

ASSIGNMENT-2

Roll no:2303A52074
Batch-37

Task 1: Cleaning Sensor Data

Expected Output:

Before cleaning: [25, -3, 18, -7, 0, 42, -1]

After cleaning: [25, 18, 0, 42]

Prompt:

Write a Python function to remove negative numbers from a list.
Print the list before and after cleaning.

Code

```
[1] ✓ 1s def remove_negative_numbers(input_list):  
    """  
    Removes negative numbers from a list.  
    Args:  
        input_list: A list of numbers.  
    Returns:  
        A new list with negative numbers removed.  
    """  
    return [num for num in input_list if num >= 0]  
  
# Example usage:  
my_list = [1, -2, 3, -4, 5, 0, -10, 8]  
  
print("Original list:", my_list)  
  
cleaned_list = remove_negative_numbers(my_list)  
  
print("List after removing negative numbers:", cleaned_list)  
  
... Original list: [1, -2, 3, -4, 5, 0, -10, 8]  
List after removing negative numbers: [1, 3, 5, 0, 8]
```

Output

```
• Original list: [1, -2, 3, -4, 5, 0, -10, 8]  
List after removing negative numbers: [1, 3, 5, 0, 8]
```

Explanation

The function checks the sensor data values one by one using a loop.
If a value is negative, it is ignored because it is not valid.
Only zero and positive values are kept in a new list.

Task 2: String Character Analysis

Expected Output:

Input:Hello123world

Vowels: 3

Consonants: 7

Digits: 3

Prompt:

Write a Python function that takes a string as user input and counts the number of vowels, consonants, and digits. Print the results.

Code

```
def count_chars(input_string):
    vowels = "aeiouAEIOU"
    vowel_count = 0
    consonant_count = 0
    digit_count = 0

    for char in input_string:
        if char.isalpha(): # Check if it's an alphabet
            if char in vowels:
                vowel_count += 1
            else:
                consonant_count += 1
        elif char.isdigit(): # Check if it's a digit
            digit_count += 1

    return vowel_count, consonant_count, digit_count

# Example usage with user input:
my_string = input("Enter a string: ")
v, c, d = count_chars(my_string)

print(f"Original string: {my_string}")
print(f"Number of vowels: {v}")
print(f"Number of consonants: {c}")
print(f"Number of digits: {d}")
```

Output

```
... Enter a string: SRU2074
Original string: SRU2074
Number of vowels: 1
Number of consonants: 2
Number of digits: 4
```

Explanation

The program accepts a string as input of a user. It checks every character to determine whether it is a vowel, consonant or a digit.

Lastly, it shows the number of vowels, consonants and digits.

Task 3: Palindrome Check – Tool Comparison

Expected Output

Input:Madam
Palindrome

Prompt

Write a Python function that takes a string as user input and checks whether it is a palindrome. Print whether it is Palindrome or Not a Palindrome.

Code

```
def is_palindrome(input_string):  
    # Convert the string to lowercase and remove spaces for case-insensitive and space-agnostic check  
    cleaned_string = "".join(char.lower() for char in input_string if char.isalnum())  
  
    # Check if the cleaned string is equal to its reverse  
    return cleaned_string == cleaned_string[::-1]  
  
# Get user input  
user_input = input("Enter a string: ")  
  
# Check if it's a palindrome and print the result  
if is_palindrome(user_input):  
    print(f'{user_input} is a Palindrome.')  
else:  
    print(f'{user_input} is Not a Palindrome.')
```

Output

```
•• Enter a string: IRON MAN  
   'IRON MAN' is Not a Palindrome.
```

Explanation

The program receives a user input in the form of a string. It switches the string and makes a comparison to the original one. In case they are identical it will then print Palindrome but if not it will print Not a Palindrome.

Task 4: Code Explanation Using AI

Expected Output

Input:7

Prime

Prompt

Write a Python program to check whether a number is prime. Then explain the code line by line.

Code

```
def is_prime(number):
    # Prime numbers are greater than 1
    if number <= 1:
        return False
    # 2 is the only even prime number
    if number == 2:
        return True
    # All other even numbers are not prime
    if number % 2 == 0:
        return False
    # Check for divisibility from 3 up to the square root of the number,
    # incrementing by 2 to check only odd divisors
    i = 3
    while i * i <= number:
        if number % i == 0:
            return False
        i += 2
    return True

# Get user input for the number to check
num = int(input("Enter an integer: "))

# Check if the number is prime and print the result
if is_prime(num):
    print(f"{num} is a prime number.")
else:
    print(f"{num} is not a prime number.")
```

Output

```
• Enter an integer: 22
  22 is not a prime number.
```

```
        return False
        i += 2
    return True

# Get user input for the number to check
num = int(input("Enter an integer: "))

# Check if the number is prime and print the result
if is_prime(num):
    print(f"{num} is a prime number.")
else:
    print(f"{num} is not a prime number.")

... Enter an integer: 22
  22 is not a prime number.
```

a detailed line-by-line explanation:

The `is_prime` function:

1. `def is_prime(number):`
 - This line defines a function named `is_prime` that takes one argument, `number`. This function's primary purpose is to check whether the provided `number` is prime and return `True` or `False` accordingly.
2. `if number <= 1:`
 - By definition, prime numbers are natural numbers greater than 1. Therefore, if the input `number` is 1 or any value less than 1 (like 0, negative numbers), it cannot be considered prime. This line acts as an initial check for such cases.
3. `return False`
 - If the condition `number <= 1` is true, the function immediately returns `False`, signifying that the number is not prime.
4. `if number == 2:`

Explanation The program takes a number as input. It checks if the number is divisible by any other number. Based on this, it prints whether the number is prime or not.