# AI Assisted Coding Lab – 2.3

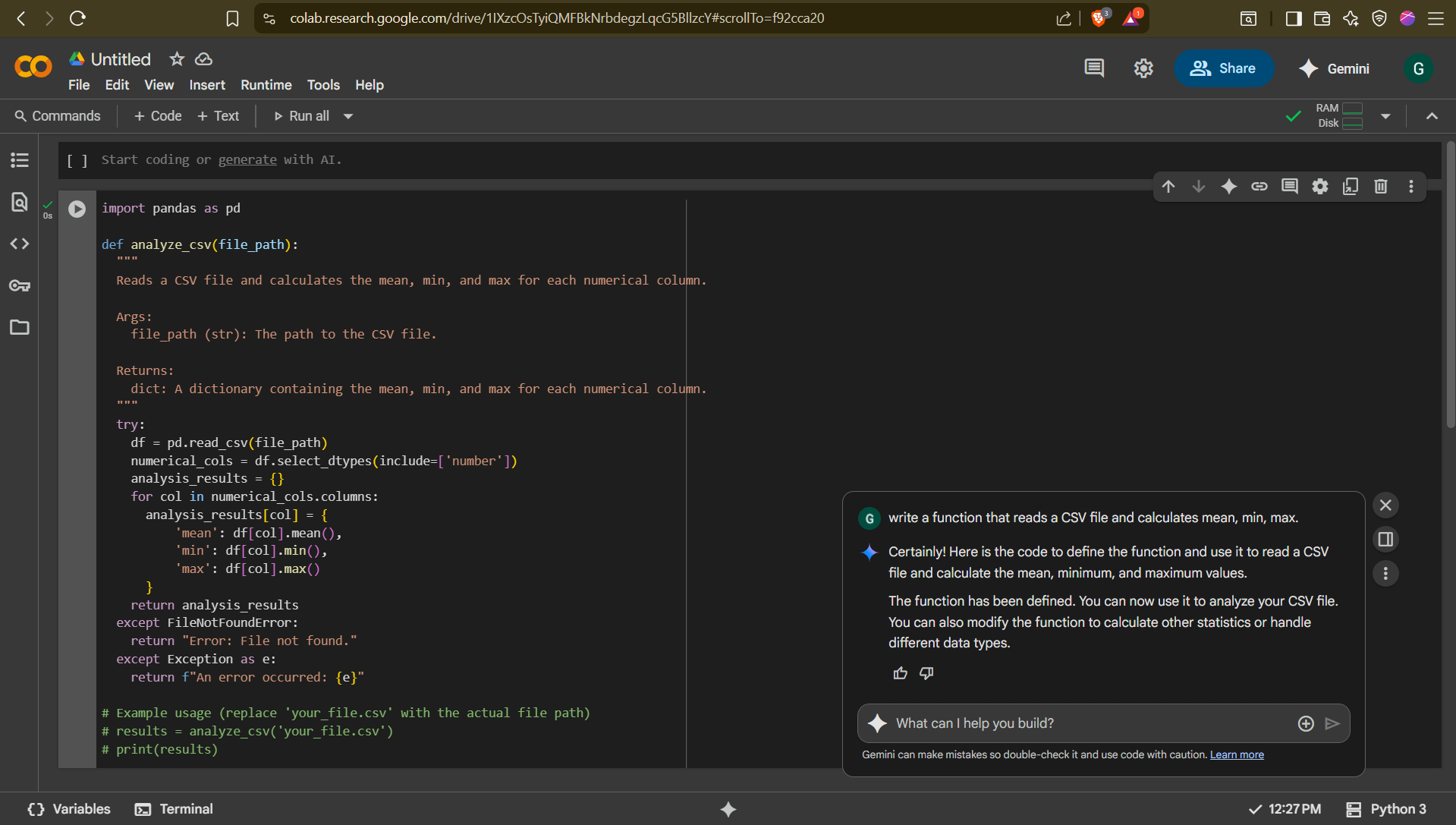
**Name:** Gattu Varshith

**Roll Number:** 2403A510E3

**Batch:** 24BTCAICSB05

**#TASK