Question **1**

Correct

Mark 1.00 out of 1.00

Background:

Dr. John Wesley maintains a spreadsheet with student records for academic evaluation. The spreadsheet contains various data fields including student IDs, marks, class names, and student names. The goal is to develop a system that can calculate the average marks of all students listed in the spreadsheet.

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≤N≤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 90 History Smith 103	84.33

Answer: (penalty regime: 0 %)

```python
1  def calculate_average_marks():
2      n = int(input())
3      columns = input().split()
4      index = columns.index("MARKS")
5      total = 0
6
7      for _ in range(n):
8          data = input().split()
9          total += int(data[index])
10
11      avg = total / n if n > 0 else 0
12      print(f"{avg:.2f}")
13
14  calculate_average_marks()
15
```

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

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **2**

Correct

Mark 1.00 out of 1.00

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

An integer `n` is a power of four, if there exists an integer `x` such that `n == 4`^{x}.

For example:

Input	Result
16	True
5	False

Answer: (penalty regime: 0 %)

```
1  n = int(input())
2  if n > 0 and (n & (n - 1)) == 0 and (n & 0x55555555) != 0:
3      print("True")
4  else:
5      print("False")
```

	Input	Expected	Got	
✔	16	True	True	✔
✔	5	False	False	✔
✔	1	True	True	✔
✔	-1	False	False	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **3**

Correct

Mark 1.00 out of 1.00

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 | nums[i] - nums[j] | = k and i < j.

Constraints:

$1 \leq n \leq 10^5$

$-10^4 \leq nums[i] \leq 10^4$

$0 \leq k \leq 10^4$

For example:

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

Answer: (penalty regime: 0 %)

```
 1  x=int(input())
 2  y=input().split()
 3  y=[int(i) for i in y]
 4  z=int(input())
 5  s=0
 6  for i in range(x):
 7      for j in range(x):
 8          if(i<j):
 9              if(abs(y[i]-y[j])==z):
10                  s+=1
11  print(s)
```

	Input	Expected	Got	
✔	4 1 2 3 4 1	3	3	✔
✔	5 1 3 1 5 4 0	1	1	✔
✔	4 1 2 2 1 1	4	4	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **4**

Correct

Mark 1.00 out of 1.00

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

Answer: (penalty regime: 0 %)

```python
1  import math
2
3  def calculate_tiles():
4      vals = input().split()
5      a = float(vals[0])
6      b = float(vals[1])
7      c = a * 100
8      d = math.pi * (c / 2) ** 2
9      s = b ** 2
10     if int(a) % 2 != 0:
11         n = math.ceil(d / s) + 100
12     else:
13         n = math.ceil(d / s)
14     print(n, "tiles")
15 calculate_tiles()
16
17
```

	Input	Expected	Got	
✔	10 20	1964 tiles	1964 tiles	✔
✔	10 30	873 tiles	873 tiles	✔
✔	5 20	591 tiles	591 tiles	✔
✔	20 20	7854 tiles	7854 tiles	✔
✔	2 10	315 tiles	315 tiles	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **5**

Correct

Mark 1.00 out of 1.00

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≤X≤1000 — Raghu's shop can hold between 1 and 1000 shoes.

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

1≤N≤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

Answer: (penalty regime: 0 %)

```
1  x = int(input())
2  y = [int(i) for i in input().split()]
```

```
 3  z = int(input())
 4
 5  m = []
 6  for _ in range(z):
 7      b = input().split()
 8      m.append(b)
 9  k = 0
10
11  for i in m:
12      for j in range(len(i)):
13          for l in y:
14              if int(i[j]) == l:
15                  k += int(i[j+1])
16                  y.remove(l)
17                  break
18  print(k)
19
```

	Input	Expected	Got	
✔	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	200	✔
✔	5 5 5 5 5 5 5 5 10 5 10 5 10 5 10 5 10	50	50	✔
✔	4 4 4 6 6 5 4 25 4 25 6 30 6 55 6 55	135	135	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ Week12_MCQ

Jump to...