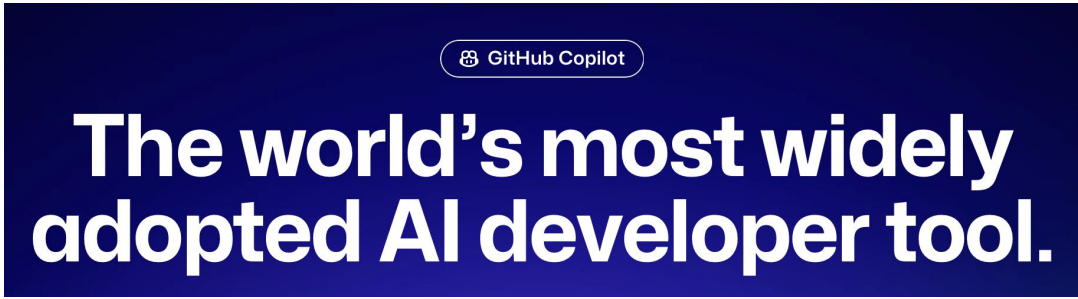
Coding with AI tools

For hardworking foxes

Github Copilot

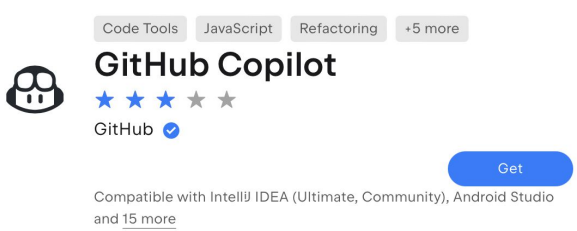
- Build's on OpenAI Codex
- Trained on Github hosted code
- Has lots of features (besides autocomplete)
- It is not the same thing as Microsoft Copilot