DESCRIPTION 1:**

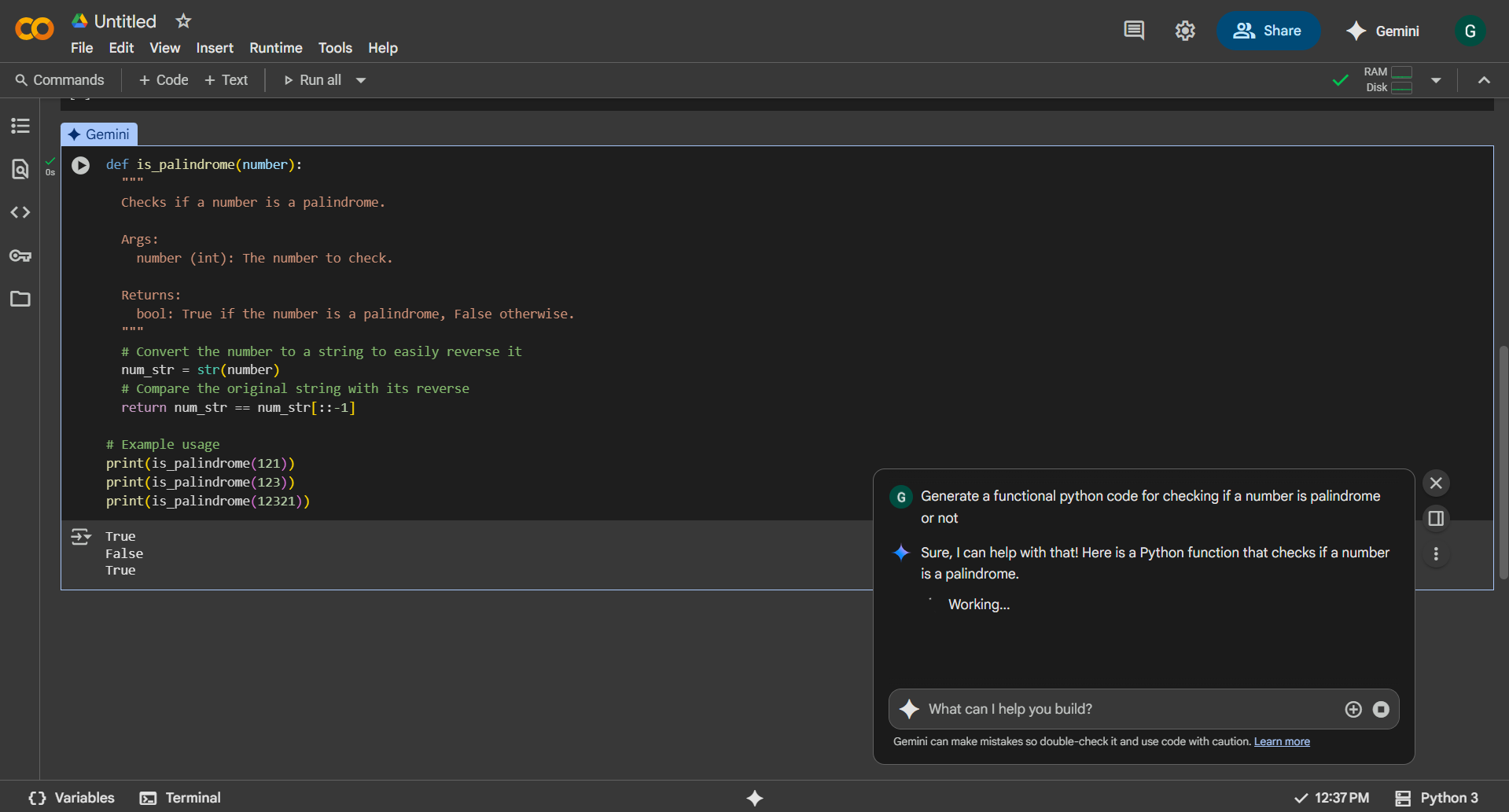
Use Google Gemini in Colab to write a function that reads a CSV file and calculates mean, min, max.



