Rajalakshmi Engineering College

Name: ZARAA JUHI SAAI K
Email: 240701044@rajalakshmi.edu.in
Roll no: 240701044
Phone: 9444513456
Branch: REC
Department: I CSE FA
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 5_MCQ

Attempt : 1
Total Mark : 20
Marks Obtained : 19

Section 1 : MCQ

1. What is the output of the following code?

```
a=(1,2,(4,5))
b=(1,2,(3,4))
print(a<b)
```

Answer

False

Status : Correct ***Marks : 1/1***

2. If 'a' is a dictionary with some key-value pairs, what does a.popitem() do?

Answer

Removes an arbitrary element

Status : Correct ***Marks : 1/1***

3. What is the output of the below Python code?

```
list1 = [1, 2, 3]
list2 = [5, 6, 7]
list3 = [10, 11, 12]
set1 = set(list2)
set2 = set(list1)
set1.update(set2)
set1.update(list3)
print(set1)
```

Answer

{1, 2, 3, 5, 6, 7, 10, 11, 12}

Status : Correct ***Marks : 1/1***

4. What is the output of the following?

```
set1 = {10, 20, 30, 40, 50}
set2 = {60, 70, 10, 30, 40, 80, 20, 50}
print(set1.issubset(set2))
print(set2.issuperset(set1))
```

Answer

TrueTrue

Status : Correct ***Marks : 1/1***

5. What is the output of the following code?

```
a={"a":1,"b":2,"c":3}
b=dict(zip(a.values(),a.keys()))
print(b)
```

Answer

{1: ‘a’, 2: ‘b’, 3: ‘c’}

Status : Correct ***Marks : 1/1***

6. Which of the following is a Python tuple?

Answer

(1, 2, 3)

Status : Correct ***Marks : 1/1***

7. What will be the output for the following code?

```
t1 = (1, 2, 4, 3)
t2 = (1, 2, 3, 4)
print(t1 < t2)
```

Answer

False

Status : Correct ***Marks : 1/1***

8. Which of the statements about dictionary values is false?

Answer

Values of a dictionary must be unique

Status : Correct ***Marks : 1/1***

9. Set s1 = {1, 2, 4, 3} and s2 = {1, 5, 4, 6}, find s1 & s2, s1 - s2, s1 | s2 and s1 ^ s2.

Answer

s1&s2 = {1, 4}s1-s2 = {2, 3}s1^s2 = {2, 3, 5, 6}s1|s2 = {1, 2, 3, 4, 5, 6}

Status : Correct ***Marks : 1/1***

10. What will be the output?

```
a={'B':5,'A':9,'C':7}
print(sorted(a))
```

Answer

['A', 'B', 'C'].

Status : Correct ***Marks : 1/1***

11. Suppose t = (1, 2, 4, 3), which of the following is incorrect?

Answer

t[3] = 45

Status : Correct ***Marks : 1/1***

12. What is the output of the following code?

```
a={1:"A",2:"B",3:"C"}
b=a.copy()
b[2]="D"
print(a)
```

Answer

{1: 'A', 2: 'B', 3: 'C'}

Status : Correct ***Marks : 1/1***

13. What is the result of print(type({}) is set)?

Answer

False

Status : Correct ***Marks : 1/1***

14. Fill in the code in order to get the following output.

Output:

Tuple: (1, 3, 4)

Max value: 4

t=(1,)

_____

print("Tuple:" ,t)
print("Max value:",_____)

Answer

1) t=t+(3,4)2) max(t)

Status : Correct ***Marks : 1/1***

15. What will be the output of the following code?

a=(1,2,3,4)
print(sum(a,3))

Answer

13

Status : Correct ***Marks : 1/1***

16. Which of the following statements is used to create an empty tuple?

Answer

( )

Status : Correct ***Marks : 1/1***

17. What will be the output of the following program?

set1 = {1, 2, 3}
set2 = set1.copy()
set2.add(4)
print(set1)