Github Copilot - Integrations

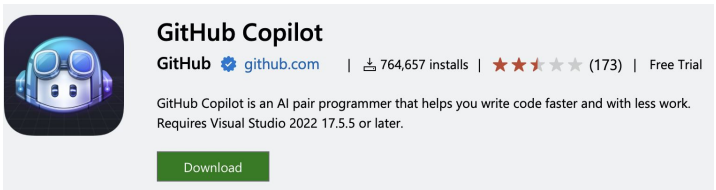
- VSCode

- Jetbrains IDEs



- (N)Vim

- (Plain Old) Visual Studio

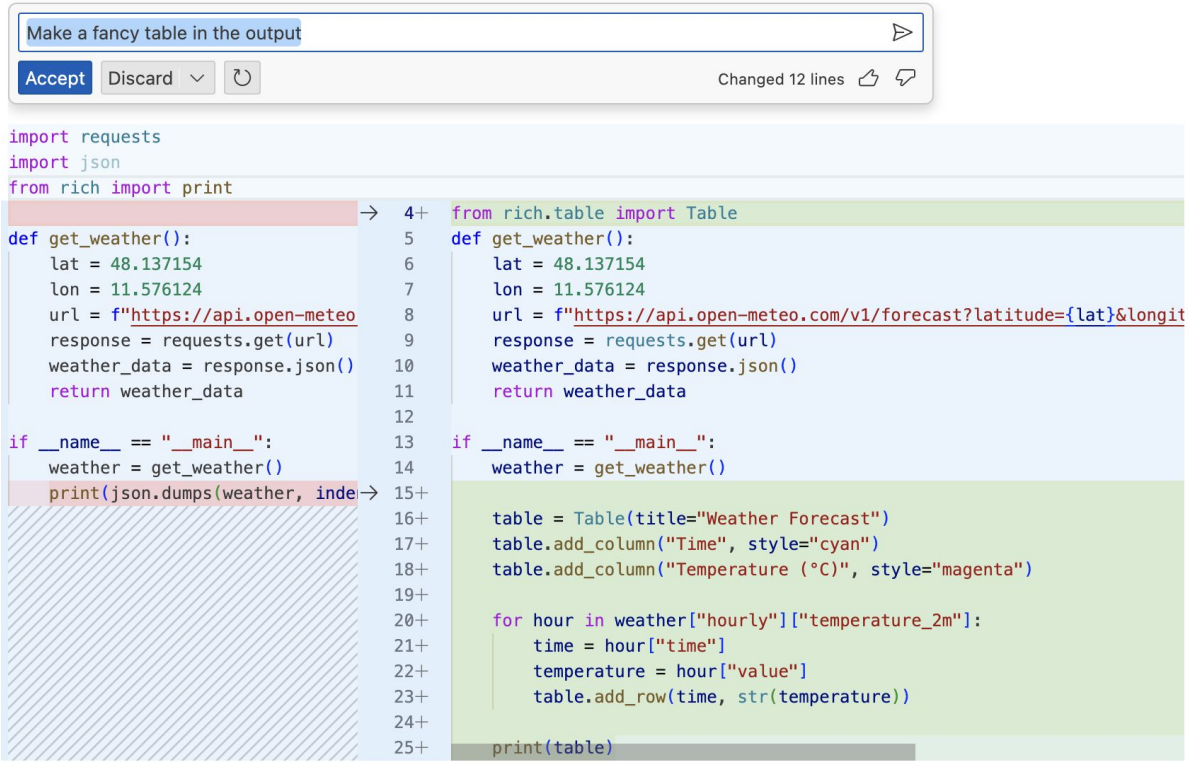


Github Copilot - Features

```python
import requests
import json

def get_weather():
    lat = 48.137154
    lon = 11.576124
    url = f"https://api.open-meteo.com/v1/forecast?latitude={lat}&longitude={lon}&current=temperature_2m,
    precipitation_1h,wind_speed_10m,wind_gusts_10m,wind_direction_10m,cloud_cover_low,cloud_cover_medium,
    cloud_cover_high,cloud_cover_total,weathercode&hourly=temperature_2m,precipitation_1h,wind_speed_10m,
    wind_gusts_10m,wind_direction_10m,cloud_cover_low,cloud_cover_medium,cloud_cover_high,cloud_cover_total,
    weathercode&daily=temperature_2m_max,temperature_2m_min,precipitation_probability,precipitation_1h_max,
    wind_speed_10m_max,wind_gusts_10m_max,wind_direction_10m_max,cloud_cover_low_max,cloud_cover_medium_max,
    cloud_cover_high_max,cloud_cover_total_max"
    response = requests.get(url)
    weather_data = response.json()
    return weather_data

if __name__ == "__main__":
    weather = get_weather()
    print(json.dumps(weather, indent=4))
```

Github Copilot - Features

Inline refactoring



Github Copilot - Features

Rexhaif

`@terminal` How to get my current location by IP?

GitHub Copilot used @terminal

Used 1 reference

You can use the `curl` command with the `ipinfo.io` service to get your current location by IP. Here is the command:

```
curl ipinfo.io
```

This will return a JSON object with your IP, city, region, country, and coordinates.

Rexhaif

`@workspace` `/explain` Explain how sample_1.py works

GitHub Copilot used @workspace

Used 1 reference

The `sample_1.py` script is designed to fetch weather data for a specific location (latitude and longitude) from the Open-Meteo API and return it as a JSON object.

Here's a step-by-step breakdown:

1. The script imports the necessary modules: `requests` for making HTTP requests, `json` for handling JSON data, and `rich` for creating rich text and beautiful formatting in the terminal.

2. The `get_weather` function is defined. This function fetches weather data from the Open-Meteo API.

3. Inside the `get_weather` function, the latitude (`lat`) and longitude (`lon`) for the location are defined. These are currently hard-coded to represent Munich, Germany.

4. A URL is constructed for the API request. This URL includes the latitude and longitude, and specifies that the hourly

Github Copilot - Features

Rexhaif

@workspace /explain What is the location at the coordinates in the file?

GitHub Copilot used @workspace

Used 1 reference

The coordinates `48.137154, 11.576124` in the file represent the city of Munich, Germany.

Latitude `48.137154` and Longitude `11.576124` point to a location in the central part of Munich, near the Marienplatz, which is a central square in the city center of Munich.