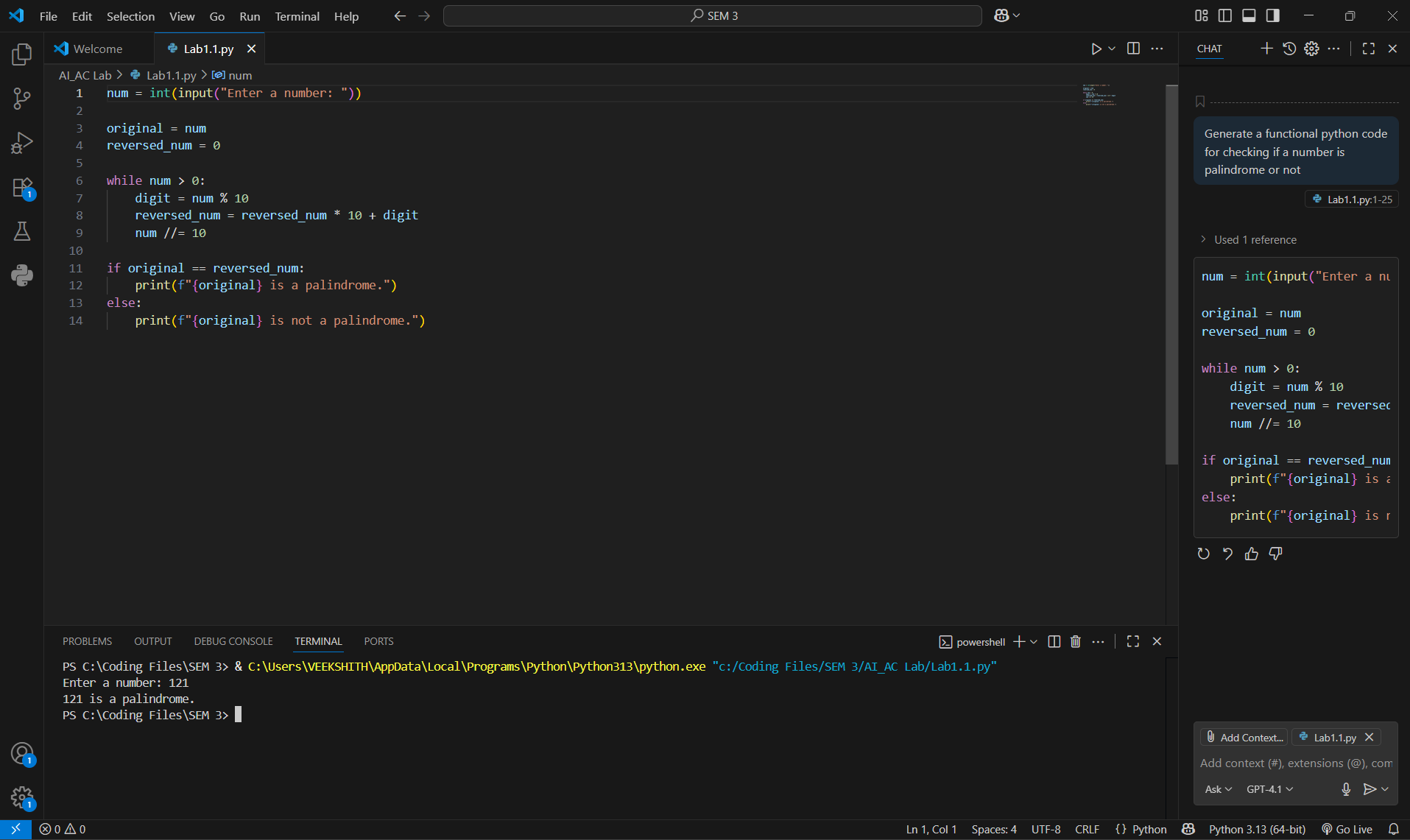
**#TASK DESCRIPTION 2:**

Compare Gemini and Copilot outputs for a palindrome check function.

Gemini’s response:



Co-pilot’s response:



Observations:

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Gemini** | **Copilot** |
| Code Style | Long, detailed, with explanations and edge case checks | Short, direct, no extra checks |
| Comments | Lots of docstrings and notes | Minimal comments |
| Input Handling | Checks for negatives and non-integers | No checks |
| Output | Looks like a tutorial | Ready-to-use snippet |
| Extra Info | No “How it works” section | Adds a short “How it works” |

**#TASK DESCRIPTION 3:**

Ask Gemini to explain a Python function (to calculate area of various shapes) line by line.