Answer

{1, 2, 3}

Status : Correct ***Marks : 1/1***

18. What will be the output for the following code?

```
a=(1,2,3)
b=('A','B','C')
c=zip(a,b)

print(c)
print(tuple(c))
```

Answer

((1, 'A'), (2, 'B'), (3, 'C'))

Status : Correct ***Marks : 1/1***

19. Predict the output of the following Python program

```
init_tuple_a = 1, 2, 8
init_tuple_b = (1, 2, 7)
set1=set(init_tuple_b)
set2=set(init_tuple_a)
print (set1 | set2)
print (init_tuple_a | init_tuple_b)
```

Answer

TypeError: unsupported operand type

Status : Wrong ***Marks : 0/1***

20. Which of the following isn't true about dictionary keys?

Answer

Keys must be integers

Status : Correct ***Marks : 1/1***

Rajalakshmi Engineering College

Name: ZARAA JUHI SAAI K
Email: 240701044@rajalakshmi.edu.in
Roll no: 240701044
Phone: 9444513456
Branch: REC
Department: I CSE FA
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 5_COD

Attempt : 1
Total Mark : 50
Marks Obtained : 50

Section 1 : Coding

1. Problem Statement

Liam is analyzing a list of product IDs from a recent sales report. He needs to determine how frequently each product ID appears and calculate the following metrics:

Frequency of each product ID: A dictionary where the key is the product ID and the value is the number of times it appears.Total number of unique product IDs.Average frequency of product IDs: The average count of all product IDs.

Write a program to read the product IDs, compute these metrics, and output the results.

Example

Input:

6 //number of product ID

101

102

101

103

101

102 //product IDs

Output:

{101: 3, 102: 2, 103: 1}

Total Unique IDs: 3

Average Frequency: 2.00

Explanation:

Input 6 indicates that you will enter 6 product IDs.

A dictionary is created to track the frequency of each product ID.

Input 101: Added with a frequency of 1.

Input 102: Added with a frequency of 1.

Input 101: Frequency of 101 increased to 2.

Input 103: Added with a frequency of 1.

Input 101: Frequency of 101 increased to 3.

Input 102: Frequency of 102 increased to 2.

The dictionary now contains 3 unique IDs: 101, 102, and 103.

Total Unique is 3.

The average frequency is 2.00.

Input Format

240701044

The first line of input consists of an integer n, representing the number of product IDs.

The next n lines each contain a single integer, each representing a product ID.

Output Format

The first line of output displays the frequency dictionary, which maps each product ID to its count.

The second line displays the total number of unique product IDs, preceded by "Total Unique IDs: ".

The third line displays the average frequency of the product IDs. This is calculated by dividing the total number of occurrences of all product IDs by the total number of unique product IDs, rounded to two decimal places. It is preceded by "Average Frequency: ".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 6
101
102
101
103
101
102

Output: {101: 3, 102: 2, 103: 1}
Total Unique IDs: 3
Average Frequency: 2.00

Answer

```
d={}
l1=[]
l2=s=0
a=0.0
n=int(input())
for i in range (0,n):
    e=int(input())
```

240701044

```
    l1.append(e)
for i in l1:
    if i not in d:
        d1=dict([(i,l1.count(i))])
        d.update(d1)
print(d)
l2=len(d)
print(f"Total Unique IDs:{l2}")
s=sum(d.values())
a=s/l2
print(f"Average frequency:{a:.2f}")
```

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

2. Problem Statement

Ella is analyzing the sales data for a new online shopping platform. She has a record of customer transactions where each customer's data includes their ID and a list of amounts spent on different items. Ella needs to determine the total amount spent by each customer and identify the highest single expenditure for each customer.

Your task is to write a program that computes these details and displays them in a dictionary.

Input Format

The first line of input consists of an integer n, representing the number of customers.

Each of the next n lines contains a numerical customer ID followed by integers representing the amounts spent on different items.

