Rajalakshmi Engineering College

Name: Tanisha Sudhagar
Email: 240701553@rajalakshmi.edu.in
Roll no: 2116240701553
Phone: 6374776137
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())

# Initialize the symmetric difference with the first set
sets = [set(map(int, input().split())) for _ in range(k)]
symmetric_diff = sets[0]

# Compute the symmetric difference across all sets
for s in sets[1:]:
    symmetric_diff ^= s

# Calculate the sum of the elements in the symmetric difference
sum_of_diff = sum(symmetric_diff)

# Display the results
print(symmetric_diff)
print(sum_of_diff)
```

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


N = int(input())

# Initialize a dictionary to store the population data
population_data = {}

# Process each country's data
for _ in range(N):
    # Read country code
    country_code = input()
    # Read populations for two years
    year1_population = int(input())
    year2_population = int(input())

    # Calculate the increase in population
    increase = year2_population - year1_population

    # Store the increase in the dictionary
    population_data[country_code] = increase

# Find the country code with the largest increase
max_increase_country = max(population_data, key=population_data.get)
max_increase_value = population_data[max_increase_country]

# Print the country code with the largest increase in population in the format
{code: difference}
print(f"{{{max_increase_country}:{max_increase_value}}}")
```

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

```
# You are using Python
```

```
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

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 compare_prices():
    try:
        # Read n1, the number of items for the first dictionary
        n1 = int(input())
        dict1 = {}
        # Loop n1 times to get item-price pairs for supplier 1
        for _ in range(n1):
            key = int(input())
            value = int(input())
            dict1[key] = value

        # Read n2, the number of items for the second dictionary
        n2 = int(input())
        dict2 = {}
        # Loop n2 times to get item-price pairs for supplier 2
        for _ in range(n2):
            key = int(input())
            value = int(input())
            dict2[key] = value

        result_dict = {}

        # Iterate through items in the first dictionary
        for item, price1 in dict1.items():
            # If item is also in the second dictionary, calculate absolute difference
            if item in dict2:
                price2 = dict2[item]
                result_dict[item] = abs(price1 - price2)
            # Otherwise, item is unique to dict1, add its original price
            else:
                result_dict[item] = price1

        # Iterate through items in the second dictionary
        for item, price2 in dict2.items():
            # If item has not already been added to result_dict (meaning it's unique to
dict2)
```

```
            if item not in result_dict:
                # Add its original price
                result_dict[item] = price2

        print(result_dict)

    except ValueError:
        print("Invalid input. Please ensure you enter integers for item keys and 
prices.")
    except Exception as e:
        print(f"An unexpected error occurred: {e}")

compare_prices()
```

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