Gemini’s response:

Here is a Python function that calculates the area of various shapes.

```python

import math

def calculate\_area(shape, \*\*kwargs):

"""

Calculates the area of various shapes based on the provided shape type and dimensions.

Args:

shape (str): The type of shape (e.g., 'circle', 'rectangle', 'triangle').

\*\*kwargs: Keyword arguments for the dimensions of the shape.

For 'circle': radius

For 'rectangle': length, width

For 'triangle': base, height

Returns:

float: The calculated area.

str: An error message if the shape is not supported or dimensions are missing.

"""

shape = shape.lower()

if shape == 'circle':

if 'radius' in kwargs:

radius = kwargs['radius']

return math.pi \* radius\*\*2

else:

return "Error: Missing radius for circle."

elif shape == 'rectangle':

if 'length' in kwargs and 'width' in kwargs:

length = kwargs['length']

width = kwargs['width']

return length \* width

else:

return "Error: Missing length or width for rectangle."

elif shape == 'triangle':

if 'base' in kwargs and 'height' in kwargs:

base = kwargs['base']

height = kwargs['height']

return 0.5 \* base \* height

else:

return "Error: Missing base or height for triangle."

else:

return f"Error: Unsupported shape '{shape}'."

```

-----

### Line-by-Line Breakdown

\* \*\*`import math`\*\*: This line imports Python's built-in \*\*`math`\*\* module. This module provides access to mathematical functions and constants, like \*\*`math.pi`\*\* (the value of Pi), which is needed to calculate the area of a circle.

\* \*\*`def calculate\_area(shape, \*\*kwargs):`\*\*: This line defines the function named \*\*`calculate\_area`\*\*. It takes two arguments:

\* \*\*`shape`\*\*: A required positional argument that expects a string representing the type of shape (e.g., `'circle'`).

\* \*\*`\*\*kwargs`\*\*: This is a special syntax that collects all other \*\*keyword arguments\*\* into a dictionary. For example, if you call `calculate\_area('circle', radius=5)`, \*\*`kwargs`\*\* will be the dictionary `{'radius': 5}`. This makes the function flexible, as it can accept different sets of parameters for different shapes.

\* \*\*`"""..."""`\*\*: This is a \*\*docstring\*\*. It's a special type of multi-line string used to explain what the function does, its arguments (`Args`), and what it returns (`Returns`). It's good practice for making code easy to understand.

\* \*\*`shape = shape.lower()`\*\*: This line converts the input `shape` string to all lowercase letters. This makes the function's checks \*\*case-insensitive\*\*, so `'Circle'` and `'circle'` are treated the same way, preventing potential errors.

\* \*\*`if shape == 'circle':`\*\*: This is the start of an \*\*`if-elif-else`\*\* block. This line checks if the lowercase shape string is exactly equal to `'circle'`.

\* \*\*`if 'radius' in kwargs:`\*\*: This nested `if` statement checks if the key `'radius'` exists within the \*\*`kwargs`\*\* dictionary. This is a crucial check to ensure the necessary dimension is provided before attempting the calculation.

\* \*\*`radius = kwargs['radius']`\*\*: If the `radius` key exists, its value is retrieved from the `kwargs` dictionary and assigned to the variable `radius`.

\* \*\*`return math.pi \* radius\*\*2`\*\*: This line calculates the area of a circle using the formula $A = \\pi r^2$ and \*\*returns\*\* the result. The `\*\*2` is the exponentiation operator, so `radius\*\*2` calculates the radius squared.

\* \*\*`else: return "Error: Missing radius for circle."`\*\*: If the `if 'radius' in kwargs:` check fails, this `else` block is executed, and it returns a descriptive error message.

\* \*\*`elif shape == 'rectangle':`\*\*: This line is part of the `if-elif-else` block. It's an "else if" that checks for the `'rectangle'` shape if the first `if` condition was false.

\* \*\*`if 'length' in kwargs and 'width' in kwargs:`\*\*: This checks for both required dimensions for a rectangle. The \*\*`and`\*\* operator ensures that both keys are present in the `kwargs` dictionary.

\* \*\*`return length \* width`\*\*: If both dimensions are present, this line calculates the area using the formula $A = l \\times w$ and returns the result.

