

ASSIGNMENT-4.2

NAME-TEENESWARI

ROLLNO:2303A51932

TASK-1: ZERO-SHOT PROMPTING

PROMPT: Write a Python function to determine whether a given number is prime.

CODE:

The screenshot shows a Jupyter Notebook interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Title Bar:** DevOps exp 2,3,4
- Left Sidebar (OPEN EDITORS):**
 - JS index.js
 - AI ASS Coding.ipynb
 - AI ASS Coding...
 - Untitled-1
 - ai 3.1.py
 - ai 4.2
 - ai 4.2.py
 - ai 2.5.py
 - Untitled-2
 - Untitled-3
 - ai 2.2.py
- Left Sidebar (DEVOPS EXP 2,3,4):**
 - github workflow...
 - .ci.yml
 - zenencoder
 - zenflow workflows
 - ai 1932
 - ai 2.2.py
 - ai 2.5.py
 - ai 3.1.py
 - ai 4.2
 - ai 4.2.py
 - node_modules
 - .gitignore
 - AI ASS Coding
 - AI ASS Coding.ipynb
 - Dockerfile
 - JS index.js
 - (1) package-lock.json
 - (1) package.json
- Code Cell:**

```
ai 1932 > ai 4.2.py > ...
1  #write a Python function to determine whether a given number is prime.
2  def is_prime(n):
3      if n <= 1:
4          return False
5      for i in range(2, int(n**0.5) + 1):
6          if n % i == 0:
7              return False
8      return True
9  # Example usage
10 print(is_prime(11)) # Should return True
11 print(is_prime(15)) # Should return False
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27  #write a Python function that calculates the sum of elements in a list.
ZenCoder
```
- Bottom Navigation:** PROBLEMS, OUTPUT, TERMINAL, DEBUG CONSOLE, PORTS, SPELL CHECKER, JUPYTER. Filter (e.g. text, lexcludeText, t...), Code.
- Output Area:**

```
Perfect number
Not a perfect number

[Done] exited with code=0 in 0.283 seconds

[Running] python -u "c:\Users\heman\OneDrive\Desktop\DevOps exp 2,3,4\ai 1932\ai 4.2.py"
True
False

[Done] exited with code=0 in 0.301 seconds

[Running] python -u "c:\Users\heman\OneDrive\Desktop\DevOps exp 2,3,4\ai 1932\ai 4.2.py"
True
False
15
```

OBSERVATION:

- AI model understands the concept of a prime number without being given any examples or additional guidance
- It applies correct mathematical reasoning purely from the instruction
- The model generates syntactically correct and logically sound Python code

TASK-2

PROMPT: Write a Python function that calculates the sum of elements in a list.

Example:

Input: [1, 2, 3, 4]

Output: 10

CODE:

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

- File Explorer:** Shows files like `index.js`, `AI ASS Coding.ipynb`, and multiple `ai` files.
- Open Editors:** Shows `ai 1932 > ai 4.2.py > ...` and other files.
- Code Editor:** Displays a Python script for determining if a number is prime. The code includes a `is_prime` function and example usage.
- Terminal:** Shows the output of running the script, including test cases and the command used.
- Status Bar:** Shows file paths, line numbers (Ln 69, Col 1), and other system information.

```
1 #Write a Python function to determine whether a given string is a prime.
2 Zencoder
3 def is_prime(n):
4     if n <= 1:
5         return False
6     for i in range(2, int(n**0.5) + 1):
7         if n % i == 0:
8             return False
9     return True
10 # Example usage
11 print(is_prime(11))  # Should return True
12 print(is_prime(15))  # Should return False
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
```

PROBLEMS 4 OUTPUT TERMINAL DEBUG CONSOLE PORTS SPELL CHECKER JUPYTER
Filter (e.g. text, exclude text, ...) Code

```
123456
[Done] exited with code=0 in 0.26 seconds

[Running] python -u "c:\Users\heman\OneDrive\Desktop\DevOps exp 2,3,4\ai 1932\ai 4.2.py"
True
False
15
123456
True
False
```

> OUTLINE
> TIMELINE
[Done] exited with code=0 in 0.279 seconds

OBSERVATION:

The single example guides the AI model to understand the expected input and output relationship. The model correctly generalizes the pattern from the example to any list of numbers.

