

School of Computer Science and Artificial Intelligence

Lab Assignment # 10.2

Program	: B. Tech (CSE)
Specialization	: -
Course Title	: AI Assisted Coding
Course Code	: 23CS002PC304
Semester	II
Academic Session	: 2025-2026
Name of Student	: P.Eshwar
Enrollment No.	: 2403A51L26
Batch No.	51
Date	: 06/02/26

Submission Starts here**Screenshots:****Task Description -1Task:**(Error Detection and Correction)**Prompt:**

Use AI to analyze a Python script and correct all syntax and logical errors.

Sample Input Code:

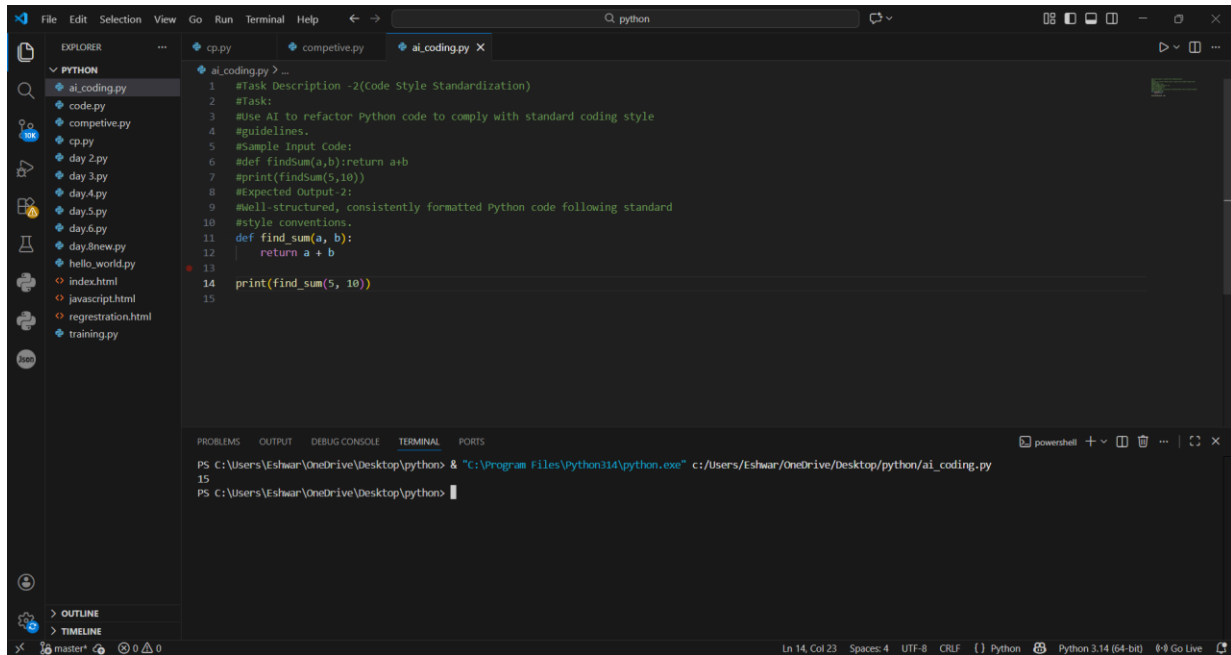
```
def calculate_total(nums)

sum = 0

for n in nums

sum += n

return total
```



Task Description -2(Code Style Standardization)

Prompt:

Use AI to refactor Python code to comply with standard coding style guidelines.

Sample Input Code:

```
def findSum(a,b):return a+b

print(findSum(5,10))
```

The screenshot shows a Visual Studio Code editor window with a Python file named `ai_coding.py` open. The file contains a function `calculate_total` that takes a list of numbers and returns their sum. The function is called with the list `[1, 2, 3, 4, 5]` and the output is `15`. The terminal window at the bottom shows the command `python ai_coding.py` being executed and the output `15`.

```

1  # Description: (Error Detection and Correction)
2
3  # analyze a Python script and correct all syntax and logical
4
5
6  def calculate_total(numbers):
7      total = 0
8      for num in numbers:
9          total += num
10     return total
11
12     # Original code had an indentation error with the 'return total' statement,
13     # which was outside the function. It has been moved inside the function to ensure it executes correctly.
14     # Original code did not have any syntax errors, but the logical error was
15     # the 'return total' statement was not part of the function, which would have caused a NameError when trying to access 'total' outside the function.
16     # usage:
17     # calculate_total([1, 2, 3, 4, 5])
18     # Output: 15
19     # Example: calculate_total([1, 2, 3, 4, 5])
20

```

```

PS C:\Users\Eshwar\OneDrive\Desktop\python> python ai_coding.py
Total: 15
PS C:\Users\Eshwar\OneDrive\Desktop\python> python ai_coding.py
Total: 15
PS C:\Users\Eshwar\OneDrive\Desktop\python>

```

Task Description -3(Code Clarity Improvement)

Prompt:

Use AI to improve code readability without changing its functionality.

Sample Input Code:

```

def f(x,y):

return x-y*2

print(f(10,3))

```

```

1  #Task Description -3(Code Clarity Improvement)
2  #Task:
3  #Use AI to improve code readability without changing its functionality.
4  #Sample Input Code:
5  #def f(x,y):
6  #return x-y*2
7  #print(f(10,3))
8  #Expected Output-3:
9  #Python code rewritten with meaningful function and variable names,
10 #proper indentation, and improved clarity.
11 def calculate_difference(x, y):
12     return x - y * 2
13 result = calculate_difference(10, 3)
14 print("The result of the calculation is:", result)
15
16

```

```

PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe" c:\Users\Eshwar\OneDrive\Desktop\python\ai_coding.py
The result of the calculation is: 4
PS C:\Users\Eshwar\OneDrive\Desktop\python>

```

Task Description -4(Structural Refactoring)

prompt:

Use AI to refactor repetitive code into reusable functions.

