ASSIGNMENT(13.2)

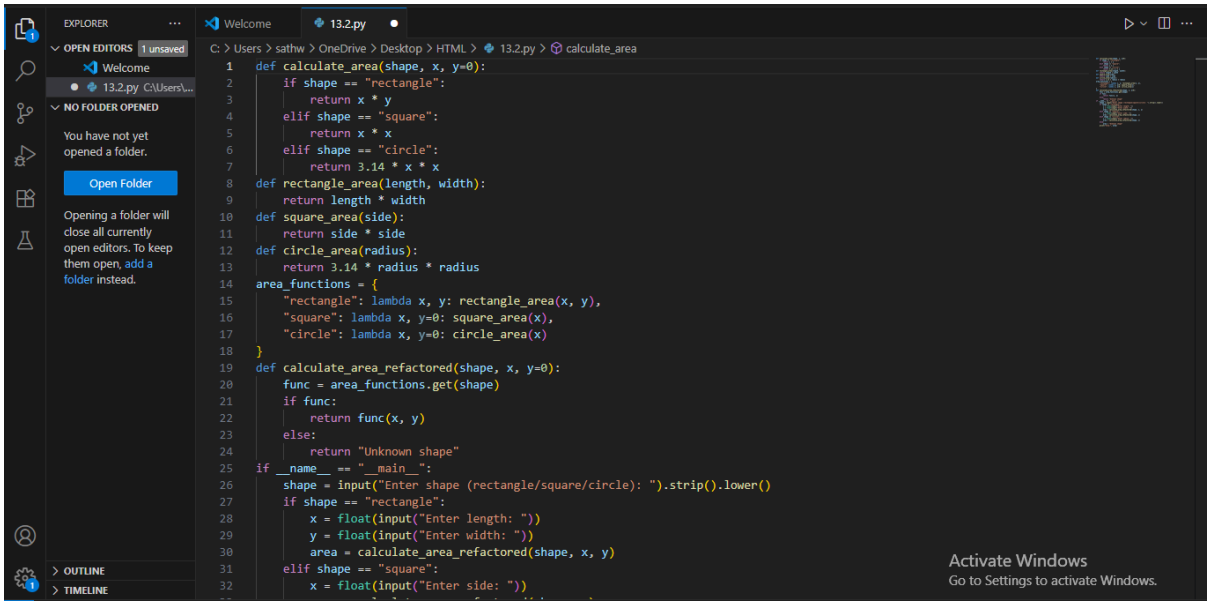
NAME:SINDHUJA.M

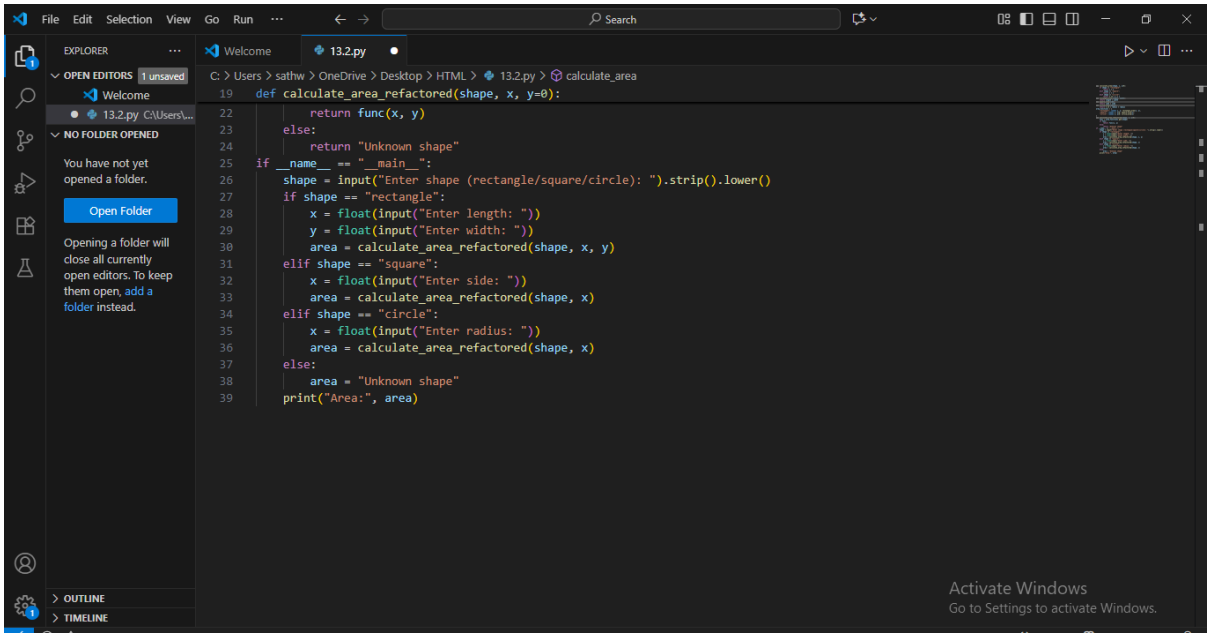
2403A52060

BATCH-03

TASK-01

PROMPT: *Write a code 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*





OUTPUT 1:

Enter shape (rectangle/square/circle): square

Enter side: 6

Area: 36.0

OUTPUT 2:

Enter shape (rectangle/square/circle): rectangle

Enter length: 12

Enter width: 56

Area: 672.0

OUTPUT 3:

Enter shape (rectangle/square/circle): CIRCLE

Enter radius: 23

Area: 1661.06