Rexhaif

/fix make code look better

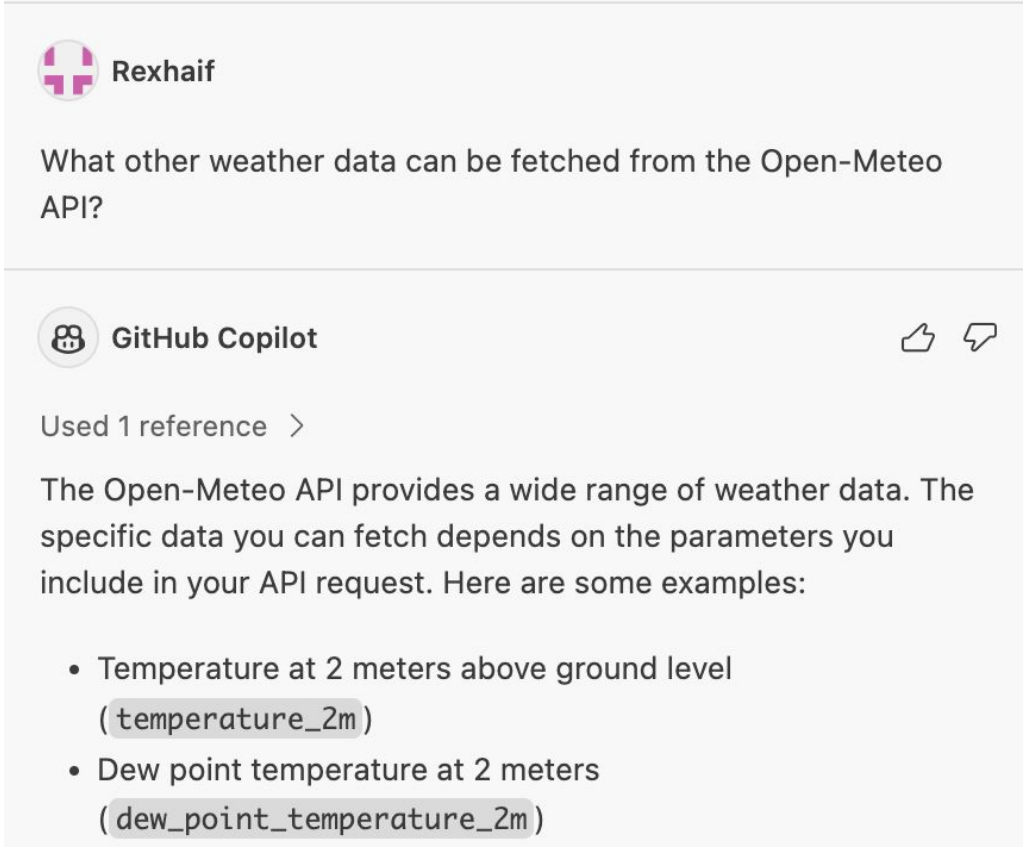
GitHub Copilot

Used 1 reference

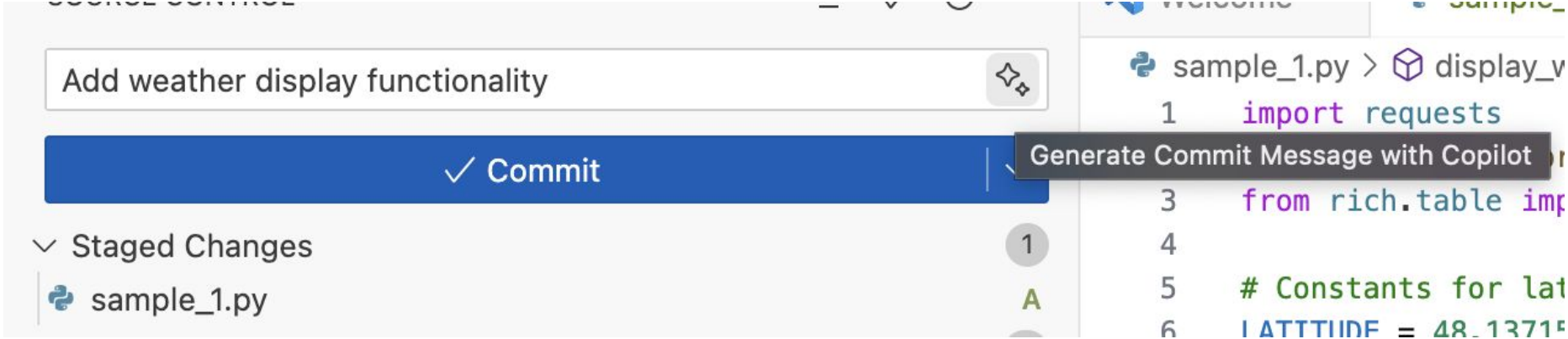
Here's a refactored version of your code. I've added some comments for clarity, used constants for the latitude and longitude, and added a function to create and display the table.

```
import requests
from rich import print
from rich.table import Table

# Constants for latitude and longitude
LATITUDE = 48.137154
LONGITUDE = 11.576124

def get_weather(lat, lon):
    """Fetch weather data from Open-Meteo API."""
    url = f"https://api.open-meteo.com/v1/forecast?lati
    response = requests.get(url)
    return response.json()

def display_weather(weather_data):
    """Create and display a table with weather data."""
    table = Table(title="Weather Forecast")
    # Add columns to the table
    table.add_column("Date", style="cyan")
```