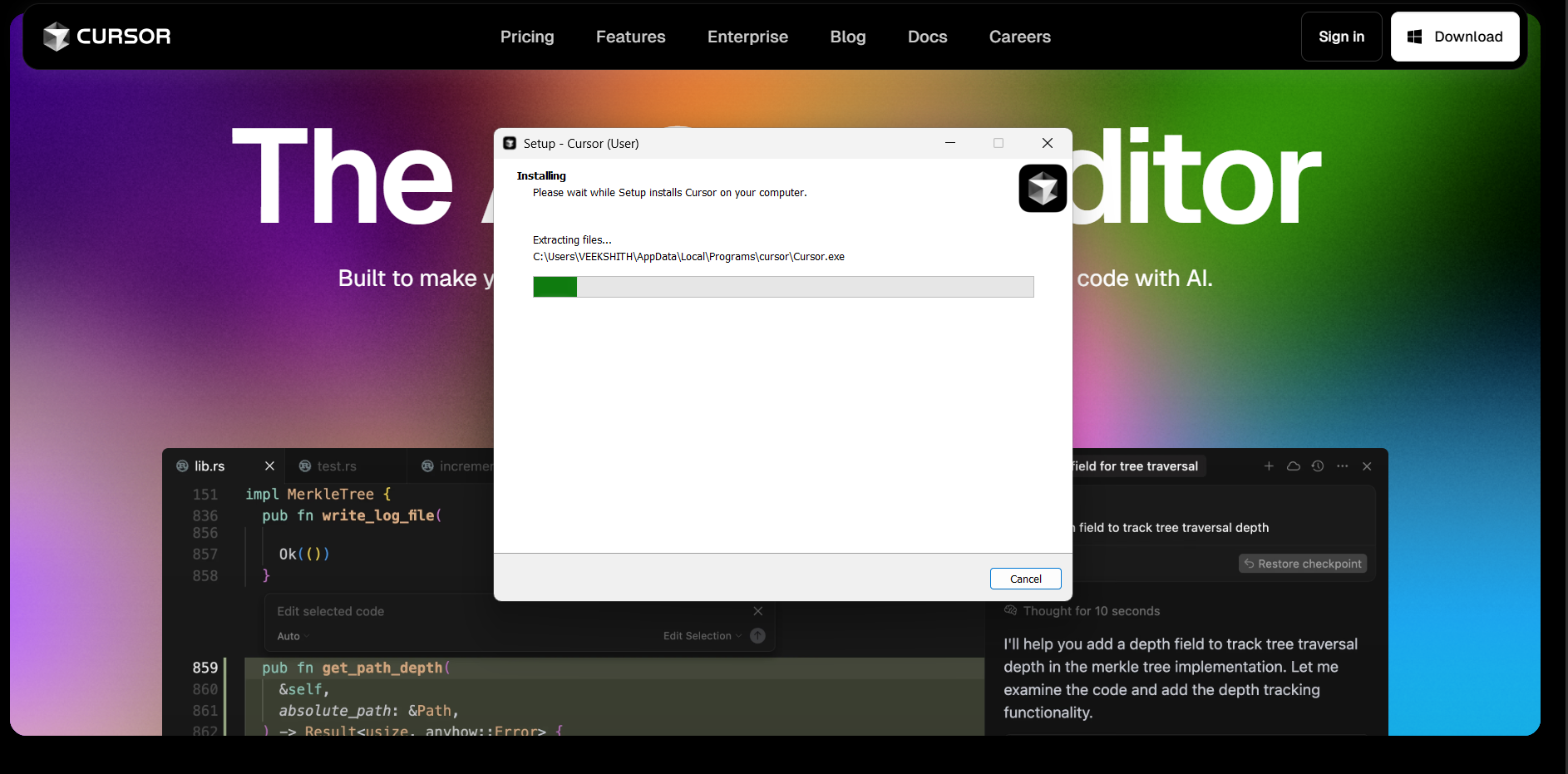
\* \*\*`elif shape == 'triangle':`\*\*: This line checks for the `'triangle'` shape.

\* \*\*`if 'base' in kwargs and 'height' in kwargs:`\*\*: This checks for the required `'base'` and `'height'` dimensions for a triangle.

\* \*\*`return 0.5 \* base \* height`\*\*: If both dimensions are present, this line calculates the area using the formula $A = \\frac{1}{2}bh$ and returns the result.

\* \*\*`else: return f"Error: Unsupported shape '{shape}'."`\*\*: This is the final `else` block of the main `if-elif-else` structure. If none of the previous `if` or `elif` conditions were met (meaning the shape is not `'circle'`, `'rectangle'`, or `'triangle'`), this code returns a formatted string (`f-string`) with a specific error message.

**#Task Description 4:**



**#Task Description 5:**

Student need to write code to calculate sum of add number and even numbers in the  
list