Output Format

The output displays a dictionary where the keys are customer IDs and the values are lists containing two integers: the total expenditure and the maximum single expenditure.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 2
101 100 150 200
102 50 75 100

Output: {101: [450, 200], 102: [225, 100]}

Answer

```
# You are using Python
n=int(input())
d={}
for i in range(0,n):
    s=input().split()
    l=list(map(int,s))
    key=int(l[0])
    l.remove(key)
    tot=sum(l)
    m=max(l)
    d[key]=[tot,m]
print(d)
```

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

3. Problem Statement

Gowshik is working on a task that involves taking two lists of integers as input, finding the element-wise sum of the corresponding elements, and then creating a tuple containing the sum values.

Write a program to help Gowshik with this task.

Example:

Given list:

[1, 2, 3, 4]

[3, 5, 2, 1]

An element-wise sum of the said tuples: (4, 7, 5, 5)

Input Format

The first line of input consists of a single integer n, representing the length of the input lists.

The second line of input consists of n integers separated by commas, representing the elements of the first list.

The third line of input consists of n integers separated by commas, representing the elements of the second list.

Output Format

The output is a single line containing a tuple of integers separated by commas, representing the element-wise sum of the corresponding elements from the two input lists.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 4
1, 2, 3, 4
3, 5, 2, 1
Output: (4, 7, 5, 5)

Answer

```
# You are using Python
n=int(input())
s1=tuple(map(int,input().split(',')))
s2=tuple(map(int,input().split(',')))
s3=tuple(s1[i]+s2[i] for i in range(n))
print(s3)
```

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

4. Problem Statement

Professor Adams needs to analyze student participation in three recent academic workshops. She has three sets of student IDs: the first set contains students who registered for the workshops, the second set contains students who actually attended, and the third set contains students who dropped out.

Professor Adams needs to determine which students who registered also attended, and then identify which of these students did not drop out.

Help Professor Adams identify the students who registered, attended, and did not drop out of the workshops.

Input Format

The first line of input consists of integers, representing the student IDs who registered for the workshops.

The second line consists of integers, representing the student IDs who attended the workshops.

The third line consists of integers, representing the student IDs who dropped out of the workshops.

Output Format

The first line of output displays the intersection of the first two sets, which shows the IDs of students who registered and attended.

The second line displays the result after removing student IDs that are in the third set (dropped out), showing the IDs of students who both attended and did not drop out.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1 2 3
2 3 4
3 4 5
Output: {2, 3}

240701044

{2}

Answer

```
# You are using Python
r=input().split()
s1=set(map(int,r))
a=input().split()
s2=set(map(int,a))
d=input().split()
s3=set(map(int,d))
s4=s1&s2
s5=s4-s3
print(s4)
print(s5)
```

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

5. Problem Statement

James is managing a list of inventory items in a warehouse. Each item is recorded as a tuple, where the first element is the item ID and the second element is a list of quantities available for that item. James needs to filter out all quantities that are above a certain threshold to find items that have a stock level above this limit.

Help James by writing a program to process these tuples, filter the quantities from all the available items, and display the results.

Note:

Use the filter() function to filter out the quantities greater than the specified threshold for each item's stock list.

Input Format

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

The next N lines each contain a tuple in the format (ID, [quantity1, quantity2, ...]), where ID is an integer and the list contains integers.

The final line consists of an integer threshold, representing the quantity

threshold.

