

Rajalakshmi Engineering College

Name: Rena J

Email: 241801227@rajalakshmi.edu.in

Roll no: 241801227

Phone: 9941271176

Branch: REC

Department: I AI & DS FC

Batch: 2028

Degree: B.E - AI & DS

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

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 $ax^2 + bx + c = 0$.

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

```
import math
```

```
n=int(input())
```

```
coeff=list(map(int,input().split()))
```

```
a,b,c=coeff
```

```
discriminant=b**2 -(4*a*c)
```

```
if discriminant>0:
```

```
    root1=(-b-math.sqrt(discriminant))/(2*a)
```

```
    root2=(-b+math.sqrt(discriminant))/(2*a)
```

```
    print(f"({root2}, {root1})")
```

```
elif discriminant==0:
```

```
    root=-b/(2*a)
```

```
    print(f"({root},{root})")
```

```
else:
```

```
    real=-b/(2*a)
```

```
    img=math.sqrt(abs(discriminant))/(2*a)
```

```
    root1=(real,img)
```

```
    root2=(real,-img)
```

```
    print(f"({root1}, {root2})")
```

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 n_1 , representing the number of items in the first dictionary.

The next n_1 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 n_2 , representing the number of items in the second dictionary

The next n_2 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 dictionary(n):
```

```
    d={}
```

```
    for _ in range(n):
```

```
        key=int(input())
```

```
        value=int(input())
```

```
        d[key]=value
```

```
    return d
```

```
n1=int(input())
```

```
dict1=dictionary(n1)
```

```
n2=int(input())
```

```
dict2=dictionary(n2)
```

```
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 dict1:
```

```
        result[key]=dict2[key]
```

```
print(result)
```

Status : Correct

Marks : 10/10

3. 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=list(map(int,input().split()))
    clubs.append(set(members))

from collections import defaultdict
freq=defaultdict(int)
for club in clubs:
    for member in club:
        freq[member]+=1
sym_diff={member for member , count in freq.items() if count==1}
sum_=sum(sym_diff)
formatted_set="{", ".join(map(str,sorted(sym_diff)))+"}"
print(formatted_set)
print(sum_)

```

Status : Correct

Marks : 10/10

4. 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

You are using Python

```
borrow=set(map(int,input().split()))
```

```
returned=set(map(int,input().split()))
```

```
added=set(map(int,input().split()))
```

```
missing=set(map(int,input().split()))
```

```
diff1=borrow-returned
```

```
sorted1=sorted(diff1,reverse=True)
```

```
diff2=added-missing
```

```
sorted2=sorted(diff2,reverse=True)
```

```
final=sorted(diff1.union(diff2),reverse=True)
```

```
def format_op(s):  
    if not s:  
        return ""  
    return "[" + ",".join(map(str,s)) + "]"  
print(format_op(sorted1))  
print(format_op(sorted2))  
print(format_op(final))
```

Status : Correct

Marks : 10/10