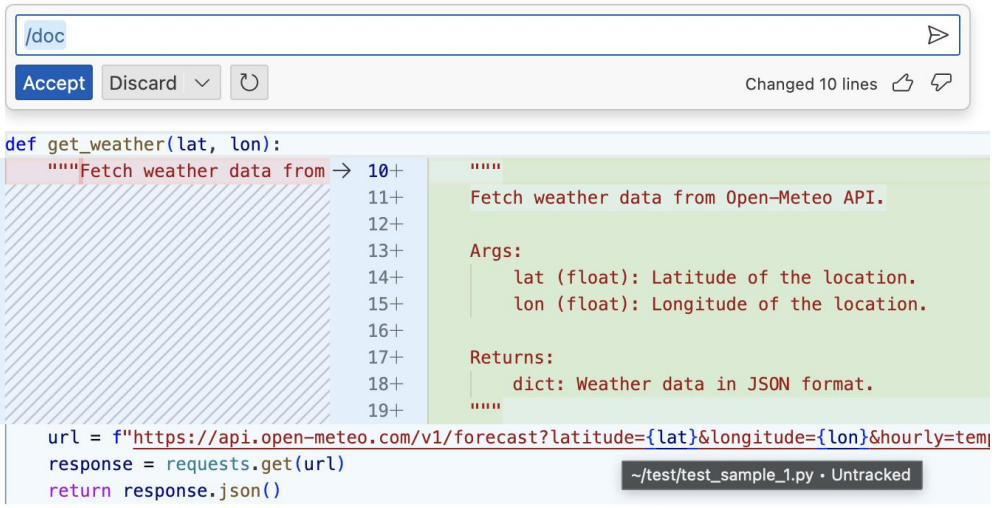
Github Copilot - Features

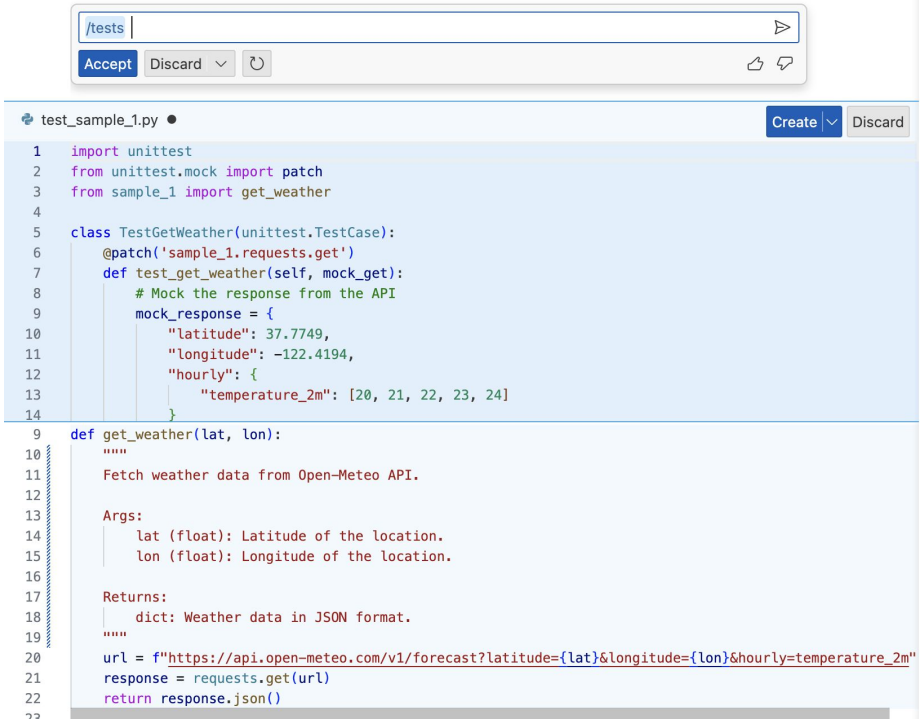


Github Copilot - Features



Github Copilot - Features





Github Copilot - Problems

Github Copilot - Problems

1. Cost
    a. 10$ per month for individuals, 19$ per month(per user) for teams, 39$ per month for Enterprise stuff
    b. 30 day free trial
    c. Free for students with github student pack:
        i. https://education.github.com/pack/offers

