

# Assignment -3.1

Hall ticket no:2303A510A6

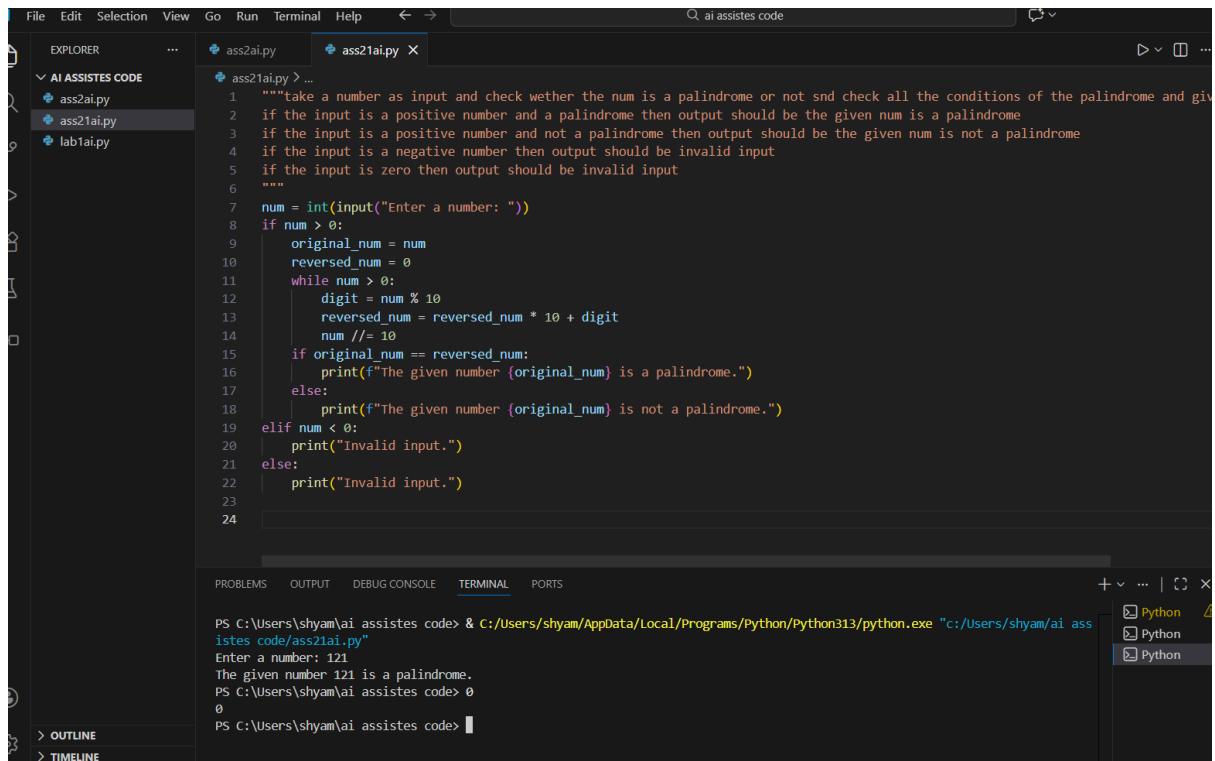
Batch no:02

## Question 1: Zero-Shot Prompting (Palindrome Number Program)

Write a zero-shot prompt (without providing any examples) to generate a Python function that checks whether a given number is a palindrome.

Task:

- Record the AI-generated code.
- Test the code with multiple inputs.
- Identify any logical errors or missing edge-case handling.



```
File Edit Selection View Go Run Terminal Help ⏎ → 🔍 ai assistes code
EXPLORER ... ass2ai.py ass21ai.py
ass21ai.py > ...
1 """Take a number as input and check whether the num is a palindrome or not and check all the conditions of the palindrome and give
2 if the input is a positive number and a palindrome then output should be the given num is a palindrome
3 if the input is a positive number and not a palindrome then output should be the given num is not a palindrome
4 if the input is a negative number then output should be invalid input
5 if the input is zero then output should be invalid input
6 """
7 num = int(input("Enter a number: "))
8 if num > 0:
9     original_num = num
10    reversed_num = 0
11    while num > 0:
12        digit = num % 10
13        reversed_num = reversed_num * 10 + digit
14        num //= 10
15    if original_num == reversed_num:
16        print(f"The given number {original_num} is a palindrome.")
17    else:
18        print(f"The given number {original_num} is not a palindrome.")
19 elif num < 0:
20     print("Invalid input.")
21 else:
22     print("Invalid input.")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\shyam\ai assistes code> & c:/Users/shyam/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/shyam/ai assistes code/ass21ai.py"
Enter a number: 121
The given number 121 is a palindrome.
PS C:\Users\shyam\ai assistes code> 0
PS C:\Users\shyam\ai assistes code>
```