TASK-3

PROMPT: Write a Python function that extracts only digits from an alphanumeric string.

Examples:

Input: "a1b2c3"

Output: "123"

Input: "x9y8z7"

Output: "987"

Input: "abc123def"

Output: "123" CODE:

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, Help, and several icons for navigating between tabs and windows.

The left sidebar displays the Explorer view, which lists open files and folders. Open files include `index.js`, `AI ASS Coding.ipynb`, `Untitled-1`, `ai 3.1.py`, `ai 4.2`, `ai 4.2.py`, `ai 2.5.py`, `Untitled-2`, `Untitled-3`, and `ai 2.2.py`. Folders like `DEVOPS EXP 2,3,4` and `node_modules` are also visible.

The main editor area contains Python code for extracting digits from alphanumeric strings and determining if a string is prime. The terminal at the bottom shows the execution of the `ai 4.2.py` script, which prints the number 15 and the string "123456".

```
15
[Done] exited with code=0 in 0.216 seconds

[Running] python -u "c:\Users\heman\OneDrive\Desktop\DevOps exp 2,3,4\ai 1932\ai 4.2.py"
True
False
15
123456

[Done] exited with code=0 in 0.26 seconds

[Running] python -u "c:\Users\heman\OneDrive\Desktop\DevOps exp 2,3,4\ai 1932\ai 4.2.py"
True
```

OBSERVATION:

-Multiple examples help the AI model clearly identify the pattern to be learned -The model focuses only on digit characters and ignores alphabetic content

-The AI demonstrates improved confidence and reduced ambiguity compared to zero shot and one shot prompting

TASK-4

PROMPT: ZERO-SHOT: Write a Python function that counts the number of vowels in a string.

FEW-SHOT: Write a Python function that counts the number of vowels in a string.

Examples:

Input: "hello"

Output: 2

Input: "AEIOU"

Output: 5

Input: "chatgpt"

Output: 2 CODE:

ZERO-SHOT:

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

- File Explorer:** Shows a tree view of files and folders. Opened files include `index.js`, `AI ASS Coding.ipynb`, `Untitled-1`, `ai 3.1.py`, `ai 4.2`, `ai 4.2.py`, `ai 2.5.py`, `Untitled-2`, and `Untitled-3`. A file `ai 4.2.py` is currently selected.
- Code Editor:** Displays a Python script named `ai 4.2.py` with the following content:

```
1 #write a Python function to determine whether a given string is a prime.
2 Zencoder
3 def is_prime(n):
4     if n <= 1:
5         return False
6     for i in range(2, int(n**0.5) + 1):
7         if n % i == 0:
8             return False
9     return True
10 # Example usage
11 print(is_prime(11)) # Should return True
12 print(is_prime(15)) # Should return False
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
```
- Bottom Status Bar:** Shows the current file is `master`, the code exited with `code=0` in 0.26 seconds, and the command `[Running] python -u "c:\Users\heman\OneDrive\Desktop\DevOps exp 2,3,4\ai 1932\ai 4.2.py"`.
- Bottom Right:** Includes tabs for `PROBLEMS`, `OUTPUT`, `TERMINAL`, `DEBUG CONSOLE`, `PORTS`, `SPELL CHECKER`, and `JUPYTER`. The `OUTPUT` tab is selected.
- Bottom Navigation:** Includes buttons for `Filter (e.g. text, excludeText, L...)`, `Code`, and other standard VS Code navigation icons.