Output Format

The output should be a single line displaying the filtered quantities, space-separated. Each quantity is strictly greater than the given threshold.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 2
(1, [1, 2])
(2, [3, 4])
2
Output: 3 4

Answer

```
# You are using Python
n=int(input())
data=[]
for i in range(n):
    tup=eval(input())
    data.append(tup)
k=int(input())
result=[]
for item in data:
    q=item[1]
    f=filter(lambda x:x>k,q)
    result.extend(f)
print(*result)
```

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

240701044

Rajalakshmi Engineering College

Name: ZARAA JUHI SAAI K
Email: 240701044@rajalakshmi.edu.in
Roll no: 240701044
Phone: 9444513456
Branch: REC
Department: I CSE FA
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 5_PAH

Attempt : 1
Total Mark : 60
Marks Obtained : 60

Section 1 : Coding

1. Problem Statement

Maya wants to create a dictionary that maps each integer from 1 to a given number n to its square. She will use this dictionary to quickly reference the square of any number up to n.

Help Maya generate this dictionary based on the input she provides.

Input Format

The input consists of an integer n, representing the highest number for which Maya wants to calculate the square.

Output Format

The output displays the generated dictionary where each key is an integer from 1 to n, and the corresponding value is its square.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5

Output: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

Answer

```
# You are using Python
def generate_square_dict(n):
    """Generate a dictionary mapping integers to their squares."""
    square_dict = {i: i**2 for i in range(1, n + 1)}
    return square_dict

def main():
    # Read the input number
    n = int(input().strip())

    # Generate the dictionary of squares
    square_dict = generate_square_dict(n)

    # Print the resulting dictionary
    print(square_dict)

if __name__ == "__main__":
    main()
```

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

2. Problem Statement

Mia is organizing a list of integers into a series of pairs for his new project. She wants to create pairs of consecutive integers from the list. The last integer should be paired with None to complete the series. The pairing happens as follows: ((Element 1, Element 2), (Element 2, Element 3)..... (Element n, None)).

Your task is to help Henry by writing a Python program that reads a list of

integers, forms these pairs, and displays the result in tuple format.

Input Format

The first line of input consists of an integer n, representing the number of elements in the tuple.

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

Output Format

The output displays a tuple containing pairs of consecutive integers from the input. The last integer in the tuple is paired with 'None'.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
5 10 15
Output: ((5, 10), (10, 15), (15, None))

Answer

```
# You are using Python
def create_pairs(n, elements):
    pairs = []

    for i in range(n - 1):
        pairs.append((elements[i], elements[i + 1]))

    # Pair the last element with None
    pairs.append((elements[-1], None))

    return tuple(pairs)

def main():
    # Read the number of elements
    n = int(input().strip())

    # Read the space-separated integers
```

```
    elements = list(map(int, input().strip().split()))

    # Create pairs
    result = create_pairs(n, elements)

    # Print the result
    print(result)

if __name__ == "__main__":
    main()
```

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

3. Problem Statement

Jordan is creating a program to process a list of integers. The program should take a list of integers as input, remove any duplicate integers while preserving their original order, concatenate the remaining unique integers into a single string, and then print the result.

Help Jordan in implementing the same.

Input Format

The input consists of space-separated integers representing the elements of the set.

Output Format

The output prints a single integer formed by concatenating the unique integers from the input in the order they appeared.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 11 11 33 50
Output: 113350

Answer

```
# You are using Python
def process_integers(input_string):
    # Split the input string into a list of integers
    integers = input_string.split()

    # Use a list to preserve the order of unique integers
    unique_integers = []
    seen = set()

    for num in integers:
        if num not in seen:
            seen.add(num)
            unique_integers.append(num)

    # Concatenate the unique integers into a single string
    result = ''.join(unique_integers)

    return result

def main():
    # Read the input as a space-separated string
    input_string = input().strip()

    # Process the integers and get the result
    result = process_integers(input_string)

    # Print the result
    print(result)

if __name__ == "__main__":
    main()
```

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

4. Problem Statement

Sophia is organizing a list of event IDs representing consecutive days of an event. She needs to group these IDs into consecutive sequences. For example, if the IDs 3, 4, and 5 appear consecutively, they should be grouped.

Write a program that helps Sophia by reading the total number of event IDs and the IDs themselves, then display each group of consecutive IDs in tuple format.

