

## LAB 2.2

Name: G.Sriram

HallTicket : 2303A51191

Date: 13-Jan-2026

### Task 1: Cleaning Sensor Data

#### PROMPT:

K Generate a Python function that removes all negative numbers from a list of integers and returns the cleaned list.

◆ Sure, here's a Python function that removes all negative numbers from a list of integers and returns the cleaned list. I've also included an example of how to use it.

[Run an example usage of the function](#)

[Explain the 'remove\\_negative\\_numbers' function](#)

What can I help you build?



Gemini 2.5 Flash ▾ ➤

## CODE:

```
] Os
▶ def remove_negative_numbers(numbers):
    """
    Removes all negative numbers from a list of integers.

    Args:
        numbers: A list of integers.

    Returns:
        A new list containing only the non-negative numbers from the input list.
    """
    return [num for num in numbers if num >= 0]

    # Example Usage
my_list = [1, -2, 3, -4, 0, 5, -6]
cleaned_list = remove_negative_numbers(my_list)
print(f"Original list: {my_list}")
print(f"Cleaned list (without negative numbers): {cleaned_list}")

...
Original list: [1, -2, 3, -4, 0, 5, -6]
Cleaned list (without negative numbers): [1, 3, 0, 5]
```

## OUTPUT:

Original list: [1, -2, 3, -4, 0, 5, -6]

Cleaned list (without negative numbers): [1, 3, 0, 5]

---

## Task 2: String Character Analysis

### PROMPT:

- K Generate a Python function to count vowels, consonants, and digits in a given string.
- ◆ Certainly! Here's a Python function to count vowels, consonants, and digits in a given string, along with an example of how to use it:

► Accept and run    ✓ Accept    X Cancel

### CODE:

```
def count_char_types(input_string):
    vowels = "aeiouAEIOU"
    digits = "0123456789"
    vowel_count = 0
    consonant_count = 0
    digit_count = 0

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

    return {"vowels": vowel_count, "consonants": consonant_count, "digits": digit_count}

my_string = "Hello World 123!"
counts = count_char_types(my_string)
print(f"Original string: '{my_string}'")
print(f"Vowel count: {counts['vowels']}")  
print(f"Consonant count: {counts['consonants']}")  
print(f"Digit count: {counts['digits']}")
```

### OUTPUT:

Original string: 'Hello World 123!'

Vowel count: 3

Consonant count: 7

Digit count: 3

---

## Task 3: Palindrome Check – Tool Comparison

### PROMPT:

K Generate a Python function to check whether a string is a palindrome.

◆ Sure! Here's a Python function to check if a string is a palindrome, along with an example of how to use it.

· Working...

What can I help you build?

+ Gemini 2.5 Flash ▾

### CODE: GEMINI GENERATED

```
def is_palindrome(s):
    # Convert to lowercase and remove non-alphanumeric characters
    cleaned_s = ''.join(char.lower() for char in s if char.isalnum())

    # Compare the cleaned string with its reverse
    return cleaned_s == cleaned_s[::-1]

print(f"'madam' is a palindrome: {is_palindrome('madam')}")  
print(f"'A man, a plan, a canal: Panama' is a palindrome: {is_palindrome('A  
man, a plan, a canal: Panama')}")  
print(f"'hello' is a palindrome: {is_palindrome('hello')}")  
print(f"'Racecar' is a palindrome: {is_palindrome('Racecar')}")  
print(f"'No lemon, no melon' is a palindrome: {is_palindrome('No lemon, no  
melon')}")
```

### OUTPUT:

```
'madam' is a palindrome: True
'A man, a plan, a canal: Panama' is a palindrome: True
'hello' is a palindrome: False
'Racecar' is a palindrome: True
'No lemon, no melon' is a palindrome: True
```

