Rajalakshmi Engineering College

Name: Sudharsan R
Email: 240701543@rajalakshmi.edu.in
Roll no: 240701543
Phone: 9176060959
Branch: REC
Department: I CSE FF
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 5_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Alex is tasked with managing the membership lists of several exclusive clubs. Each club has its own list of members, and Alex needs to determine the unique members who are part of exactly one club when considering all clubs together.

Your goal is to help Alex by writing a program that calculates the symmetric difference of membership lists from multiple clubs and then finds the total number of unique members.

Input Format

The first line of input consists of an integer k, representing the number of clubs.

The next k lines each contain a space-separated list of integers, where each

integer represents a member's ID.

Output Format

The first line of output displays the symmetric difference of the membership lists as a set.

The second line displays the sum of the elements in this symmetric difference.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 3
1 2 3
2 3 4
5 6 7

Output: {1, 4, 5, 6, 7}
23

Answer

```
# You are using Python
k = int(input())
clubs = []

for _ in range(k):
    members = set(map(int, input().split()))
    clubs.append(members)

# Calculate symmetric difference across all sets
# Initialize with the first club's members
sym_diff = clubs[0]

for i in range(1, k):
    sym_diff = sym_diff.symmetric_difference(clubs[i])

# Print output as per the format
print("{" + ", ".join(map(str, sorted(sym_diff))) + "}")
print(sum(sym_diff))
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Riya owns a store and keeps track of item prices from two different suppliers using two separate dictionaries. He wants to compare these prices to identify any differences. Your task is to write a program that calculates the absolute difference in prices for items that are present in both dictionaries. For items that are unique to one dictionary (i.e., not present in the other), include them in the output dictionary with their original prices.

Help Riya to implement the above task using a dictionary.

Input Format

The first line of input consists of an integer n1, representing the number of items in the first dictionary.

The next n1 lines contain two integers

1. The first line contains the item (key), and
2. The second line contains the price (value).

The following line consists of an integer n2, representing the number of items in the second dictionary

The next n2 lines contain two integers

1. The first line contains the item (key), and
2. The second line contains the price (value).

Output Format

The output should display a dictionary that includes:

1. For items common to both dictionaries, the absolute difference between their prices.
2. For items that are unique to one dictionary, the original price from that dictionary.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1
4
4
1
8
7

Output: {4: 4, 8: 7}

Answer

```
# You are using Python
def solve():
    # Read the number of items for the first dictionary
    n1 = int(input())

    # Initialize the first dictionary
    dict1 = {}
    # Populate the first dictionary with n1 items
    for _ in range(n1):
        key = int(input())
        value = int(input())
        dict1[key] = value

    # Read the number of items for the second dictionary
    n2 = int(input())

    # Initialize the second dictionary
    dict2 = {}
    # Populate the second dictionary with n2 items
    for _ in range(n2):
        key = int(input())
        value = int(input())
        dict2[key] = value

    # Initialize the result dictionary
    result_dict = {}

    # Iterate through items in the first dictionary
    for item, price1 in dict1.items():
        # Check if the item is also present in the second dictionary
        if item in dict2:
```

```
            # If common, calculate the absolute difference in prices
            price2 = dict2[item]
            result_dict[item] = abs(price1 - price2)
        else:
            # If unique to the first dictionary, add its original price
            result_dict[item] = price1

    # Iterate through items in the second dictionary
    for item, price2 in dict2.items():
        # Check if the item is unique to the second dictionary (not already
processed from dict1)
        if item not in dict1:
            # If unique to the second dictionary, add its original price
            result_dict[item] = price2

    # Print the resulting dictionary
    print(result_dict)

# Call the solve function to run the program
solve()
```

Status : Correct **Marks : 10/10**

3. Problem Statement

Samantha is working on a text analysis tool that compares two words to find common and unique letters. She wants a program that reads two words, w1, and w2, and performs the following operations:

Print the letters common to both words, in alphabetical order.Print the letters that are unique to each word, in alphabetical order.Determine if the set of letters in the first word is a superset of the letters in the second word.Check if there are no common letters between the two words and print the result as a Boolean value.

Ensure the program ignores case differences and leading/trailing spaces in the input words.

Your task is to help Samantha in implementing the same.

Input Format

The first line of input consists of a string representing the first word, w1.

The second line consists of a string representing the second word, w2.

Output Format

The first line of output should display the sorted letters common to both words, printed as a list.

The second line should display the sorted letters that are unique to each word, printed as a list.

The third line should display a Boolean value indicating if the set of letters in w1 is a superset of the set of letters in w2.

The fourth line should display a Boolean value indicating if there are no common letters between w1 and w2.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: program
Peace
Output: ['a', 'p']
['c', 'e', 'g', 'm', 'o', 'r']
False
False

Answer

```
# You are using Python
# Read input words, strip spaces and convert to lowercase
w1 = input().strip().lower()
w2 = input().strip().lower()

# Convert to sets of letters
set_w1 = set(w1)
set_w2 = set(w2)
```

```
# 1. Letters common to both words, sorted
common_letters = sorted(set_w1.intersection(set_w2))

# 2. Letters unique to each word, sorted
unique_letters = sorted(set_w1.symmetric_difference(set_w2))

# 3. Check if set of letters in w1 is a superset of set_w2
is_superset = set_w1.issuperset(set_w2)

# 4. Check if no common letters exist
no_common = len(common_letters) == 0

# Print the results as specified
print(common_letters)
print(unique_letters)
print(is_superset)
print(no_common)
```

Status : Correct ***Marks : 10/10***

4. Problem Statement

Alex is working with grayscale pixel intensities from an old photo that has been scanned in a single row. To detect edges in the image, Alex needs to calculate the differences between each pair of consecutive pixel intensities.

Your task is to write a program that performs this calculation and returns the result as a tuple of differences.

Input Format

The first line of input contains an integer n, representing the number of pixel intensities.

The second line contains n space-separated integers representing the pixel intensities.

Output Format

The output displays a tuple containing the absolute differences between consecutive pixel intensities.

Refer to the sample output for format specifications.

Sample Test Case

Input: 5
200 100 20 80 10
Output: (100, 80, 60, 70)

Answer

```
# You are using Python
def solve():
    # Read the number of pixel intensities
    n = int(input())

    # Read the space-separated pixel intensities and convert them to a list of
integers
    pixel_intensities = list(map(int, input().split()))

    # Initialize an empty list to store the differences
    differences = []

    # Iterate from the first pixel to the second-to-last pixel
    # to calculate differences with the next pixel
    for i in range(n - 1):
        # Calculate the absolute difference between the current pixel and the next
one
        diff = abs(pixel_intensities[i + 1] - pixel_intensities[i])
        differences.append(diff)

    # Convert the list of differences to a tuple and print it
    print(tuple(differences))

# Call the solve function to execute the program
solve()
```

Status : Correct ***Marks : 10/10***