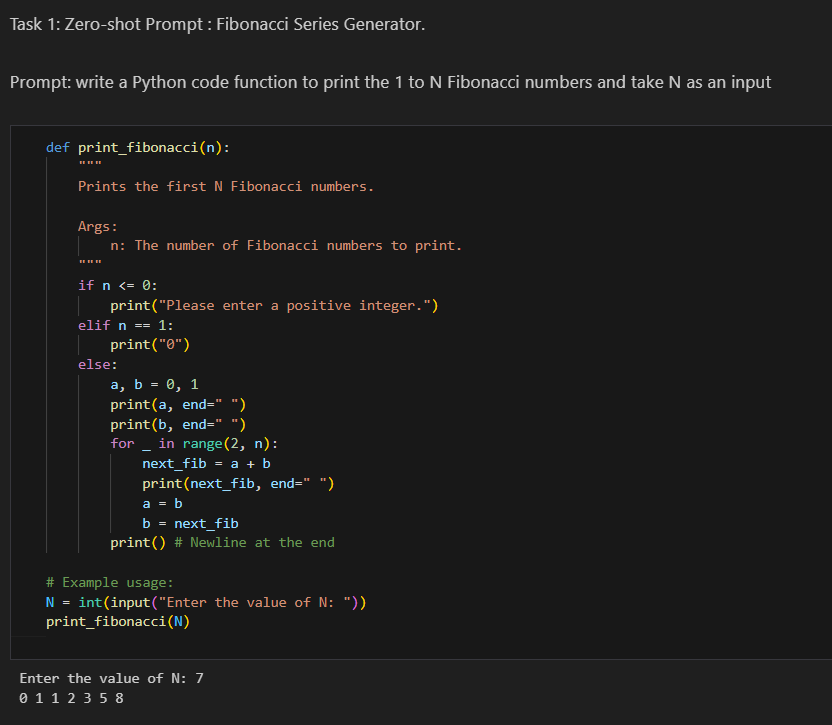
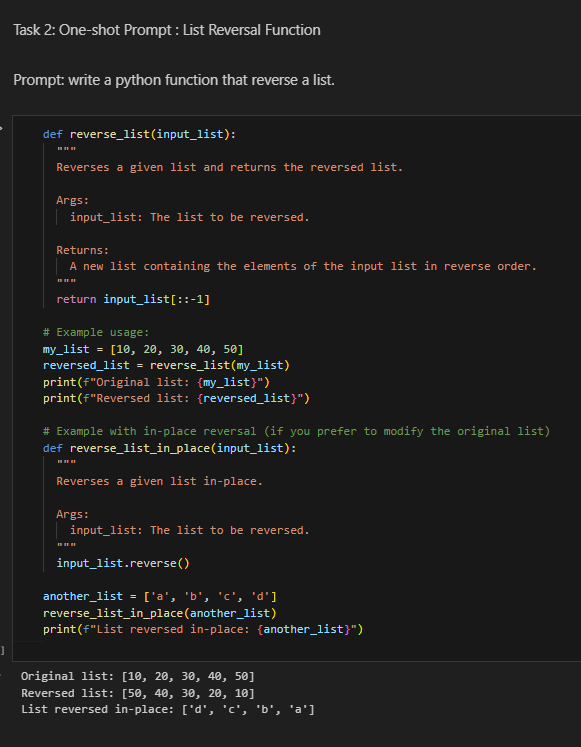
**ASSIGNMENT-3.4**

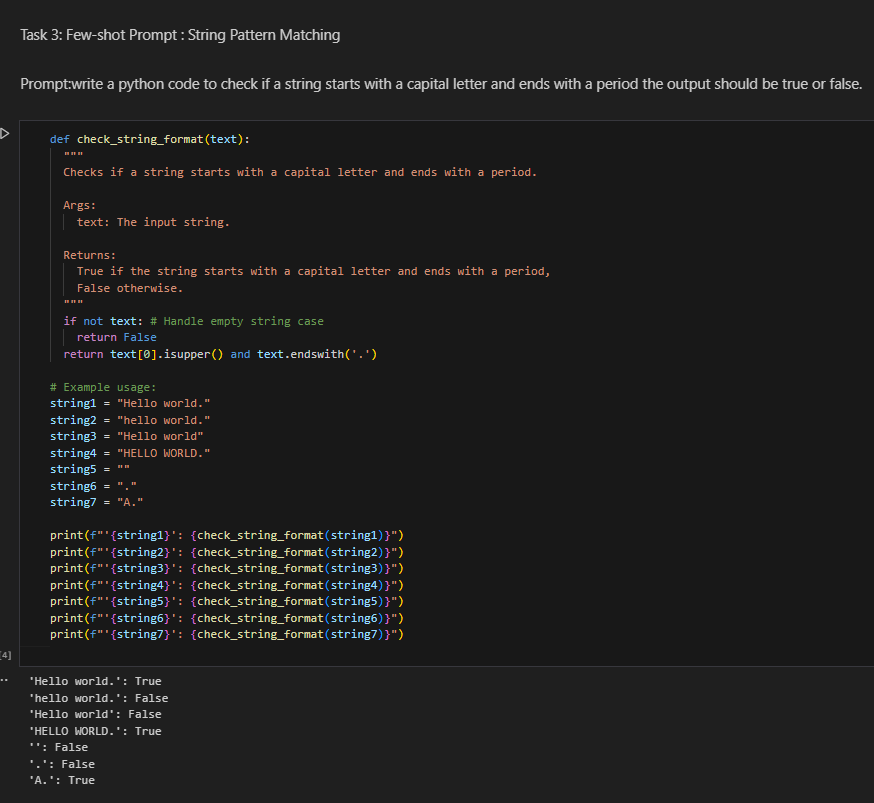
**Task 1: Zero-shot Prompt – Fibonacci Series Generator  
Task Description #1  
• Without giving an example, write a single comment prompt asking GitHub Copilot to  
generate a Python function to print the first N Fibonacci numbers.  
Expected Output #1  
• A complete Python function generated by Copilot without any example provided.  
• Correct output for sample input N = 7 ➝ 0 1 1 2 3 5 8  
• Observation on how Copilot understood the instruction with zero contex**



