Rajalakshmi Engineering College

Name: krithika narasimhan
Email: 240701277@rajalakshmi.edu.in
Roll no: 240701277
Phone: 9677451731
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 5_CY

Attempt : 2
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. 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

```
n1 = int(input())
dict1 = {}
for _ in range(n1):
    key = int(input())
```

```
    value = int(input())
    dict1[key] = value
n2 = int(input())
dict2 = {}
for _ in range(n2):
    key = int(input())
    value = int(input())
    dict2[key] = value
result = {}
for key in dict1:
    if key in dict2:
        result[key] = abs(dict1[key] - dict2[key])
    else:
        result[key] = dict1[key]

for key in dict2:
    if key not in result:
        result[key] = dict2[key]
print(result)
```

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

2. Problem Statement

Emily is a librarian who keeps track of books borrowed and returned by her patrons. She maintains four sets of book IDs: the first set represents books borrowed, the second set represents books returned, the third set represents books added to the collection, and the fourth set represents books that are now missing. Emily wants to determine which books are still borrowed but not returned, as well as those that were added but are now missing. Finally, she needs to find all unique book IDs from both results.

Help Emily by writing a program that performs the following operations on four sets of integers:

Compute the difference between the borrowed books (first set) and the returned books (second set).Compute the difference between the added books (third set) and the missing books (fourth set).Find the union of the results from the previous two steps, and sort the final result in descending

order.

Input Format

The first line of input consists of a list of integers representing borrowed books.

The second line of input consists of a list of integers representing returned books.

The third line of input consists of a list of integers representing added books.

The fourth line of input consists of a list of integers representing missing books.

Output Format

The first line of output displays the difference between sets P and Q, sorted in descending order.

The second line of output displays the difference between sets R and S, sorted in descending order.

The third line of output displays the union of the differences from the previous two steps, sorted in descending order.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1 2 3
2 3 4
5 6 7
6 7 8

Output: [1]
[5]
[5, 1]

Answer

```
# Read input sets
P = set(map(int, input().split()))
Q = set(map(int, input().split()))
R = set(map(int, input().split()))
S = set(map(int, input().split()))
```

```
# Compute differences
diff1 = P - Q
diff2 = R - S

# Compute union of differences
union_diff = diff1.union(diff2)

# Output results in descending order
print(f"{sorted(diff1, reverse=True)}")
print(f"{sorted(diff2, reverse=True)}")
print(f"{sorted(union_diff, reverse=True)}")
```

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

3. Problem Statement

James is an engineer working on designing a new rocket propulsion system. He needs to solve a quadratic equation to determine the optimal launch trajectory. The equation is of the form ax2 +bx+c=0.

Your task is to help James find the roots of this quadratic equation. Depending on the discriminant, the roots might be real and distinct, real and equal, or complex. Implement a program to determine and display the roots of the equation based on the given coefficients.

Input Format

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

The second line contains three space-separated integers a,b, and c representing the coefficients of the quadratic equation.

Output Format

The output displays:

1. If the discriminant is positive, display the two real roots.
2. If the discriminant is zero, display the repeated real root.
3. If the discriminant is negative, display the complex roots as a tuple with real and imaginary parts.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
1 5 6
Output: (-2.0, -3.0)

Answer

```
# Read the size of the tuple (this will be 3 for the quadratic equation case)
tuple_size = int(input().strip())

# Read the coefficients as a tuple
coefficients = tuple(map(int, input().split()))

# Unpack the tuple into a, b, and c
a, b, c = coefficients

# Calculate the discriminant
discriminant = b * b - 4 * a * c

# Determine the roots
if discriminant > 0:
    # Compute square root approximation without using functions
    sqrt_discriminant = discriminant ** 0.5
    root1 = (-b + sqrt_discriminant) / (2 * a)
    root2 = (-b - sqrt_discriminant) / (2 * a)
    roots = (root1, root2)
elif discriminant == 0:
    root = -b / (2 * a)
    roots = (root, root)
else:
    # Calculate the square root approximation for negative discriminant
    temp = -discriminant
    sqrt_temp = temp ** 0.5
    realPart = -b / (2 * a)
    imaginaryPart = sqrt_temp / (2 * a)
    roots = ((realPart, imaginaryPart), (realPart, -imaginaryPart))
```

```
# Output
print(roots)
```

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

```
size1 = int(input().strip())

dna1 = tuple(map(int, input().strip().split()))


size2 = int(input().strip())

dna2 = tuple(map(int, input().strip().split()))


matching_bases = []

if size1 == size2:

    for i in range(size1):
        if dna1[i] == dna2[i]:
            matching_bases.append(dna1[i])

# Print the result
if len(matching_bases) == 1:
    print(matching_bases[0])
elif len(matching_bases) > 1:
    print(' '.join(map(str, matching_bases)))
```

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