-Sourav(24/48044)

1. Write a program that takes two numbers, A and B. Find numbers between A and B where each digit is an even number. How many such numbers are there?

PROGRAM

```
def find_even_digit_numbers(start, end):
    even_numbers = []  # List to store numbers with all even digits
    for num in range(start, end + 1):  # Loop through the range
        all_even = True  # Flag to check if all digits are even
        for digit in str(num):  # Check each digit
            if int(digit) % 2 != 0:  # If any digit is odd
                all_even = False
                break
        if all_even:  # If all digits are even
            even_numbers.append(num)
    return even_numbers

# Input from user
A = int(input("Enter the starting number (A): "))
B = int(input("Enter the ending number (B): "))

# Find numbers with all even digits
result = find_even_digit_numbers(A, B)

# Output the results
print(f"Numbers with all even digits: {result}")
print(f"Total count of such numbers: {len(result)}")
```

OUTPUT

```
================ RESTART: D:\Python\Assignment\tempassignment.py ===============
Enter the starting number (A): 33
Enter the ending number (B): 52
Numbers with all even digits: [40, 42, 44, 46, 48]
Total count of such numbers: 5
```

2. Write a program that accepts a positive number and subtracts the sum of its digits from this number. Repeat this process until the number becomes negative or zero, and print the number of such operations required to make the resulting number non-positive

PROGRAM

-Sourav(24/48044)

```
def count_operations(num):
    if num <= 0:
        return "Please enter a positive number"

    digit_sum = 0  # Initialize sum of digits
    for digit in str(num):  # Calculate the sum of digits
        digit_sum += int(digit)

    count = 0  # Counter for the number of operations
    while num > 0:
        num -= digit_sum  # Subtract the sum of digits from the number
        count += 1  # Increment the count

    return count

# Input from the user
num = int(input("Enter a positive number: "))

# Call the function and display the result
result = count_operations(num)
if isinstance(result, int):
    print(f"Total number of operations required: {result}")
else:
    print(result)
```

OUTPUT

```
================ RESTART: D:\Python\Assignment\tempassignment.py ===============
Enter a positive number: 22
Total number of operations required: 6
```

3. Write a program to replace all subsequent occurrences of the first character of a given string with another given character (#).

Sample input: jumping jack.

Expected output: jumping #ack.

PROGRAM

-Sourav(24/48044)

```
def replacing(string):
    if len(string) == 0:  # Check if the string is empty
        return "Please provide a valid string"

    first_char = string[0]  # Identify the first character
    new_string = first_char  # Initialize new string with the first character

    for char in string[1:]:  # Loop through the rest of the string
        if char == first_char:  # If the character matches the first character
            new_string += "#"  # Replace with '#'
        else:
            new_string += char  # Otherwise, keep the original character

    return new_string

# Input from the user
string = input("Enter the string value: ")
result = replacing(string)
print(result)
```

OUTPUT

```
================ RESTART: D:\Python\Assignment\tempassignment.py ===============
Enter the string value: Jumping Jack
Jumping #ack
```

4. Write a program to get the sum of comma-separated numbers. If the sum is more than 100, print 0. Else, print the actual sum.

PROGRAM

```
def num_splitsum(num):
    num_list = num.split(",")  # Split the input string by commas
    total_sum = 0  # Initialize the sum

    for item in num_list:  # Loop through the list of numbers
        total_sum += int(item)  # Convert to integer and add to the sum

    if total_sum >= 100:  # Check if the sum exceeds or equals 100
        return 0
    else:
        return total_sum

# Input from the user
string = input("Enter the comma-separated numbers: ")

# Calculate result
result_split = num_splitsum(string)

# Output the result
print("Output using split method:", result_split)
```

-Sourav(24/48044)

OUTPUT

```
================ RESTART: D:\Python\Assignment\tempassignment.py ===============
Enter the comma-separated numbers: 23,55,3,12
Output using split method: 93
```

5. Given a list of integers with duplicate elements in it. The task is to generate another list that contains only duplicate elements. The new list should include elements that appear more than once.

Example 1: - Input: 10, 20, 30, 20, 20, 30, 40, 50, -20, 60, 60, -20, -20

Output: [20, 30, -20, 60]

Example 2: - Input: -1, 1, -1, 8

Output: -1

PROGRAM

```
def lst_modify(lst):
    repeated_elements = []  # List to store duplicate elements

    for i in lst:
        if lst.count(i) > 1 and i not in repeated_elements:  # Check if the element appears more than once and is not already in repeated_elements
            repeated_elements.append(i)  # Add it to the list of repeated elements

    return repeated_elements

# Input from the user
given_lst = eval(input("Enter a list of integers (comma-separated): "))

# Call the function and display the result
result = lst_modify(given_lst)
print("Required Output:", result)
```

OUTPUT

```
================ RESTART: D:\Python\Assignment\tempassignment.py ===============
Enter a list of integers (comma-separated): [10, 20, 30, 20, 20, 30, 40, 50, -20, 60, 60, -20, -20]
Required Output: [20, 30, -20, 60]
```

6. Write a function intreverse(n) that takes as input a positive integer n and returns the integer obtained by reversing the digits in n.

PROGRAM

-Sourav(24/48044)

```
def interverse_slicing(n):
    reverse = str(n)[::-1]  # Convert the integer to a string and reverse it
    return int(reverse)  # Convert the reversed string back to an integer

# Input from the user
num = int(input("Enter a number: "))

# Call the slicing method
result_slicing = interverse_slicing(num)

# Output the result
print("Required output:", result_slicing)
```

OUTPUT

```
================ RESTART: D:\Python\Assignment\tempassignment.py ===============
Enter a number: 1234587
Required output: 7854321
```

7. Write a function matched(s) that takes as input a string s and checks if the brackets "(" and ")" in s are matched: that is, every "(" has a matching ")" after it and every ")" has a matching "(" before it. Your function should ignore all other symbols that appear in s. Your function should return True if s has matched brackets and False if it does not.

PROGRAM

```
def matched(s):
    count = 0  # Counter to track the unmatched opening parentheses
    for i in s:
        if i == "(":  # If it's an opening parenthesis, increment the count
            count += 1
        elif i == ")":  # If it's a closing parenthesis, decrement the count
            count -= 1
            if count < 0:  # If count becomes negative, there are unmatched closing parentheses
                return False
    return count == 0  # If the count is zero, all parentheses are matched

# Input from the user
string = input("Enter a string with brackets: ")

# Call the function and print the result
result = matched(string)
print(result)
```

OUTPUT

```
================ RESTART: D:\Python\Assignment\tempassignment.py ===============
Enter a string with brackets: ()(())
True

================ RESTART: D:\Python\Assignment\tempassignment.py ===============
Enter a string with brackets: ())
False
```