EXPLANATION:

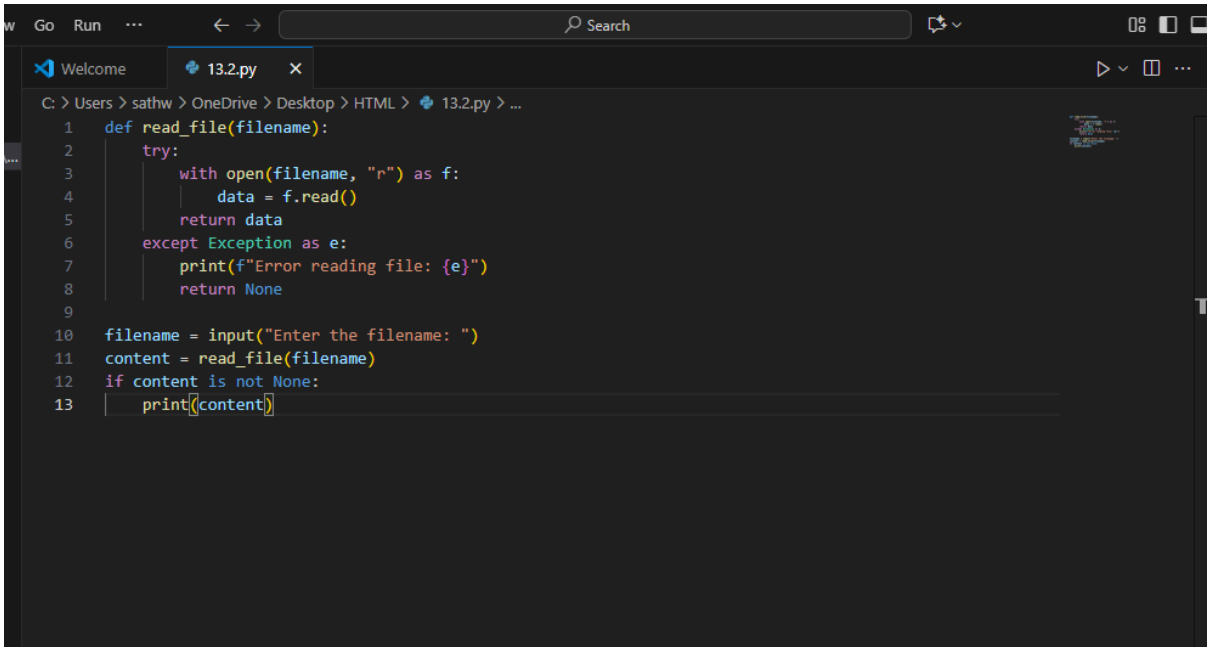
- The first function, `calculate_area`, uses simple if-elif statements to compute the area for each shape.
- Then, there are separate functions for each shape: `rectangle_area`, `square_area`, and `circle_area`.
- The `area_functions` dictionary maps shape names to their corresponding area calculation functions using lambdas.
- The `calculate_area_refactored` function looks up the correct function from the dictionary and calls it with the provided arguments.

