

AI Assisted Coding

NAME:THOTA SHIRISHA

HTNO:2403A52077

BATCH:04

ASSIGNMENT:13.2

TASK 1:Task: Provide AI with the following redundant code and ask it to refactor

Python Code

```
def calculate_area(shape, x, y=0):  
if shape == "rectangle":  
return x * y  
elif shape == "square":  
return x * x  
elif shape == "circle":  
return 3.14 * x * x
```

PROMPT:

Task: Provide AI with the following redundant code and ask it to refactor

Python Code

```
def calculate_area(shape, x, y=0):  
if shape == "rectangle":  
return x * y  
elif shape == "square":  
return x * x  
elif shape == "circle":  
return 3.14 * x * x
```

CODE& OUTPUT:

```
1  import math
2
3  def calculate_area(shape, x, y=0):
4      shape = shape.lower()
5      area_functions = {
6          "rectangle": lambda x, y: x * y,
7          "square": lambda x, _: x * x,
8          "circle": lambda x, _: math.pi * x * x,
9      }
10
11     if shape not in area_functions:
12         raise ValueError("Unsupported shape")
13
14     return area_functions[shape](x, y)
15
16     # Example calls with output
17     print("Rectangle area:", calculate_area("rectangle", 5, 3))
18     print("Square area:", calculate_area("square", 4))
19     print("Circle area:", calculate_area("circle", 7))
20
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\kurapati pruthvi\AppData\Local\Programs\Microsoft VS Code> & "C:\Users\kurapati pruthvi\AppData\Local\Programs\Python\Python313\python.exe" "c:/Users/kurapati pruthvi/OneDrive/Documents/AI_7_9.py"
PS C:\Users\kurapati pruthvi\AppData\Local\Programs\Microsoft VS Code> & "C:\Users\kurapati pruthvi\AppData\Local\Programs\Python\Python313\python.exe" "c:/Users/kurapati pruthvi/OneDrive/Documents/AI_7_9.py"
Rectangle area: 15
Square area: 16
Circle area: 153.93804002589985
PS C:\Users\kurapati pruthvi\AppData\Local\Programs\Microsoft VS Code> |
```

- 1.Function Definition:** Defines a function [`calculate_area`](#) that takes the shape name and dimensions as input.
- 2.Shape Handling:** Converts the shape name to lowercase and defines a dictionary of lambda functions for calculating areas of different shapes.
- 3.Area Calculation:** If the shape is supported, it retrieves the corresponding lambda function and calculates the area using the provided dimensions.
- 4.Error Handling:** If the shape is not supported, it raises a [`ValueError`](#).
- 5.Example Usage:** Calls the function with different shapes and dimensions, printing the calculated areas.

TASK2: Legacy function without proper error handling

Python Code

```
def read_file(filename):  
f = open(filename, "r")  
data = f.read()  
.close()  
return data
```

PROMPT:

Legacy function without proper error handling

Python Code

```
def read_file(filename):  
f = open(filename, "r")  
data = f.read()  
.close()  
return data
```

CODE AND OUTPUT:

C: > Users > kurapati pruthvi > OneDrive > Documents > AI_7_2.py > ...

```
1  def read_file_safe(filename):
2      try:
3          # The 'with' statement handles opening and automatically closing the file
4          with open(filename, "r") as f:
5              data = f.read()
6          return data
7      except FileNotFoundError:
8          # Handle the case where the file does not exist
9          print(f"ERROR: The file '{filename}' was not found.")
10         return None # Return None or raise a custom exception
11     except IOError as e:
12         # Handle general I/O errors (e.g., permission issues, reading errors)
13         print(f"ERROR: An I/O error occurred while reading '{filename}': {e}")
14         return None
15
16 # 1. Success case
17 # Create a dummy file for testing
18 with open("test_file.txt", "w") as f:
19     f.write("Hello, Legacy World!")
20 content = read_file_safe("test_file.txt")
21 if content is not None:
22     print(f"\n--- SUCCESS ---")
23     print(f"File content: {content}")
24
25 # 2. File not found case
26 print("\n--- ERROR: File Not Found ---")
27 content_missing = read_file_safe("non_existent_file.txt")
28 print(f"Result for missing file: {content_missing}")
29
30 # 3. Handling potential permission issues (conceptual example)
31 # In a real-world scenario, you might get an IOError if you try
32 # to read a file without the necessary permissions.
33 print("\n--- ERROR: General I/O (Conceptual) ---")
34 # If 'locked_file.dat' existed but had restricted permissions
35 # the function would print the I/O error message.
36 content_io_error = read_file_safe("locked_file.dat")
37 print(f"Result for IO error test: {content_io_error}")
38
39 import os
40 os.remove("test_file.txt")
```

OUTPUT:

```
--- SUCCESS ---  
File content: Hello, Legacy World!  
  
--- ERROR: File Not Found ---  
ERROR: The file 'non_existent_file.txt' was not found.  
Result for missing file: None  
  
--- ERROR: General I/O (Conceptual) ---  
ERROR: The file 'locked_file.dat' was not found.  
Result for IO error test: None  
PS C:\Users\kurapati pruthvi\AppData\Local\Programs\Microsoft VS Code> █
```

OBSERVATION:

- 1.read_file_safe: Defines a function to safely read files, handling errors.
- 2.Success**: Creates, writes to, and reads "test_file.txt", printing the content.
- 3.File Not Found**: Attempts to read "non_existent_file.txt", printing an error.
- 4.IOException (Conceptual)**: Attempts to read "locked_file.dat", demonstrating potential I/O error handling.
- 5.Cleanup**: Deletes "test_file.txt"

Task3: Provide this legacy class to AI for readability and modularity improvements:

Python Code

```
class Student:
```

```
def __init__(self, n, a, m1, m2, m3):
```

```
self.n = n
```

```
self.a = a
```

```
self.m1 = m1
```

```
self.m2 = m2
```

```
self.m3 = m3
```

```
def details(self):
```

```
print("Name:", self.n, "Age:", self.a)
```

```
def total(self):
```

```
return self.m1+self.m2+self.m3
```

comprehension

PROMPT: Provide this legacy class to AI for readability and modularity improvements:

Python Code

```
class Student:
```

```
def __init__(self, n, a, m1, m2, m3):
```

```
self.n = n
```

```
self.a = a
```

```
self.m1 = m1
```

```
self.m2 = m2
```

```
self.m3 = m3
```

```
def details(self):
```

```
print("Name:", self.n, "Age:", self.a)
```

```
def total(self):
```

```
return self.m1+self.m2+self.m3
```

CODE & OUTPUT:

```
C: > Users > kurapati pruthvi > OneDrive > Documents > AI_7.3.py > Student > get_average

1 class Student:
2     def __init__(self, name: str, age: int, marks: list[int]):
3         self.name = name
4         self.age = age
5         self.marks = marks # A list of marks (e.g., [85, 90, 92])
6     def get_details(self) -> str:
7         return f"Name: {self.name}, Age: {self.age}"
8     def get_total_marks(self) -> int:
9         return sum(self.marks)
10    def get_average(self) -> float:
11        return sum(self.marks) / len(self.marks)
12    def __str__(self) -> str:
13        return self.get_details()
14    # Example usage
15    if __name__ == "__main__":
16        # Create a student object with name, age, and list of 3 marks
17        student1 = Student("Alice", 17, [85, 90, 92])
18
19        # Display student details and calculated information
20        print(student1.get_details())
21        print("Total Marks:", student1.get_total_marks())
22        print("Average Marks:", student1.get_average())
23
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\kurapati pruthvi\AppData\Local\Programs\Microsoft VS Code> & "C:\Users\kurapati pruthvi\AppData\Local\Programs\Python\Python313\python.exe" "c:/Users/kurapati pruthvi/OneDrive/Documents/AI_7.3.py"
PS C:\Users\kurapati pruthvi\AppData\Local\Programs\Microsoft VS Code> & "C:\Users\kurapati pruthvi\AppData\Local\Programs\Python\Python313\python.exe" "c:/Users/kurapati pruthvi/OneDrive/Documents/AI_7.3.py"
Name: Alice, Age: 17
Total Marks: 267
Average Marks: 89.0
PS C:\Users\kurapati pruthvi\AppData\Local\Programs\Microsoft VS Code> & "C:\Users\kurapati pruthvi\AppData\Local\Programs\Python\Python313\python.exe" "c:/Users/kurapati pruthvi/OneDrive/Documents/AI_7.3.py"
Name: Alice, Age: 17
Total Marks: 267
Average Marks: 89.0
PS C:\Users\kurapati pruthvi\AppData\Local\Programs\Microsoft VS Code>
```

OBSERVATION:

1.Class Definition: Defines a Student class to represent student data.

- 1.__init__ **(Constructor):** Initializes a Student object with a name (string), age (integer), and marks (list of integers).
- 2.get_details **Method:** Returns a formatted string containing the student's name and age.
- 3.get_total_marks **Method:** Calculates and returns the sum of the student's marks.
- 4.get_average **Method:** Calculates and returns the average of the student's marks.
- 5.**Example Usage:** Creates a Student object, then calls the methods to display the student's details, total marks, and average marks.
- 6.**Class Definition:** Defines a Student class to represent student data.
- 7.**Class Definition:** Defines a Student class to represent student data.

Task4: Refactor this inefficient loop with AI help

Python Code

```
nums = [1,2,3,4,5,6,7,8,9,10]
```

```
squares = []
```

```
for i in nums:
```

```
squares.append(i * i)
```

PROMPT:

Task: Refactor this inefficient loop with AI help

Python Code

```
nums = [1,2,3,4,5,6,7,8,9,10]
```

```
squares = []
```

```
for i in nums:
```

```
squares.append(i * i)
```

Expected Output: AI suggested a list comprehension

CODE:

```
1  nums = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
2  squares = [i * i for i in nums]
3  print(squares)
4
```

OUTPUT:

```
hon\Python313\python.exe" "c:/Users/kurapati pruthvi/OneDrive/Documents/AI-7.4
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
PS C:\Users\kurapati pruthvi\AppData\Local\Programs\Microsoft VS Code>
```

OBSERVATION:

1.num = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]: Creates a list named num containing the integers from 1 to 10.

2.squares = [i * i for i in num]: This is a list comprehension. It creates a new list called squares.

3.for i in num: The list comprehension iterates through each element i in the num list.

4.i * i: For each element i, it calculates the square of i (i.e., i multiplied by itself).

5.List Creation: The results of these calculations are collected into a new list called squares.

6.print(squares): Prints the squares list to the console. The output will be [1, 4, 9, 16, 25, 36, 49, 64, 81, 100].