Sample Input Code:

```
print("Hello Ram")
```

```
print("Hello Sita")
```

```
print("Hello Ravi")
```

```

1 #Task Description -4(Structural Refactoring)
2 #Task:
3 #Use AI to refactor repetitive code into reusable functions.
4 #Sample Input Code:
5 #print("Hello Ram")
6 #print("Hello Sita")
7 #print("Hello Ravi")
8 #Expected Output 4:
9 #Modular Python code using reusable functions to eliminate repetition.
10 def greet(name):
11     print(f"Hello {name}")
12 names = ["Ram", "Sita", "Ravi"]
13 for name in names:
14     greet(name)
15
16

```

```

PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe" c:/Users/Eshwar/OneDrive/Desktop/python/ai_coding.py
Hello Ram
Hello Sita
Hello Ravi
PS C:\Users\Eshwar\OneDrive\Desktop\python>

```

Task Description -5(Efficiency Enhancement)

Prompt:

Use AI to optimize Python code for better performance.

Sample Input Code:

```

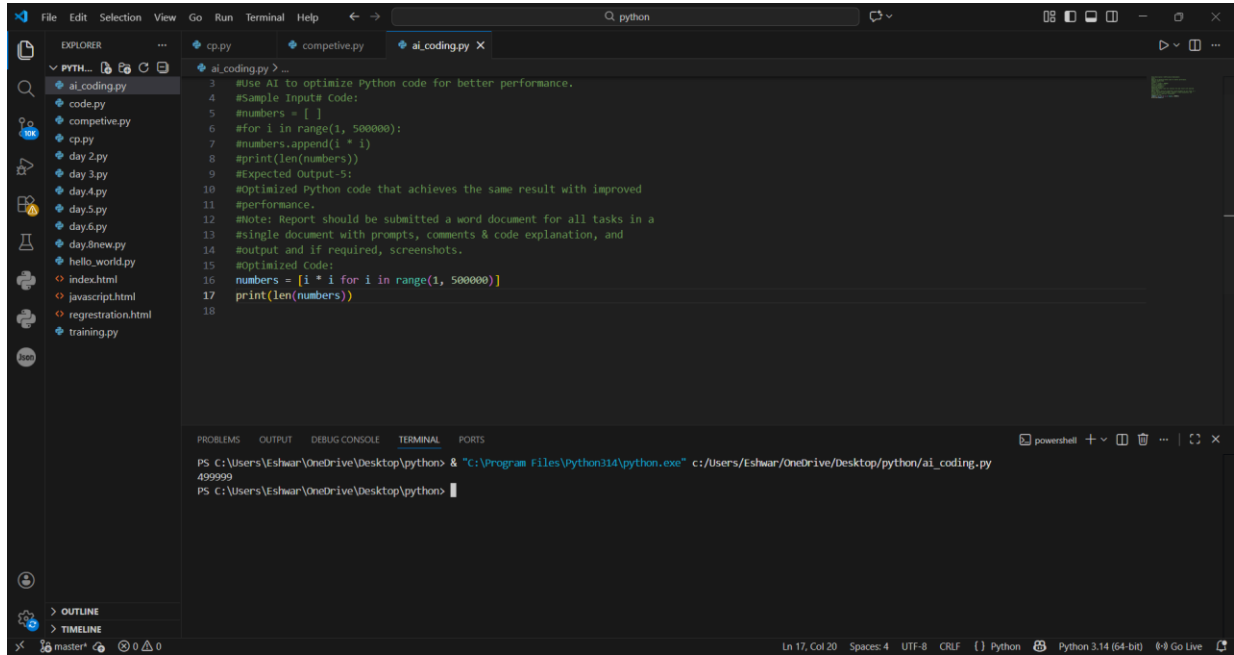
numbers = [ ]

for i in range(1, 500000):

numbers.append(i * i)

print(len(numbers))

```



```
File Edit Selection View Go Run Terminal Help python
EXPLORER
PYTHON ai_coding.py
ai_coding.py
code.py
competitive.py
cp.py
day 2.py
day 3.py
day 4.py
day 5.py
day 6.py
day 8new.py
hello_world.py
index.html
javascript.html
regrestation.html
training.py

3 #Use AI to optimize Python code for better performance.
4 #Sample Input# Code:
5 #numbers = [ ]
6 #for i in range(1, 500000):
7 #numbers.append(i * i)
8 #print(len(numbers))
9 #Expected Output-5:
10 #Optimized Python code that achieves the same result with improved
11 #performance.
12 #Note: Report should be submitted a word document for all tasks in a
13 #single document with prompts, comments & code explanation, and
14 #output and if required, screenshots.
15 #Optimized Code:
16 numbers = [i * i for i in range(1, 500000)]
17 print(len(numbers))
18

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Eshwar\OneDrive\Desktop\python> & "C:\Program Files\Python314\python.exe" c:\Users\Eshwar\OneDrive\Desktop\python\ai_coding.py
4999999
PS C:\Users\Eshwar\OneDrive\Desktop\python>

Ln 17, Col 20 Spaces: 4 UTF-8 CRLF Python Python 3.14 (64-bit) Go Live
```