AI ASSISTANT CODING ASSIGNMENT-4.2

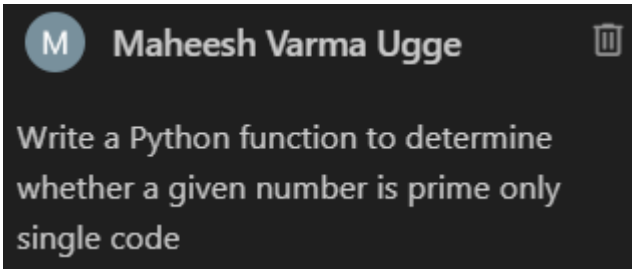
Task Description-1

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

Expected Output-1

- A basic Python function to check if a number is prime, demonstrating correct logical conditions without relying on examples or additional context

PROMPT:



CODE:

```python
def is_prime(n):
    if n <= 1:
        return False
    if n <= 3:
        return True
    if n % 2 == 0 or n % 3 == 0:
        return False
    i = 5
    while i * i <= n:
        if n % i == 0 or n % (i + 2) == 0:
            return False
        i += 6
    return True

if __name__ == "__main__":
    try:
        num = int(input("Enter a number: "))
        if is_prime(num):
            print(f"{num} is a prime number.")
        else:
            print(f"{num} is not a prime number.")
    except ValueError:
        print("Invalid input. Please enter an integer.")
```

Output:

```
PS C:\Users\mahee\OneDrive\Documents\Ai_coding> & C:/Users/mah
ding/is_prime.py
Enter a number: 3
3 is a prime number.
PS C:\Users\mahee\OneDrive\Documents\Ai_coding> & C:/Users/mah
ding/is_prime.py
Enter a number: 4
4 is not a prime number.
PS C:\Users\mahee\OneDrive\Documents\Ai_coding> & C:/Users/mah
ding/is_prime.py
Enter a number: 1.5
Invalid input. Please enter an integer.
PS C:\Users\mahee\OneDrive\Documents\Ai_coding> & C:/Users/mah
ding/is_prime.py
Enter a number: -3
-3 is not a prime number.
PS C:\Users\mahee\OneDrive\Documents\Ai_coding>
```

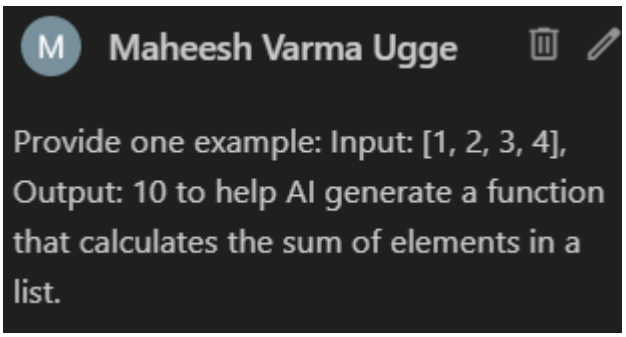
Task Description-2

- 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.

Expected Output-2

- A correct conversion function guided by the single example.

PROMPT:



CODE:

```python
def calculate_sum(numbers):
    total = 0
    for number in numbers:
        total += number
    return total

if __name__ == "__main__":
    input_list = [1, 2, 3, 4]
    output_sum = calculate_sum(input_list)
    print(f"Input: {input_list}, Output: {output_sum}")
```

Output:

```
PS C:\Users\mahee\OneDrive\Documents\Ai_coding> & C:/Users/mahee
ding/list_sum.py
Input: [1, 2, 3, 4], Output: 10
PS C:\Users\mahee\OneDrive\Documents\Ai_coding>
```

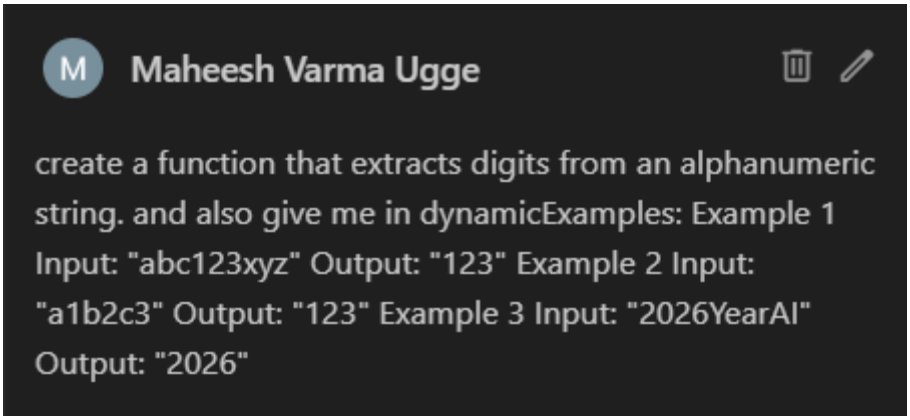
Task Description-3

- Few-shot: Give 2–3 examples to create a function that extracts digits from an alphanumeric string**.**

Expected Output-3

- Accurate function that returns only the digits from alphanumeric string

PROMPT:



CODE:

```python
def extract_digits(s):
    return "".join(filter(str.isdigit, s))

def run_tests():
    test_cases = {
        "abc123xyz": "123",
        "a1b2c3": "123",
        "2026YearAI": "2026",
        "NoDigits": "",
        "12345": "12345",
        "": ""
    }
    for input_str, expected_output in test_cases.items():
        assert extract_digits(input_str) == expected_output
    print("All test cases passed!")

def interactive_mode():
    print("\nEnter a string to extract digits from (or 'quit' to exit).")
    while True:
        user_input = input("Input: ")
        if user_input.lower() == 'quit':
            break
        output = extract_digits(user_input)
        print(f"Output: {output}")
if __name__ == "__main__":
    run_tests()
    interactive_mode()
```

OUTPUT:

```
Enter a string to extract digits from (or 'quit' to exit).
Input: xdfgdhtjknl1234567
Output: 1234567
Input: geg3
Output: 3
Input: 323s
Output: 323
Input: -234sdfgh
Output: 234
Input: 4-3
Output: 43
```

Task Description-4

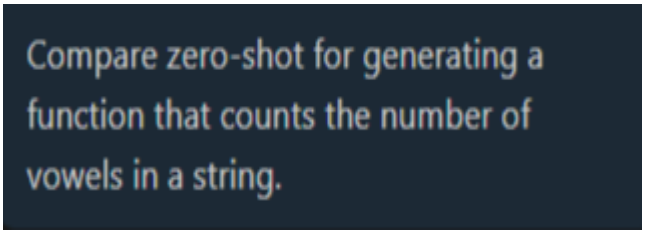
- Compare zero-shot vs few-shot prompting for generating a function that counts the number of vowels in a string.

Expected Output-4

- Output comparison + student explanation on how examples helped the model.

PROMPT:

zero-shot:



CODE:

```python
# Approach 1: Simple Loop
def count_vowels_loop(string):
    vowels = "aeiouAEIOU"
    count = 0
    for char in string:
        if char in vowels:
            count += 1
    return count

# Approach 2: List Comprehension
def count_vowels_comprehension(string):
    vowels = "aeiouAEIOU"
    return len([char for char in string if char in vowels])

# Test Cases
if __name__ == "__main__":
    test_strings = ["Hello World", "Python", "AEIOU"]

    for test in test_strings:
        result1 = count_vowels_loop(test)
        result2 = count_vowels_comprehension(test)
        print(f"Input: \"{test}\" -> Loop: {result1}, Comprehension: {result2}")
```

OUTPUT:

```
PS C:\Users\mahee\OneDrive\Documents\Ai_coding> & C:/Users/mahee/AppData
coding/vowel_counter_comparison.py
Input: "Hello World" -> Loop: 3, Comprehension: 3
Input: "Python" -> Loop: 1, Comprehension: 1
Input: "AEIOU" -> Loop: 5, Comprehension: 5
PS C:\Users\mahee\OneDrive\Documents\Ai_coding>
```

PROMPT:

Few-shot:

few-shot prompting for generating a function that counts the number of vowels in a string.
Examples:

Input: "hello"
Output: 2

Input: "AEIOU"
Output: 5

Input: "ChatGPT"
Output: 2

CODE:

```python
# Approach 1: Simple Loop
def count_vowels_loop(string):
    vowels = "aeiouAEIOU"
    count = 0
    for char in string:
        if char in vowels:
            count += 1
    return count

# Approach 2: List Comprehension
def count_vowels_comprehension(string):
    vowels = "aeiouAEIOU"
    return len([char for char in string if char in vowels])

# Test Cases - Few-Shot Examples
if __name__ == "__main__":
    examples = [
        ("hello", 2),
        ("AEIOU", 5),
        ("ChatGPT", 2),
    ]

    print("Few-Shot Examples:")
    for test, expected in examples:
        result1 = count_vowels_loop(test)
        result2 = count_vowels_comprehension(test)
        print(f"Input: \"{test}\" -> Output: {result1} (Expected: {expected})")
```

OUTPUT:

```
PS C:\Users\mahee\OneDrive\Documents\Ai_coding> & C:/
coding/vowel_counter_comparison.py
Few-Shot Examples:
Input: "hello" -> Output: 2 (Expected: 2)
Input: "AEIOU" -> Output: 5 (Expected: 5)
Input: "ChatGPT" -> Output: 1 (Expected: 2)
PS C:\Users\mahee\OneDrive\Documents\Ai_coding>
```

Task Description-5

- 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.

Expected Output-5

- A function that handles all cases with correct logic based on example patterns**.**

PROMPT:

```
generate a function that determines the
minimum of three numbers without using
the built-in min() function.

Examples:

Input: a = 3, b = 7, c = 5
Output: 3

Input: a = -1, b = 4, c = 2
Output: -1

Input: a = 10, b = 10, c = 5
Output: 5
```

CODE:

```python
def minimum_of_three(a, b, c):
    """Find the minimum of three numbers without using min()."""
    if a <= b and a <= c:
        return a
    elif b <= c:
        return b
    else:
        return c
# Test cases
print(minimum_of_three(3, 7, 5))      # Output: 3
print(minimum_of_three(-1, 4, 2))     # Output: -1
print(minimum_of_three(10, 10, 5))    # Output: 5
```

OUTPUT:

```
PS C:\Users\mahee\OneDrive\Documents\Ai_coding> & C:/Users/ma
coding/MINIMUM.PY
3
-1
5
```