write a one-shot prompt by providing one input-output example and ask the AI to generate a Python function to compute the factorial of a given number.

Example:

Input: 5 → Output: 120

Task:

- Compare the generated code with a zero-shot solution.

- Examine improvements in clarity and correctness.

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows files: ass2ai.py, ass21ai.py, ass22ai.py, and lab1ai.py. ass22ai.py is the active file.
- Code Editor:** Displays Python code for calculating factorial. The code uses an iterative approach with a for loop. A comment at the top indicates it computes the factorial of a given number.
- Terminal:** Shows the command PS C:\Users\shyam\ai assistes code> & C:/Users/shyam/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/shyam/ai assistes code/ass22ai.py". It then prompts for input: Enter a number: 5. The output shows the factorial of 5 is 120.
- Bottom Status Bar:** Shows the outline icon.

Aspect	One-Shot Prompting	Zero-Shot Prompting
Guidance	Includes example (5 → 120)	No example provided
Clarity	More explicit task understanding	Relies on model's assumption
Approach	Iterative solution	Recursive solution
Readability	Easier for beginners	Slightly more complex
Correctness	Correct for non-negative integers	Correct but risk of recursion depth

### Question 3: Few-Shot Prompting (Armstrong Number Check)

**Write a few-shot prompt by providing multiple input-output examples to guide the AI in generating a Python function to check whether a given number is an Armstrong number.**

**Examples:**

- Input: 153 → Output: Armstrong Number
- Input: 370 → Output: Armstrong Number
- Input: 123 → Output: Not an Armstrong Number

## Task:

- Analyze how multiple examples influence code structure and accuracy.
- Test the function with boundary values and invalid inputs.

The screenshot shows a VS Code interface with the following details:

- EXPLORER** view: Shows files ass2ai.py, ass21ai.py, ass22ai.py, and lab1ai.py.
- CODE EDITOR**: The active file is ass22ai.py, containing Python code to check if a number is Armstrong. It includes comments and logic for handling negative numbers, zero, and non-integer inputs.
- TERMINAL**: Shows command-line output for three runs of the script with inputs 153, 370, and 123, correctly identifying them as Armstrong or not.
- AI ASSISTES CODE**: A sidebar on the right provides AI-generated code snippets and instructions for interacting with the AI.

## Influence of Multiple Examples on Code Structure and Accuracy

- Multiple examples clearly demonstrate **both positive and negative cases**, reducing ambiguity.
- The AI correctly identifies that:
  - Each digit must be raised to the **number of digits**.
  - The sum must be compared with the original number.

Few-shot prompting helps the AI infer:

- The definition of an Armstrong number
- The expected output wording
- The correct handling of digit extraction and exponentiation

## (Optional Extension)

### Question 4: Context-Managed Prompting (Optimized Number Classification)

Design a context-managed prompt with clear instructions and constraints to generate an optimized Python program that classifies a number as prime, composite, or neither.

**ask:**

- Ensure proper input validation.
- Optimize the logic for efficiency.
- Compare the output with earlier prompting strategies.

The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows files: ass2ai.py, ass21ai.py, ass22ai.py, and lab1ai.py.
- Code Editor:** Displays the content of ass22ai.py, which is a Python program to classify numbers as prime, composite, or neither. The code uses a loop to check divisibility from 2 to the square root of the number.
- Terminal:** Shows command-line interactions with the code. It asks for a number (5, 4, 1) and prints the results: "The given number 5 is a prime number.", "The given number 4 is a composite number.", and "The given number 1 is neither prime nor composite."
- AI Assistant Panel:** A sidebar titled "Build with Agent" contains a message: "AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase." It also shows a "Python" icon.
- Bottom Bar:** Includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. It also shows a "Describe what to build next" input field and some status indicators.

