Rajalakshmi Engineering College

Name: Manju sri N
Email: 241801151@rajalakshmi.edu.in
Roll no: 241801151
Phone: 8946059431
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_PAH

Attempt : 1
Total Mark : 60
Marks Obtained : 46

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

```
n = int(input())
squares_dict = {i: i ** 2 for i in range(1, n + 1)}
print(squares_dict)
```

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

```
n = int(input())
elements = list(map(int, input().split()))
pairs = tuple((elements[i], elements[i + 1] if i + 1 < n else None) for i in range(n))


print(pairs)
```

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

3. 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

```
# You are using Python
# Read the number of event IDs
n = int(input())

# Read the event IDs into a list
event_ids = [int(input()) for _ in range(n)]

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

# Iterate through the event IDs to form consecutive groups
for i in range(1, n):
    if event_ids[i] == event_ids[i - 1] + 1:
        current_group.append(event_ids[i])
    else:
        groups.append(tuple(current_group))  # Append the current group as a tuple
        current_group = [event_ids[i]]  # Start a new group

# Append the last group to the result
groups.append(tuple(current_group))

# Print each group on a new line
for group in groups:
    print(group)
```

Status : Partially correct ***Marks : 6/10***

4. 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
# Read the input as a space-separated string of integers
input_list = input().split()

# Remove duplicates while preserving order using a set
unique_list = []
seen = set()

for num in input_list:
    if num not in seen:
        unique_list.append(num)
        seen.add(num)

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

# Print the result
print(result)
```

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
# Function to check if a number is prime
def is_prime(num):
    if num < 2:
        return False
    for i in range(2, int(num ** 0.5) + 1):
        if num % i == 0:
            return False
    return True

# Function to generate the first n primes
def generate_primes(n):
    primes = {}
    num = 2  # Start checking from the number 2
    count = 1  # Counter for prime numbers

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

    return primes

# Read the input value n
n = int(input())

# Generate the dictionary of first n primes
prime_dict = generate_primes(n)

# Print the result
print(prime_dict)
```

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

```
# Function to handle the user's choice and perform the required operation
```

```
def manipulate_set(nums, choice, num_to_remove=None):
    # Sort the set in descending order
    sorted_set = sorted(nums, reverse=True)

    # Print the original set in descending order
    print(f"{set(sorted_set)}")

    # Perform the operation based on the user's choice
    if choice == 1:
        # Find the maximum value in the set
        print(max(nums))
    elif choice == 2:
        # Find the minimum value in the set
        print(min(nums))
    elif choice == 3:
        # Remove the specific number from the set
        if num_to_remove in nums:
            nums.remove(num_to_remove)
            # Print the set after removal in descending order
            print(f"[set(sorted(nums,reverse=True))]")
        else:
            print("Number not found in the set")
    else:
        print("Invalid choice")

# Read the initial set of integers
nums = set(map(int, input().split()))

# Read the choice of operation
choice = int(input())

if choice == 3:
    # If choice is 3, we also need to take the number to remove
    num_to_remove = int(input())
    manipulate_set(nums, choice, num_to_remove)
else:
    # If choice is 1 or 2, we don't need a number to remove
    manipulate_set(nums, choice)
```

Status : Wrong ***Marks : 0/10***