Rajalakshmi Engineering College

Name: Nivedhitha K
Email: 240701371@rajalakshmi.edu.in
Roll no: 240701371
Phone: 9790413580
Branch: REC
Department: I CSE FD
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

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 and clean the input
w1 = input().strip().lower()
w2 = input().strip().lower()

# Convert to sets of unique characters
set1 = set(w1)
set2 = set(w2)
```

```
# Common letters
common = sorted(set1 & set2)

# Unique letters to each word
unique = sorted((set1 ^ set2))

# Is w1 a superset of w2?
is_superset = set1.issuperset(set2)

# Are there no common letters?
no_common = set1.isdisjoint(set2)

# Output
print(common)
print(unique)
print(is_superset)
print(no_common)
```

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

2. Problem Statement

Noah, a global analyst at a demographic research firm, has been tasked with identifying which country experienced the largest population growth over a two-year period. He has a dataset where each entry consists of a country code and its population figures for two consecutive years. Noah needs to determine which country had the highest increase in population and present the result in a specific format.

Help Noah by writing a program that outputs the country code with the largest population increase, along with the increase itself.

Input Format

The first line of input consists of an integer N, representing the number of countries.

Each of the following N blocks contains three lines:

1. The first line is a country code.
2. The second line is an integer representing the population of the country in the

first year.
3. The third line is an integer representing the population of the country in the second year.

Output Format

The output displays the country code and the population increase in the format {code: difference}, where code is the country code and difference is the increase in population.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
01
1000
1500
02
2000
2430
03
1500
3000

Output: {03:1500}

Answer

```
# You are using Python
# Read number of countries
N = int(input())

# Variables to track the maximum growth and corresponding country code
max_increase = -1
max_country_code = ""

# Read and process each country's data
for _ in range(N):
    code = input().strip()
    pop_year1 = int(input())
    pop_year2 = int(input())
```

```
    increase = pop_year2 - pop_year1

    if increase > max_increase:
        max_increase = increase
        max_country_code = code

# Output in the required format
print(f"{{{max_country_code}:{max_increase}}}")
```

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

3. 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
# Read number of pixel intensities
n = int(input())

# Read the pixel intensities
pixels = list(map(int, input().split()))

# Compute absolute differences between consecutive pixels
differences = tuple(abs(pixels[i] - pixels[i + 1]) for i in range(n - 1))

# Print the result
print(differences)
```

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

4. Problem Statement

Riley is analyzing DNA sequences and needs to determine which bases match at the same positions in two given DNA sequences. Each DNA sequence is represented as a tuple of integers, where each integer corresponds to a DNA base.

Your task is to write a program that compares these two sequences and identifies the bases that match at the same positions and print it.

Input Format

The first line of input consists of an integer n, representing the size of the first tuple.

The second line contains n space-separated integers, representing the elements of the first DNA sequence tuple.

The third line of input consists of an integer m, representing the size of the second tuple.

The fourth line contains m space-separated integers, representing the elements of the second DNA sequence tuple.

Output Format

The output is a space-separated integer of the matching bases at the same positions in both sequences.

Refer to the sample output for format specifications.

Sample Test Case

Input: 4
5 1 8 4
4
4 1 8 2
Output: 1 8

Answer

```
# You are using Python
# Read the size and elements of the first sequence
n = int(input())
seq1 = tuple(map(int, input().split()))

# Read the size and elements of the second sequence
m = int(input())
seq2 = tuple(map(int, input().split()))

# Find matching elements at the same positions
matching_bases = [str(seq1[i]) for i in range(min(n, m)) if seq1[i] == seq2[i]]

# Print the result as space-separated integers
print(" ".join(matching_bases))
```

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