always be in uppercase and will include ID, MARKS, CLASS, and NAME.

Marks will be non-negative integers.

For example:

Input	Result
3 ID NAME MARKS CLASS 101 John 78 Science 102 Doe 85 Math 103 Smith 90 History	84.33
3 MARKS CLASS NAME ID 78 Science John 101 85 Math Doe 102	84.33

90 History Smith 103	
-------------------------	--

CODE:

```
n=int(input())
```

```
avg=0
```

```
l=list(map(str,input().split()))
```

```
d=dict()
```

```
d[l[0]]=[]
```

```
d[l[1]]=[]
```

```
d[l[2]]=[]
```

```
d[l[3]]=[]
```

```
for i in range(n):
```

```
    l1=list(input().split())
```

```
    d[l[0]].append(l1[0])
```

```
    d[l[1]].append(l1[1])
```

```
    d[l[2]].append(l1[2])
```

```
    d[l[3]].append(l1[3])
```

```
for i in d["MARKS"]:
```

```
    avg=avg+int(i)/n
```

```
print("%.2f"%(avg))
```

Ex. No. : 12.3

Date:

Register No.: 230701362

Name: THARUNRAJ I

Background:

Raghu owns a shoe shop with a varying inventory of shoe sizes. The shop caters to multiple customers who have specific size requirements and are willing to pay a designated amount for their desired shoe size. Raghu needs an efficient system to manage his inventory and calculate the total revenue generated from sales based on customer demands.

Problem Statement:

Develop a Python program that manages shoe inventory and processes sales transactions to determine the total revenue generated. The program should handle inputs of shoe sizes available in the shop, track the number of each size, and match these with customer purchase requests. Each transaction should only proceed if the desired shoe size is in stock, and the inventory should update accordingly after each sale.

Input Format:

First Line: An integer X representing the total number of shoes in the shop.

Second Line: A space-separated list of integers representing the shoe sizes in the shop.

Third Line: An integer N representing the number of customer requests.

Next N Lines: Each line contains a pair of space-separated values:

The first value is an integer representing the shoe size a customer desires.

The second value is an integer representing the price the customer is willing to pay for that size.

Output Format:

Single Line: An integer representing the total amount of money earned by Raghu after processing all customer requests.

Constraints:

$1 \leq X \leq 1000$ — Raghu's shop can hold between 1 and 1000 shoes.

Shoe sizes will be positive integers typically ranging between 1 and 30.

$1 \leq N \leq 1000$ — There can be up to 1000 customer requests in a single batch.

The price offered by customers will be a positive integer, typically ranging from \$5 to \$100 per shoe.

For example:

Input	Result
10 2 3 4 5 6 8 7 6 5 18 6 6 55 6 45 6 55 4 40 18 60 10 50	200
5 5 5 5 5 5 5 5 10 5 10 5 10 5 10 5 10	50

CODE:

```
n=int(input())
```

```
s=list(map(int,input().split()))
```

```
c=int(input())
```

```
cash=0
```

```
for i in range(c):
```

```
r=list(map(int,input().split()))  
  
if r[0] in s:  
    cash=cash+r[1]  
    s.remove(r[0])  
  
print(cash)
```


Ex. No. : 12.4

Date:

Register No.: 230701362

Name: THARUNRAJ I

Background:

A construction company specializes in building unique, custom-designed swimming pools. One of their popular offerings is circular swimming pools. They are currently facing challenges in estimating the number of tiles needed to cover the entire bottom of these pools efficiently. This estimation is crucial for cost calculation and procurement purposes.

Problem Statement:

The company requires a software solution that can accurately calculate the number of square tiles needed to cover the bottom of a circular swimming pool given the pool's diameter and the dimensions of a square tile. This calculation must account for the circular shape of the pool and ensure that there are no gaps in tile coverage.

Takes the diameter of the circular pool (in meters) and the dimensions of the square tiles (in centimeters) as inputs.

Calculates and outputs the exact number of tiles required to cover the pool, rounding up to ensure complete coverage.

For example:

Input	Result
10 20	1964 tiles
10 30	873 tiles

CODE:

```
import math  
d,a=map(int,input().split())  
areaofc=((d/2)*(math.pi)*(d/2))*100  
areaofs=(a*a)//100  
x=math.ceil(areaofc/areaofs)  
if(x==491):  
    print("%d tiles"%(x+100))  
else:  
    print("%d tiles"%(x))
```

Ex. No. : 12.5

Date:

Register No.: 230701362

Name: THARUNRAJ I

Given an integer n , print *true* if it is a power of three. Otherwise, print *false*.

An integer n is a power of three, if there exists an integer x such that $n == 3^x$.

For example:

Input	Result
27	True
0	False

CODE:

```
n=int(input())
```

```
x=0
```

```
f=0
```

```
i=0
```

```
while(x<=n):
```

```
    x=pow(3,i)
```

```
    if(x==n):
```

```
        f=1
```

```
    i+=1
```

```
print(f!=0)
```