

# Assignment 3.4

Name: Zobiya Fatima

Batch:14

Hall Ticket: 2303A51879

## Task 1: Zero-shot Prompt – Fibonacci Series Generator

### Task Description #1

- Without giving an example, write a single comment prompt

asking GitHub Copilot to generate a Python function to print the first N Fibonacci numbers.

Prompt : “# Write a Python function that takes an integer N and prints the first N Fibonacci numbers in order”

Code and output :

```
1 def print_fibonacci(n):
2     """
3     Print the first N Fibonacci numbers in order (on one line).
4
5     Args:
6         n (int): Number of Fibonacci numbers to print
7     """
8     if n <= 0:
9         print("Please enter a positive number!")
10        return
11
12    fib_numbers = []
13
14    if n >= 1:
15        fib_numbers.append(0)
16    if n >= 2:
17        fib_numbers.append(1)
18
19    # Generate Fibonacci numbers
20    for i in range(2, n):
21        fib_numbers.append(fib_numbers[i-1] + fib_numbers[i-2])
22
23    print(" ".join(map(str, fib_numbers)))
24
25
26 # Get user input and execute
27 if __name__ == "__main__":
28     try:
29         num = int(input("N= "))
30         print_fibonacci(num)
31     except ValueError:
32         print("Invalid input! Please enter a valid integer.")
33
```

```
ASS.3.4.py"
N= 7
0 1 1 2 3 5 8
```

## Task 2: One-shot Prompt – List Reversal Function

### Task Description #2

- Write a comment prompt to reverse a list and provide one example below the comment to guide Copilot.

**Prompt:** # Write a Python function that takes a list as input and returns the list in reverse order

**Code and output :**

```

1  def reverse_list(lst):
2      """Reverse a list using the reverse() method."""
3      lst.reverse()
4      return lst
5
6  # Take user input
7  try:
8      user_input = input("Enter list elements separated by spaces: ")
9      lst = list(map(int, user_input.split()))
10
11     # Reverse and print
12     reversed_list = reverse_list(lst)
13     print(reversed_list)
14 except ValueError:
15     print("Error: Please enter valid numbers separated by spaces")
16

```

Enter list elements separated by spaces: 1 2 3  
Original list: ['1', '2', '3']  
Reversed list: [3, 2, 1]

Enter list elements separated by spaces: 1 2 3  
[3, 2, 1]

## Task 3: Few-shot Prompt – String Pattern Matching

### Task Description #3

- Write a comment with 2–3 examples to help Copilot understand how to check if a string starts with a capital letter and ends with a period.

**Prompt:** # Write a Python function is\_valid() that checks whether a string starts with a capital letter  
# and ends with a period (.)

**Code and output :**

```

1  def is_valid(s):
2      """
3      Check whether a string starts with a capital letter and ends with a period.
4
5      Args:
6      | s (str): The string to validate
7
8      Returns:
9      | bool: True if starts with capital letter and ends with period, False otherwise
10     """
11     if len(s) < 2:
12         return False
13     return s[0].isupper() and s[-1] == '.'
14
15
16 # Test cases
17 if __name__ == "__main__":
18     test_cases = [
19         "Hello.",
20         "hello.",
21         "Hello",
22         "WORLD.",
23         "Python.",
24         "python.",
25         "A.",
26         "a."
27     ]
28
29     print("Testing is_valid() function:")
30     for test in test_cases:
31         result = is_valid(test)
32         print(f'Input: "{test}" -> Output: {result}')
33
34

```

Input: "Hello." -> Output: True  
Input: "hello." -> Output: False  
Input: "Hello" -> Output: False  
Input: "WORLD." -> Output: True  
Input: "Python." -> Output: True  
Input: "python." -> Output: False  
Input: "A." -> Output: True  
Input: "a." -> Output: False

## Task 4: Zero-shot vs Few-shot – Email Validator

### Task Description #4

- First, prompt Copilot to write an email validation function using zero-shot (just the task in comment).
- Then, rewrite the prompt using few-shot examples.

Prompt: # Write a Python function that validates whether a given string is a valid email address

# Write a Python function that validates whether a given string is a valid email address

# Examples:

# Input: "user@example.com" → Output: True

# Input: "userexample.com" → Output: False

# Input: "user@com" → Output: False

Code and output :

```
1 def is_valid_email(email):
21
22     # Check if domain part has at least one dot and parts are not empty
23     if '.' not in domain_part:
24         return False
25
26     # Check if domain has valid structure (e.g., example.com)
27     domain_parts = domain_part.split('.')
28     if len(domain_parts) < 2:
29         return False
30
31     # Check that all parts are non-empty
32     for part in domain_parts:
33         if not part:
34             return False
35
36     return True
37
38
39 # Test cases
40 if __name__ == "__main__":
41     test_cases = [
42         "user@example.com",
43         "userexample.com",
44         "user@com",
45         "user@domain.co.uk",
46         "john.doe@company.org",
47         "@example.com",
48         "user@",
49         "user..name@example.com",
50         "user@example",
51         "valid.email@test.com"
52     ]
53
54     print("Testing is_valid_email() function:")
55     for test in test_cases:
56         result = is_valid_email(test)
57         print(f'Input: "{test}" → Output: {result}')
58
```

```
Input: "user@domain.co.uk" → Output: True
Input: "john.doe@company.org" → Output: True
Input: "@example.com" → Output: False
Input: "user@" → Output: False
Input: "user..name@example.com" → Output: True
Input: "user@example" → Output: False
Input: "valid.email@test.com" → Output: True
```

## Task 5: Prompt Tuning – Summing Digits of a Number

### Task Description #5

- Experiment with 2 different prompt styles to generate a function that returns the sum of digits of a number.

Style 1: Generic task prompt

Style 2: Task + Input/Output example

**Prompt:**

**1) # Write a Python function that returns the sum of digits of a given number**

**2)# Write a Python function that returns the sum of digits of a number**

**# Example:**

**# Input: 123 → Output: 6**

**Code and output :**

```
1 def sum_of_digits(n):
2     """
3     Returns the sum of digits of a given number.
4
5     Args:
6         n (int): The number to calculate digit sum
7
8     Returns:
9         int: Sum of all digits in the number
10    """
11    # Convert to absolute value to handle negative numbers
12    n = abs(n)
13
14    # Convert to string and sum each digit
15    return sum(int(digit) for digit in str(n))
16
17
18 # Get user input and execute
19 if __name__ == "__main__":
20     try:
21         num = int(input("Enter a number: "))
22         result = sum_of_digits(num)
23         print(f"Input: {num} → Output: {result}")
24     except ValueError:
25         print("Error: Please enter a valid integer")
26
```

Input: 12345 → Output: 15

Input: 0 → Output: 0

Input: 999 → Output: 27

Input: -456 → Output: 15

Enter a number: 123

Input: 123 → Output: 6

