WEEK 12:

1. Rose manages a personal library with a diverse collection of books. To streamline her library management, she needs a program that can categorize books based on their genres, making it easier to find and organize her collection.

Problem Statement:

Develop a Python program that reads a series of book titles and their corresponding genres from user input, categorizes the books by genre using a dictionary, and outputs the list of books under each genre in a formatted manner.

Input Format:

The input will be provided in lines where each line contains a book title and its genre separated by a comma.

Input terminates with a blank line.

Output Format:

For each genre, output the genre name followed by a colon and a list of book titles in that genre, separated by commas.

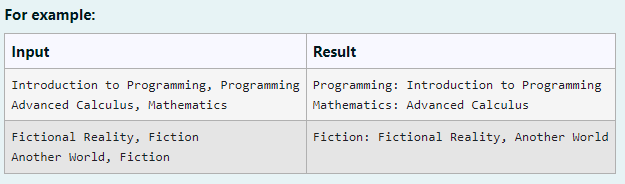
Constraints:

Book titles and genres are strings.

Book titles can vary in length but will not exceed 100 characters.

Genres will not exceed 50 characters.

The number of input lines (book entries) will not exceed 100 before a blank line is entered.



**Program:**

d = {}

while True:

try:

book = input().split(',')

if len(book) < 2:

continue

book\_name = book [0].strip()

category = book[1].strip()

if category in d:

d[category].append(book\_name)

else:

d[category] = [book\_name]

except EOFError:

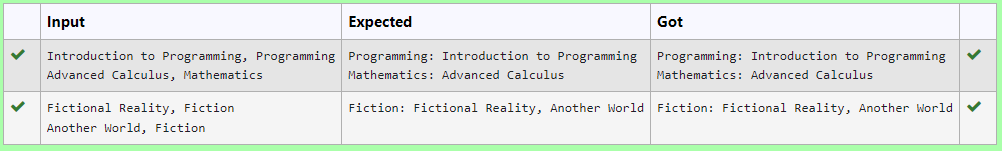
break

for k, v in d.items():

print(f"{k}: ", end='')

print(', '.join(v))

**Output:**

****

2. 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:

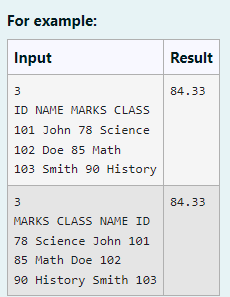
A single line containing the average marks, corrected to two decimal places.

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.



**Program:**

def calculate\_average\_marks (N, columns, student\_data):

total\_marks = 0

num\_students = 0

marks\_index=columns.index('MARKS')

for student in student\_data:

marks = int(student [marks\_index])

total\_marks += marks

num\_students += 1

average\_marks=total\_marks / num\_students if num\_students > 0 else 0

return average\_marks

N=int(input())

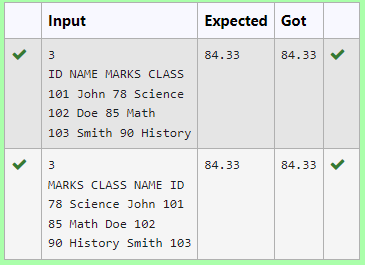
columns=input().split()

student\_data = [input().split() for \_ in range(N)]

average\_marks=calculate\_average\_marks (N, columns, student\_data)

print("{:.2f}".format(average\_marks))

**Output:**

****

3. 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 == 3x.

**Program:**

n=int(input())

c=0

for i in range(100):

if pow(3,i)==n:

c+=1

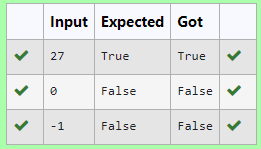
if c!=0:

print("True")

else:

print("False")

**Output:**

****

4. 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.

**Program:**

def count\_unique\_pairs (n, activities,k):

unique\_pairs = set ()

for i in range(n):

for j in range (i + 1, n) :

if abs(activities[i]-activities[j])==k:

unique\_pairs.add((i,j))

return len(unique\_pairs)

def main():

n=int(input())

activities=list(map(int, input().split()))

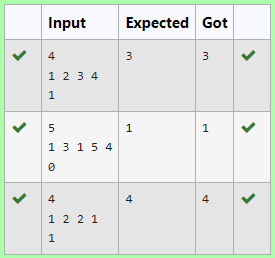
k=int(input())

result=count\_unique\_pairs(n, activities,k)

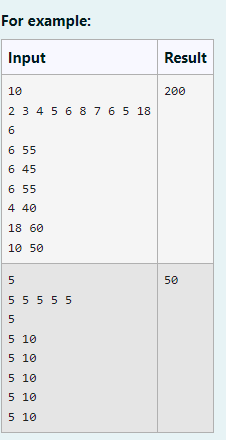
print(result)

main()

**Output:**

****

5. 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.



**Program:**

def calculate\_total\_revenue():

from collections import Counter

X = int(input().strip())

shoe\_sizes = list(map(int, input().strip().split()))

N = int(input().strip())

shoe\_inventory = Counter(shoe\_sizes)

total\_revenue = 0

for \_ in range(N):

size, price = map(int, input().strip().split())

if shoe\_inventory[size] > 0:

total\_revenue += price

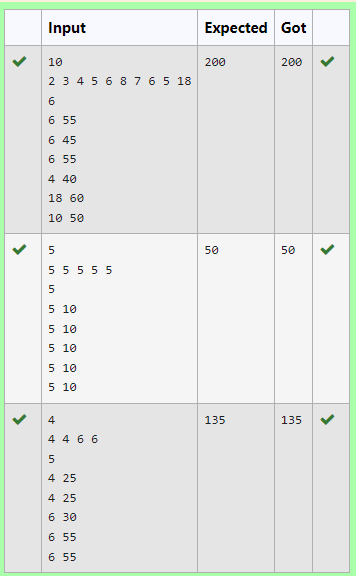
shoe\_inventory[size] -= 1

print(total\_revenue)

if \_\_name\_\_ == "\_\_main\_\_":

calculate\_total\_revenue()

**Output:**

****