FEW-SHOT:

The screenshot shows the Microsoft Visual Studio Code interface. The left sidebar has 'EXPLORER' and 'AI CODING' sections, with 'ai.py' selected. The main area shows the code for 'Assignment-3.1.py'. The terminal at the bottom displays the execution of the script, showing syntax errors and the output of the function calls.

```
File Edit Selection View Go Run Terminal Help

EXPLORER ... Assignment-3.1.py x
AI CODING
vscode
ai.py

D:\> Assignment-3.1.py <
Enter your OpenAI API key (Press 'Enter' to confirm or 'Escape' to cancel)

1 #
2 def extract_digits(text):
3     result = ""
4     for ch in text:
5         if ch.isdigit():
6             result += ch
7     return result
8
9 # Print statements
10 print(extract_digits("a1b2c3"))
11 print(extract_digits("abc09xyz"))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
+ - | x
Python Python

PS C:\Users\Jashwanth\AI coding> & C:/Users/Jashwanth/AppData/Local/Microsoft/WindowsApps/python3.12.exe d:/Assignment-3.1.py
12
99
PS C:\Users\Jashwanth\AI coding> & C:/Users/Jashwanth/AppData/Local/Microsoft/WindowsApps/python3.12.exe d:/Assignment-3.1.py
10
99
PS C:\Users\Jashwanth\AI coding> & C:/Users/Jashwanth/AppData/Local/Microsoft/WindowsApps/python3.12.exe d:/Assignment-3.1.py
True
False
PS C:\Users\Jashwanth\AI coding> & C:/Users/Jashwanth/AppData/Local/Microsoft/WindowsApps/python3.12.exe d:/Assignment-3.1.py
File "d:/Assignment-3.1.py", line 2
    result = ""
IndentationError: unexpected indent
PS C:\Users\Jashwanth\AI coding> & C:/Users/Jashwanth/AppData/Local/Microsoft/WindowsApps/python3.12.exe d:/Assignment-3.1.py
12
99
PS C:\Users\Jashwanth\AI coding>

Ln 1, Col 2  Spaces: 4  UTF-8  CRLF  ( ) Python  3.12.10 (Microsoft Store)
```

```
def count_vowels(text):
```

```
    vowels = "aeiouAEIOU"
```

```
    count = 0 for ch in text:
```

```
        if ch in vowels:
```

```
            count += 1 return
```

```
count
```

OBSERVATION:

FEW-SHOT OBSERVATION

The provided examples clearly define what characters should be counted as vowels. The model confidently includes both uppercase and lowercase vowels due to examples ZERO

SHOT:

zero shot prompting the AI guesses the intent based on general knowledge which may vary for ambiguous tasks

TASK-5

PROMPT:

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

Examples:

Input: 3, 7, 5

Output: 3

Input: 10, 2, 8

Output: 2

Input: 4, 4, 9

Output: 4 CODE:

```
File Edit Selection View Go Run Terminal Help
EXPLORER Assignment-3.1.py
AI CODING .vscode ai.py
Enter your OpenAI API key (Press Enter to confirm or Escape to cancel)

1 def find_min(a, b, c):
2     if a <= b and a <= c:
3         return a
4     elif b <= a and b <= c:
5         return b
6     else:
7         return c
8
9 # Print statements
10 print(find_min(3, 7, 5))
11 print(find_min(10, 2, 8))
12 print(find_min(4, 6, 9))
13
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Jashwanth\AI coding> & C:/Users/Jashwanth/AppData/Local/Microsoft/WindowsApps/python3.12.exe d:/Assignment-3.1.py
File "d:/Assignment-3.1.py", line 2
    print(find_min(3, 7, 5))
          ^
IndentationError: unexpected indent
PS C:\Users\Jashwanth\AI coding> & C:/Users/Jashwanth/AppData/Local/Microsoft/WindowsApps/python3.12.exe d:/Assignment-3.1.py
123
59
PS C:\Users\Jashwanth\AI coding> & C:/Users/Jashwanth/AppData/Local/Microsoft/WindowsApps/python3.12.exe d:/Assignment-3.1.py
True
False
PS C:\Users\Jashwanth\AI coding> & C:/Users/Jashwanth/AppData/Local/Microsoft/WindowsApps/python3.12.exe d:/Assignment-3.1.py
True
False
PS C:\Users\Jashwanth\AI coding> & C:/Users/Jashwanth/AppData/Local/Microsoft/WindowsApps/python3.12.exe d:/Assignment-3.1.py
5
2
4
```

OBSERVATION:

The examples clearly establish the comparison pattern needed to identify the smallest value. The AI model infers the requirement to handle equality cases correctly. Conditional logic is generated without relying on built-in functions.