### Question 5: Zero-Shot Prompting (Perfect Number Check)

Write a zero-shot prompt (without providing any examples) to generate a Python function that checks whether a given number is a perfect number.

**Task:**

- Record the AI-generated code.
- Test the program with multiple inputs.
- Identify any missing conditions or inefficiencies in the logic.

```

1 def is_perfect_number(n):
2     """
3         Checks if a given number is a perfect number.
4         A perfect number is a positive integer that is equal to the sum of its proper positive divisors. print true or False
5     """
6     if n <= 0:
7         return False
8
9     sum_of_divisors = 0
10    for i in range(1, n):
11        if n % i == 0:
12            sum_of_divisors += i
13
14    return sum_of_divisors == n
15 num = int(input("Enter a number: "))
16 if is_perfect_number(num):
17     print("True")
18 else:
19     print("False")
20

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\shyam\ai assistes code> & C:/Users/shyam/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/shyam/ai assistes code/ass2ai.py"  
 Enter a number: 12  
 False  
 PS C:\Users\shyam\ai assistes code> & C:/Users/shyam/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/shyam/ai assistes code/ass2ai.py"  
 Enter a number: 6  
 True  
 PS C:\Users\shyam\ai assistes code> & C:/Users/shyam/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/shyam/ai assistes code/ass2ai.py"  
 Enter a number: 0  
 False  
 PS C:\Users\shyam\ai assistes code> & C:/Users/shyam/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/shyam/ai assistes code/ass2ai.py"  
 Enter a number: 1  
 False

Ln 4, Col 121 Spaces:4 UTF-8 CRLF { } Python

## Missing Conditions and Inefficiencies

- No validation for negative numbers or zero.
- Loop runs up to n-1, which is inefficient for large values.
- Function does not handle non-integer inputs.

## Question 6: Few-Shot Prompting (Even or Odd Classification with Validation)

**Write a few-shot prompt by providing multiple input-output examples to guide the AI in generating a Python program that determines whether a given number is even or odd, including proper input validation.**

### Examples:

- Input: 8 → Output: Even
- Input: 15 → Output: Odd
- Input: 0 → Output: Even

### Task:

- Analyze how examples improve input handling and output clarity.
- Test the program with negative numbers and non-integer inputs.

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows files in the "AI ASSISTES CODE" folder: ass2ai.py, ass21ai.py, ass22ai.py, and lab1ai.py. ass22ai.py is the active file.
- Code Editor:** Displays the content of ass22ai.py:

```
1  ...
2  | Python program that determines whether a given number is even or odd, including proper input validation.
3  | Input: 8 → Output: Even
4  | • Input: 15 → Output: Odd
5  | • Input: 0 → Output: Even
6  ...
7  ...
8  num = int(input("Enter a number: "))
9  if num > 0:
10     if num % 2 == 0:
11         print(f"The given number {num} is Even.")
12     else:
13         print(f"The given number {num} is Odd.")
14 elif num < 0:
15     print("Invalid input.")
16 else:
17     print(f"The given number {num} is Even.")
```
- Terminal:** Shows command-line output for running the script:

```
PS C:\Users\shyam\ai assistes code> & C:/Users/shyam/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/shyam/ai assistes code/ass22ai.py"
Enter a number: 15
The given number 15 is Odd.
PS C:\Users\shyam\ai assistes code> & C:/Users/shyam/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/shyam/ai assistes code/ass22ai.py"
Enter a number: 8
The given number 8 is Even.
PS C:\Users\shyam\ai assistes code> & C:/Users/shyam/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/shyam/ai assistes code/ass22ai.py"
Enter a number: 0
The given number 0 is Even.
PS C:\Users\shyam\ai assistes code>
```

### Analysis: Effect of Examples on Input Handling and Output Clarity

- Examples guide the program to return only “Even” or “Odd”.
- Including 0 → Even ensures correct handling of zero.
- Input validation is added to prevent non-integer inputs.