#### CODE :(COPILOT GENERATED):

```
s   def is_palindrome(s):
        s = s.lower().replace(" ", "") # normalize: lowercase and remove spaces
        return s == s[::-1]

        # Example usage:
        print(is_palindrome("madam"))      # True
        print(is_palindrome("racecar"))    # True
        print(is_palindrome("hello"))      # False
        print(is_palindrome("A man a plan a canal Panama")) # True

    ... True
    True
    False
    True
```

#### SIDE BY SIDE OBSERVATION TABLE :

Tool	Code Clarity	Simplicity	Readability
Gemini	High	Very simple	Easy to understand
Copilot	Medium	Slightly complex	More control

#### ❖ Student Observation:

Gemini generated more concise and beginner-friendly code, while Copilot provided a detailed implementation with better input handling.

## Task 4: Code Explanation Using AI

CODE:

```
def find_max_difference(nums):
    if len(nums) < 2:
        return 0

    min_val = nums[0]
    max_diff = 0

    for i in range(1, len(nums)):
        if nums[i] < min_val:
            min_val = nums[i]
        else:
            max_diff = max(max_diff, nums[i] - min_val)

    return max_diff
```

PROMPT:



Explain the following Python function line by line in simple terms.  
def find\_max\_difference(nums):  
if len(nums) < 2:  
return 0  
min\_val =  
nums[0]  
max\_diff = 0  
for i in range(1, len(nums)):  
if nums[i] < min\_val:  
min\_val = nums[i]  
else:  
max\_diff = max(max\_diff, nums[i] - min\_val)  
return max\_diff

## AI EXPLANATION:

Let's go through the `find_max_difference` function line by line:

1. **`def find_max_difference(nums):`**

- This line defines a new function called `find_max_difference`. A function is a block of organized, reusable code that performs a single, related action.
- `nums` is the input to this function. It's expected to be a list of numbers.

2. **`if len(nums) < 2:`**

- This is a check to see if the `nums` list has fewer than 2 elements.
- If a list has less than two numbers, you can't find a "difference" between two *different* numbers where the second number appears after the first, which is what this function is designed to do (finding the maximum difference between `nums[j] - nums[i]` where  $j > i$ ).

3. **`return 0`**

- If the list has less than 2 elements (as checked in the previous line), the function immediately stops and returns 0 because no meaningful difference can be calculated.

4. **`min_val = nums[0]`**

- We initialize a variable called `min_val` with the value of the first number in the `nums` list.
- This `min_val` will keep track of the smallest number encountered *so far* as we go through the list.

5. **`max_diff = 0`**

- We initialize a variable called `max_diff` to 0.
- This variable will store the largest difference we find between any number and the `min_val` encountered *before* it.

6. **`for i in range(1, len(nums)):`**

- This line starts a loop that will go through each number in the `nums` list, starting from the *second* number (at index 1) up to the end of the list.
- We start from the second number because we've already used the first number to initialize `min_val`.

7. **`if nums[i] < min_val:`**

- Inside the loop, for each number `nums[i]`, this line checks if the current number is smaller than the `min_val` we've tracked so far.

8. **`min_val = nums[i]`**

- If the current number `nums[i]` is smaller than `min_val`, it means we've found a new, smaller minimum value. So, we update `min_val` to this new smaller number.

**`max_diff = max(max_diff, nums[i] - min_val)`**

- In this case, `nums[i]` is a number that came *after* `min_val` (or is the same as `min_val`). We calculate the difference between the current number `nums[i]` and our `min_val`.
- Then, we compare this new difference with the `max_diff` we've found so far. The `max()` function picks the larger of the two values.
- `max_diff` is then updated to store the largest difference found up to this point.

9. **`return max_diff`**

- After the loop has finished checking all numbers in the list, this line returns the final `max_diff` value, which represents the largest difference found between any two numbers `nums[j] - nums[i]` where  $j > i$ .

### Student Understanding / Comments:

Initially, I found the logic confusing. After Gemini's explanation, I understood that the function finds the maximum difference between two elements such that the smaller number appears before the larger number. The explanation helped me understand how minimum tracking works.