

# LAB ASSIGNMENT - 03

---

**Hall Ticket No :** 2303a51027

**Name :** Aravind Reddy

**Course Title :** AI Assistant Coding

---

## Experiment – Prompt Engineering Techniques

### Task Description :

Design and refine prompts using different prompting strategies to generate Python programs for basic computational problems.

---

### 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.

The screenshot shows a Visual Studio Code (VS Code) interface. The left sidebar has sections for EXPLORER, SOURCE CONTROL, CHANGES, AI ASSISTED CODING, and TERMINAL. The main area displays a Python file named 'Assignment-03.py' with the following code:

```
79
80  ...
81 Write a Python Program to find a number is Palindrome or not Palindrome.
82 ...
83
84 def is_palindrome(num):
85     if num < 0:
86         return False
87     return str(num) == str(num)[::-1]
88
89 try:
90     num = int(input("Enter a number: "))
91     if is_palindrome(num):
92         print(f"{num} is a Palindrome")
93     else:
94         print(f"{num} is not a Palindrome")
95 except ValueError:
96     print("Accept only numbers")
97
```

The TERMINAL tab at the bottom shows the output of running the script:

```
/usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: 121
121 is a Palindrome
% (base) =  AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
121 is not a Palindrome
% (base) =  AI Assisted Coding %%
```

## Question 2: One-Shot Prompting (Factorial Calculation)

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 a VS Code interface with the following details:

- Explorer:** Shows a tree view of files: AI ASSISTED CODING, Assignments, Assignment - 01.py, Assignment - 03.py, Program - 1.py, and Program - 2.py.
- Editor:** Displays a Python script named Assignment - 03.py. The code defines a factorial function and handles user input for a number to calculate its factorial.
- Terminal:** Shows the output of running the script. It prompts for a number, calculates the factorial of 5, and prints the result.
- Sidebar:** Includes sections for Outline, Timeline, and Pretty TypeScript Error.

```
98
99 ...
100 Write a Python Program to find factorial of a Number.
101
102 num = 5
103 display Factorial of 5 is 120
104 ...
105 def factorial(n):
106     if n < 0:
107         return None
108     elif n == 0:
109         return 1
110     else:
111         result = 1
112         for i in range(1, n + 1):
113             result *= i
114         return result
115
116 try:
117     num = int(input("Enter a number: "))
118     fact = factorial(num)
119     if fact is None:
120         print("Factorial is not defined for negative numbers")
121     else:
122         print(f"Factorial of {num} is {fact}")
123 except ValueError:
124     print("Accept only numbers")
125
```

```
/usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
● (base) ➜ AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: 5
Factorial of 5 is 120
● (base) ➜ AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: 120
Factorial of 5 is 120
↳ (base) ➜ AI Assisted Coding
```

## 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 code editor interface with a dark theme. In the Explorer sidebar, there are several projects and files listed under 'SOURCE CONTROL: CHANGES' and 'AI ASSISTED CODING'. The main editor area displays a Python script named 'Assignment-03.py'. The code defines a function 'is\_armstrong' that checks if a given number is an Armstrong number by summing the powers of its digits. It also includes a try-except block to handle non-integer inputs. The terminal below the editor shows the execution of the script and its output for various test cases.

```
126 """
127 Write a Python Program to check a Number is Armstrong or not.
128
129 num = 153
130 display 153 is an Armstrong number
131
132 num = 123
133 display 123 is not an Armstrong number
134
135 num = -370
136 display -370 is not an Armstrong number
137
138 num = 'abc'
139 display accept only numbers
140
141 num = 0
142 display 0 is an Armstrong number
143 """
144
145 def is_armstrong(num):
146     if num < 0:
147         return False
148     num_str = str(num)
149     order = len(num_str)
150     sum_of_powers = sum(int(digit) ** order for digit in num_str)
151     return sum_of_powers == num
152
153 try:
154     num = int(input("Enter a number: "))
155     if is_armstrong(num):
156         print(f"{num} is an Armstrong number")
157     else:
158         print(f"{num} is not an Armstrong number")
159     except ValueError:
160         print("Accept only numbers")
161
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER
/usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
● (base) - AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: 153
153 is an Armstrong number
● (base) - AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
● (base) - AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: 123
123 is not an Armstrong number
● (base) - AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: -100
-100 is not an Armstrong number
● (base) - AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: abc
Accept only numbers
● (base) - AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: 0
0 is an Armstrong number
● (base) - AI Assisted Coding
```

## (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.

#### Task:

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

## 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.

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

- EXPLORER**: Shows a file tree with projects like "AI ASSISTED CODING", "Assignment - 01.py", "Assignment - 03.py", "Program-1.py", and "Program-2.py".
- SOURCE CONTROL: CHANGES**: Shows modifications for "Assignment - 03.py".
- CODE EDITOR**: Displays the content of "Assignment-03.py". The code defines a function `is\_perfect\_number` and uses a try-except block to handle user input.
- TERMINAL**: Shows the execution of the script and its output. It asks for a number, accepts 6, and prints "6 is a Perfect Number". It then asks for another number, accepts -6, and prints "-6 is not a Perfect Number".

## 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 (VS Code) interface. The top bar includes standard icons for file operations and a search bar labeled "AI Assisted Coding". The left sidebar has sections for "EXPLORER", "SOURCE CONTROL: CHANGES", and "AI ASSISTED CODING". The main workspace displays a Python script named "Assignment-03.py". The code is as follows:

```
219
220     ...
221
222 Write a Python Program to check whether a number is even or odd.
223
224 num = 24
225 display 24 is an Even number
226
227 num = 97
228 display 97 is an Odd number
229
230 num = -12
231 display -12 is an Even number
232
233 num = 'hello'
234 display accept only numbers
235
236 num = 0
237 display 0 is an Even number
238 ...
239
240
241 def check_even_odd(num):
242     return "Even number" if num % 2 == 0 else "Odd number"
243 try:
244     num = int(input("Enter a number: "))
245     result = check_even_odd(num)
246     print(f"\{num\} is an {result}")
247 except ValueError:
248     print("accept only numbers")
```

The terminal below shows the execution of the script with various inputs:

```
/usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
● (base) - AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: 24
24 is an Even number
● (base) - AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: 99
99 is an Odd number
● (base) - AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: -34
-34 is an Even number
● (base) - AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: -33
-33 is an Odd number
● (base) - AI Assisted Coding /usr/local/bin/python3 "/Users/aravindreddy/Desktop/My-Information/College/AI Assisted Coding/Assignments/Assignment-03.py"
Enter a number: 0
0 is an Even number
● (base) - AI Assisted Coding
```

The bottom status bar indicates the current file is "Assignment-03.py", the line is 220, column 1, and the encoding is UTF-8.