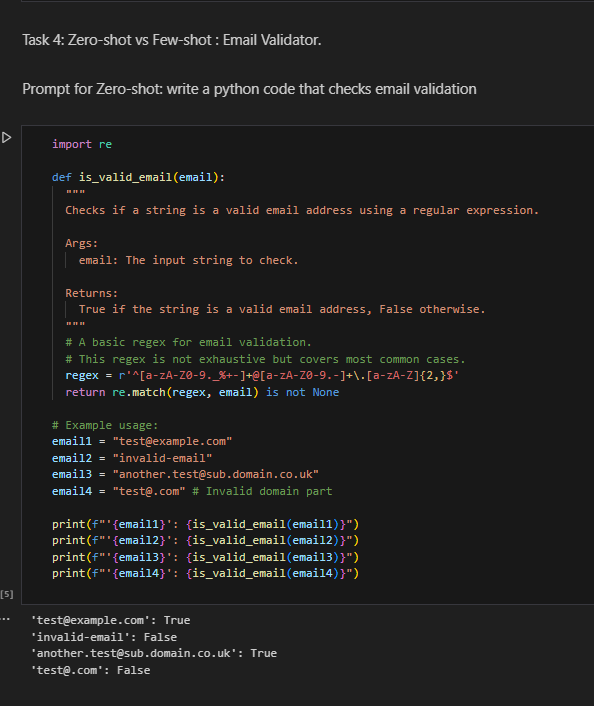
Task 2: One-shot Prompt – List Reversal Function  
Task Description #2  
• Write a comment prompt to reverse a list and provide one example below the  
comment to guide Copilot.  
Expected Output #2  
• Copilot-generated function to reverse a list using slicing or loop.  
• Output: [3, 2, 1] for input [1, 2, 3]  
• Observation on how adding a single example improved Copilot’s accuracy

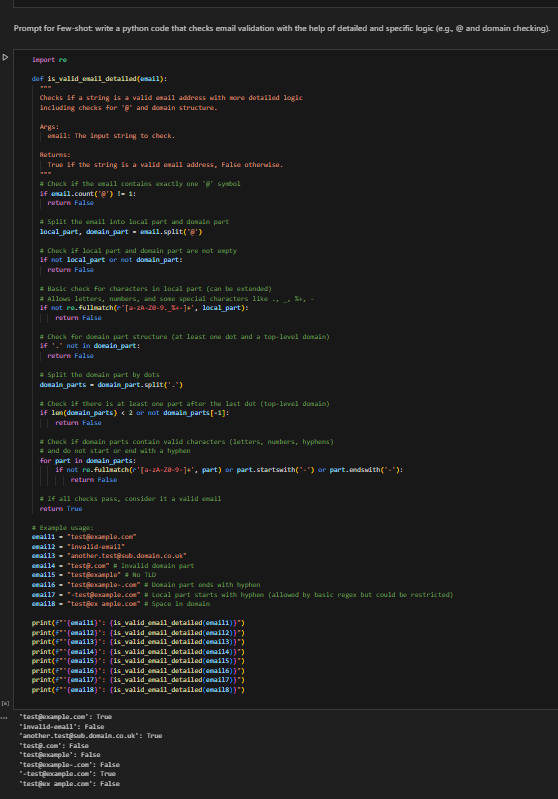


Task 3: Few-shot Prompt – String Pattern Matching  
Task Description #3  
• Write a comment with 2–3 examples to help Copilot understand how to check if a  
string starts with a capital letter and ends with a period.  
Expected Output #3  
• A function is\_valid() that checks the pattern.  
• Output: True or False based on input.  
• Students reflect on how multiple examples guide Copilot to generate more accurate  
code



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.  
Expected Output #4  
• Compare both outputs:  
Zero-shot may result in basic or generic validation.  
Few-shot gives detailed and specific logic (e.g., @ and domain checking).  
• Submit both code versions and note how few-shot improves reliability





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  
Expected Output #5  
• Two versions of the sum\_of\_digits() function.  
• Example Output: sum\_of\_digits(123) ➝ 6  
• Short analysis: which prompt produced cleaner or more optimized code and why?