Input Format

The first line of input consists of an integer n, representing the number of event IDs.

The next n lines contain integers representing the event IDs, where each integer corresponds to an event ID.

Output Format

The output should display each group of consecutive event IDs in a tuple format. Each group should be printed on a new line, and single event IDs should be displayed as a single-element tuple.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
1
2
3
Output: (1, 2, 3)

Answer

```
def group_consecutive_ids(event_ids):
    if not event_ids:
        return []

    # Sort the event IDs to ensure they are in order
    event_ids.sort()

    # Initialize the list to hold groups of consecutive IDs
    grouped_ids = []
    current_group = [event_ids[0]]
```

```
    for i in range(1, len(event_ids)):
        # Check if the current ID is consecutive
        if event_ids[i] == event_ids[i - 1] + 1:
            current_group.append(event_ids[i])
        else:
            # If not consecutive, save the current group and start a new one
            grouped_ids.append(tuple(current_group))
            current_group = [event_ids[i]]

    # Don't forget to add the last group
    grouped_ids.append(tuple(current_group))

    return grouped_ids

def main():
    n = int(input().strip())
    event_ids = [int(input().strip()) for _ in range(n)]
    result = group_consecutive_ids(event_ids)

    for group in result:
        if len(group) == 1:
            # Print single element group as (element) without comma
            print(f"({group[0]})")
        else:
            print(group)

if __name__ == "__main__":
    main()
```

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

5. Problem Statement

Tom wants to create a dictionary that lists the first n prime numbers, where each key represents the position of the prime number, and the value is the prime number itself.

Help Tom generate this dictionary based on the input she provides.

Input Format

The input consists of an integer n, representing the number of prime numbers Tom wants to generate.

Output Format

The output displays the generated dictionary where each key is an integer from 1 to n, and the corresponding value is the prime number.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
Output: {1: 2, 2: 3, 3: 5, 4: 7}

Answer

```
# You are using Python
def is_prime(num):
    """Check if a number is prime."""
    if num <= 1:
        return False
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            return False
    return True

def generate_primes(n):
    """Generate a dictionary of the first n prime numbers."""
    primes = {}
    count = 0
    num = 2  # Start checking for prime numbers from 2

    while count < n:
        if is_prime(num):
            count += 1
            primes[count] = num
        num += 1

    return primes
```

```
def main():
    # Read the number of prime numbers to generate
    n = int(input().strip())

    # Generate the dictionary of prime numbers
    prime_dict = generate_primes(n)

    # Print the resulting dictionary
    print(prime_dict)

if __name__ == "__main__":
    main()
```

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

6. Problem Statement

Rishi is working on a program to manipulate a set of integers. The program should allow users to perform the following operations:

Find the maximum value in the set.Find the minimum value in the set.Remove a specific number from the set.

The program should handle these operations based on user input. If the user inputs an invalid operation choice, the program should indicate that the choice is invalid.

Input Format

The first line contains space-separated integers that will form the initial set. Each integer x is separated by a space.

The second line contains an integer ch, representing the user's choice:

- 1 to find the maximum value
- 2 to find the minimum value
- 3 to remove a specific number from the set

If ch is 3, the third line contains an integer n1, which is the number to be removed from the set.

Output Format

The first line of output prints the original set in descending order.

For choice 1: Print the maximum value from the set.

For choice 2: Print the minimum value from the set.

For choice 3: Print the set after removing the specified number, in descending order.

