SR UNIVERSITY

School of Computer Science and Artificial Intelligence

Lab Assignment # 4.2

Program	**: B. Tech (CSE)**
Specialization	**:**
Course Title	**: AI Assisted coding**
Course Code	**:**
Semester	**II**
Academic Session	**: 2025-2026 Name of**
Student	**: Kattherapaka Dhanush**
Enrollment No.	**: 2403A51L32**
Batch No.	**52**
Date	**:20-01-2026**

Task Description-1

- Zero-shot: Prompt AI with only the instruction. Write a Python function to determine
- whether a given number is prime

```
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n ** 0.5) + 1):
        if n % i == 0:
            return False
    return True
n=int(input())
print(is_prime(n))
```

```
7
True
```

Step-by-step explanation:

1. **Function Definition**
   - def is_prime(n):
   - This function takes an integer n as input and checks whether it is a prime number.
2. **Check for invalid prime numbers**
   - if n <= 1:

- Numbers less than or equal to 1 are **not prime**, so the function returns False.

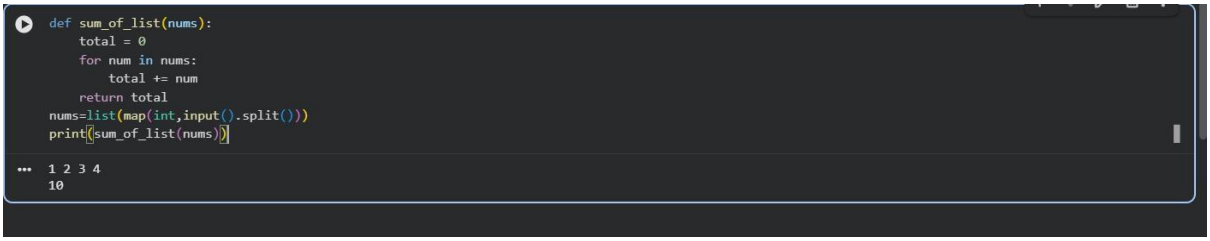
3. **Optimized loop for checking factors**
   - for i in range(2, int(n ** 0.5) + 1):
   - A number only needs to be checked for divisibility up to its **square root**.
   - If n has a factor greater than its square root, it must also have a smaller factor.
4. **Divisibility test**
   - if n % i == 0:
   - If n is divisible by any number i in the loop, it is **not prime**, so return False.
5. **Prime confirmation**
   - return True
   - If no divisors are found, the number is **prime**.

Example:

- is_prime(7) → True (7 has no divisors other than 1 and itself)
- is_prime(10) → False (10 is divisible by 2)

❖ **Task2:**One-shot:

❖ Provide one example: Input: [1, 2, 3, 4], Output: 10 to help AI generate a function that calculates the sum of elements in a list.



```
def sum_of_list(nums):
    total = 0
    for num in nums:
        total += num
    return total
nums=list(map(int,input().split()))
print(sum_of_list(nums))
```

```
1 2 3 4
10
```

Step-by-step explanation:

1. **Function Definition**
   - def sum_of_list(nums):
   - This function accepts a list of numbers called nums.
2. **Initialize the sum**
   - total = 0
   - A variable total is created to store the running sum of the elements.
3. **Loop through the list**
   - for num in nums:
   - The loop goes through each element (num) in the list.
4. **Add each element**
   - total += num
   - Each number in the list is added to total.

5. **Return the result**

   - return total

   - After all elements are processed, the function returns the final sum.

Example using the one-shot input:

- Input: [1, 2, 3, 4]
- Calculation: 1 + 2 + 3 + 4 = 10
- Output: 10

❖ **Task3:Few-shot:**

❖ Give 2–3 examples to create a function that extracts digits from a alphanumeric string.

```
def extract_digits(s):
    result = ""
    for ch in s:
        if ch.isdigit():
            result += ch
    return result
s=input()
print(extract_digits(s))
```

```
12s34g5h6
123456
```

Explanation:

1. The function takes a string s as input.

2. An empty string result is initialized to store digits.
3. The function loops through each character in the string.
4. isdigit() checks whether the character is a digit (0–9).
5. If it is a digit, it is added to result.
6. Finally, the function returns all extracted digits as a string.

❖ **Task4:**Compare zero-shot vs few-shot prompting for generating a function that counts the number of vowels in a string.

```
def count_vowels(s):
    vowels = "aeiouAEIOU"
    count = 0
    for ch in s:
        if ch in vowels:
            count += 1
    return count
s=input()
print(count_vowels(s))
```

```
vowels
2
```

Step-by-step explanation:

1. **Function definition**
   - def count_vowels(s):
   - Defines a function named count_vowels that takes a string s as input.
2. **Vowel list**
   - vowels = "aeiouAEIOU"
   - Stores all lowercase and uppercase vowels.
   - This ensures the function counts vowels regardless of case.
3. **Initialize counter**

- count = 0
- A variable to keep track of how many vowels are found.

4. **Loop through the string**
   - for ch in s:
   - Iterates over each character (ch) in the input string.
5. **Check for vowels**
   - if ch in vowels:
   - Checks whether the current character is a vowel.
6. **Increase count**
   - count += 1
   - Increments the counter whenever a vowel is found.
7. **Return result**
   - return count
   - Sends back the total number of vowels in the string.
8. **User input**
   - s = input()
   - Takes a string input from the user.

9. **Print output**

- print(count_vowels(s))
- Calls the function and prints the number of vowels

❖ **Task5:**Use few-shot prompting with 3 sample inputs to generate a function that determines the minimum of three numbers without using the built-in min() function.

```
def minimum_of_three(a, b, c):
    min_val = a
    if b < min_val:
        min_val = b
    if c < min_val:
        min_val = c
    return min_val
print(minimum_of_three(3,2,1))
print(minimum_of_three(3,2,4))
print(minimum_of_three(3,5,4))
```

```
1
2
3
```

Explanation:

- The function starts by assuming a is the smallest.
- It compares b with the current minimum and updates if needed.
- It then compares c with the updated minimum.
- Finally, it returns the smallest of the three numbers.