TASK-02

PROMPT: write a python code that has Legacy function without proper error handling

Python Code

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



OUTPUT: This is some example content. Error: File not found at non_existent_file.txt

EXPLANATION:

The script then asks the user for a filename, calls read_file, and prints the file content if reading was successful.

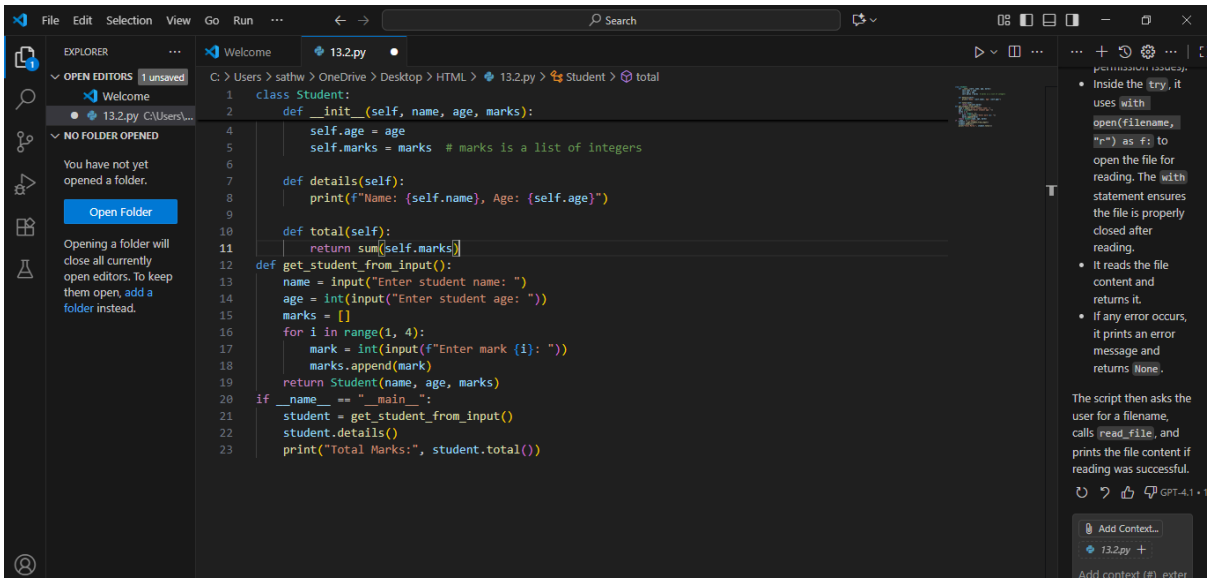
TASK-03

PROMPT: Write a code related to legacy class 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



OUTPUT:

Enter student name: alice

Enter student age: 21

Enter mark 1: 45

Enter mark 2: 78

Enter mark 3: 90

Name: alice, Age: 21

Total Marks: 213

EXPLANATION:

- The `details` method prints the student's name and age.
- The `total` method returns the sum of the student's marks.
- The `get_student_from_input` function prompts the user to enter the student's name, age, and three marks, then creates and returns a `Student` object.

- In the main block, the program gets student details from user input, prints the student's name and age, and then prints the total marks.

TASK-04

PROMPT: *Write a code to Refactor this inefficient loop*

Python Code

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

squares = []

for i in nums:

*squares.append(i * i)*

OUTPUT:

Enter numbers separated by spaces: 1 6 7 3 9 4 23

[1, 36, 49, 9, 81, 16, 529]

EXPLANATION:

- `input("Enter numbers separated by spaces: ")` prompts the user to enter numbers.
- `.split()` splits the input string into a list of strings.
- `map(int, ...)` converts each string to an integer.
- `list(...)` creates a list of those integers and stores it in `nums`.

- `[i * i for i in nums]` creates a new list called `squares` containing the square of each number in `nums`.
- `print(squares)` displays the list of squared numbers.