Rajalakshmi Engineering College

Name: Haripreeth CJ
Email: 241501065@rajalakshmi.edu.in
Roll no: 241501065
Phone: 9445359004
Branch: REC
Department: I AI & ML FA
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 5_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 37.5

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 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
import cmath

N = int(input())
a, b, c = map(int, input().split())

if a == 0:
    # Handle as a linear equation: bx + c = 0
    if b != 0:
        root = -c / b
        print((round(root, 1),))
    else:
        # No solution or infinite solutions (0x + c = 0)
        print(("No valid equation",))
else:
    D = b**2 - 4*a*c
    if D > 0:
        root1 = (-b -( D**0.5)) / (2 * a)
        root2 = (-b + (D**0.5)) / (2 * a)
        roots = tuple([round(root2, 1), round(root1, 1)])
```

```
    print(roots)
elif D == 0:
    root = -b / (2 * a)
    print((round(root, 1),))
else:
    root1 = (-b + cmath.sqrt(D)) / (2 * a)
    root2 = (-b - cmath.sqrt(D)) / (2 * a)
    complex_root1 = (round(root1.real, 1), round(root1.imag, 1))
    complex_root2 = (round(root2.real, 1), round(root2.imag, 1))
    print((complex_root1, complex_root2))
```

Status : Partially correct ***Marks : 7.5/10***

2. 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())
unique= set()
duplicates = set()
for i in range(k):
    club = set(map(int,input().split()))
    for m in club:
        if m in unique:
          unique.remove(m)
          duplicates.add(m)
        elif m not in duplicates:
          unique.add(m)
print(unique)
print(sum(unique))
```

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

3. 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
pr  =  set(map(int,input().split()))
rt =  set(map(int,input().split()))
ad =  set(map(int,input().split()))
ms =  set(map(int,input().split()))
sbr = sorted(pr-rt,reverse = True)
abm  = sorted(ad-ms,reverse = True)
final = sorted(set(sbr).union(abm),reverse = True)
print(sbr)
print(abm)
print(final)
```

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
# Input: size of the first DNA sequence
n = int(input())
# Input: elements of the first DNA sequence
seq1 = tuple(map(int, input().split()))

# Input: size of the second DNA sequence
m = int(input())
# Input: elements of the second DNA sequence
seq2 = tuple(map(int, input().split()))

# Find the minimum length to avoid index errors
min_len = min(n, m)

# Collect matching bases at the same positions
matching_bases = []
for i in range(min_len):
    if seq1[i] == seq2[i]:
        matching_bases.append(seq1[i])

# Output the result as space-separated values
print(" ".join(map(str, matching_bases)))
```

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