For invalid choices: Print "Invalid choice".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1 2 3 4 5
1
Output: {5, 4, 3, 2, 1}
5

Answer

```
# You are using Python

def main():
    # Read the initial set of integers
    initial_set = list(map(int, input().strip().split()))

    # Sort the set in descending order
    sorted_set = sorted(initial_set, reverse=True)

    # Print the original set
    print(f"{{{', '.join(map(str, sorted_set))}}}")

    # Read the user's choice
    choice = int(input().strip())

    if choice == 1:
        # Find the maximum value
```

```
        max_value = max(sorted_set)
        print(max_value)

    elif choice == 2:
        # Find the minimum value
        min_value = min(sorted_set)
        print(min_value)

    elif choice == 3:
        # Remove a specific number from the set
        number_to_remove = int(input().strip())
        if number_to_remove in sorted_set:
            sorted_set.remove(number_to_remove)
        # Print the set after removal
        print(f"{{{', '.join(map(str, sorted(sorted_set, reverse=True)))}}}")

    else:
        # Invalid choice
        print("Invalid choice")

if __name__ == "__main__":
    main()
```

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

240701044

Rajalakshmi Engineering College

Name: ZARAA JUHI SAAI K
Email: 240701044@rajalakshmi.edu.in
Roll no: 240701044
Phone: 9444513456
Branch: REC
Department: I CSE FA
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 : 36.5

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

def find_roots(a, b, c):
    # Calculate the discriminant
    discriminant = b**2 - 4*a*c

    if discriminant > 0:
        # Two distinct real roots
        root1 = (-b + discriminant**0.5) / (2*a)
        root2 = (-b - discriminant**0.5) / (2*a)
        return (root1, root2)
    elif discriminant == 0:
        # One repeated real root
        root = -b / (2*a)
        return (root,)
    else:
        # Complex roots
        real_part = -b / (2*a)
        imaginary_part = (abs(discriminant)**0.5) / (2*a)
```

```
        root1 = (real_part, imaginary_part)
        root2 = (real_part, -imaginary_part)
        return (root1, root2)

# Input reading
N = int(input())
coefficients = list(map(int, input().split()))
a, b, c = coefficients

# Finding roots
roots = find_roots(a, b, c)

# Output formatting
if len(roots) == 1:
    print(roots[0])
else:
    print(roots)
```

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

2. 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

```
def analyze_words(w1, w2):
    # Normalize the input by stripping spaces and converting to lowercase
    w1 = w1.strip().lower()
    w2 = w2.strip().lower()

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

    # Find common letters
    common_letters = sorted(set1.intersection(set2))

    # Find unique letters combined and sort
```

```
    unique_letters = sorted((set1 - set2).union(set2 - set1))

    # Check if set1 is a superset of set2
    is_superset = set1.issuperset(set2)

    # Check if there are no common letters
    no_common_letters = len(common_letters) == 0

    # Print results
    print(common_letters)
    print(unique_letters)
    print(is_superset)
    print(no_common_letters)

# Input reading
w1 = input()
w2 = input()

analyze_words(w1, w2)
```

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
def symmetric_difference_of_clubs(k, club_memberships):
    # Initialize the set for the symmetric difference
    symmetric_diff = set()

    # Iterate through each club's membership
    for members in club_memberships:
        current_set = set(members)
        # Update the symmetric difference
        symmetric_diff = symmetric_diff.symmetric_difference(current_set)

    return symmetric_diff

# Input reading
k = int(input())
club_memberships = []

for _ in range(k):
    members = list(map(int, input().split()))
    club_memberships.append(members)
```

240701044

```
# Calculate the symmetric difference
result = symmetric_difference_of_clubs(k, club_memberships)

# Output the results
print(result)
print(sum(result))
```

Status : Partially correct **Marks : 9/10**

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
def find_matching_bases(seq1, seq2):
    # Find the minimum length to avoid index errors
    min_length = min(len(seq1), len(seq2))

    # Collect matching bases
    matching_bases = []
    for i in range(min_length):
        if seq1[i] == seq2[i]:
            matching_bases.append(seq1[i])

    return matching_bases

# Input reading
n = int(input())
seq1 = tuple(map(int, input().split()))
m = int(input())
seq2 = tuple(map(int, input().split()))

# Find matching bases
result = find_matching_bases(seq1, seq2)

# Output the results
print(" ".join(map(str, result)))
```

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

240701044