Github Copilot - Problems

2. Incorrect code generated

```python
import requests
import json

def get_weather():
    lat = 48.137154
    lon = 11.576124
    url = f"https://api.open-meteo.com/v1/forecast?latitude={lat}&longitude={lon}&current=temperature_2m,
    precipitation_1h,wind_speed_10m,wind_gusts_10m,wind_direction_10m,cloud_cover_low,cloud_cover_medium,
    cloud_cover_high,cloud_cover_total,weathercode&hourly=temperature_2m,precipitation_1h,wind_speed_10m,
    wind_gusts_10m,wind_direction_10m,cloud_cover_low,cloud_cover_medium,cloud_cover_high,cloud_cover_total,
    weathercode&daily=temperature_2m_max,temperature_2m_min,precipitation_probability,precipitation_1h_max,
    wind_speed_10m_max,wind_gusts_10m_max,wind_direction_10m_max,cloud_cover_low_max,cloud_cover_medium_max,
    cloud_cover_high_max,cloud_cover_total_max"
    response = requests.get(url)
    weather_data = response.json()
    return weather_data

if __name__ == "__main__":
    weather = get_weather()
    print(json.dumps(weather, indent=4))
```

```
❯ python sample_1.py
{
    "reason": "Data corrupted at path ''. Cannot initialize SurfaceAndPressureVariable<ForecastSurfaceVariable, ForecastPressureVariable> fro
m invalid String value precipitation_1h.",
    "error": true
}
```

Github Copilot - Problems

3. Identical copies of existing code

Does the GPL require that source code of modified versions be posted to the public?
(#GPLRequireSourcePostedPublic)

The GPL does not require you to release your modified version, or any part of it. You are free to make modifications and use them privately, without ever releasing them. This applies to organizations (including companies), too; an organization can make a modified version and use it internally without ever releasing it outside the organization.

But *if* you release the modified version to the public in some way, the GPL requires you to make the modified source code available to the program's users, under the GPL.

Thus, the GPL gives permission to release the modified program in certain ways, and not in other ways; but the decision of whether to release it is up to you.

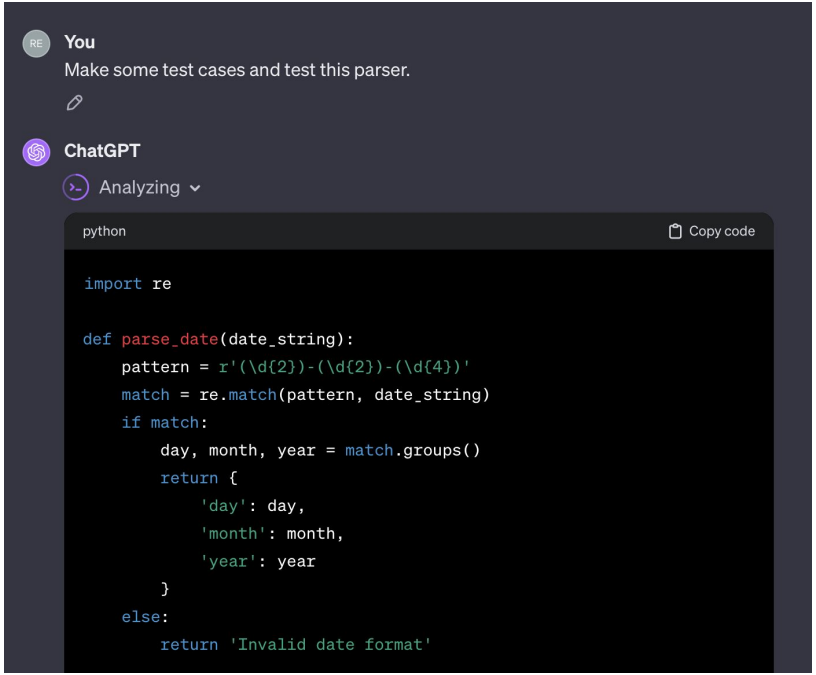
Other tools

Codeium - Copilot for free

- https://codeium.com/
- Has free tier for individuals
- Doesn't train on permission-less code, licensed with GPL => no legal risks
- Works for a lot more editors/IDEs:
  - Jupyterlab/Emacs
  - SublimeText/Eclipse
  - Colaboratory
- You do need to make an account

ChatGPT+ with Data Analysis tool

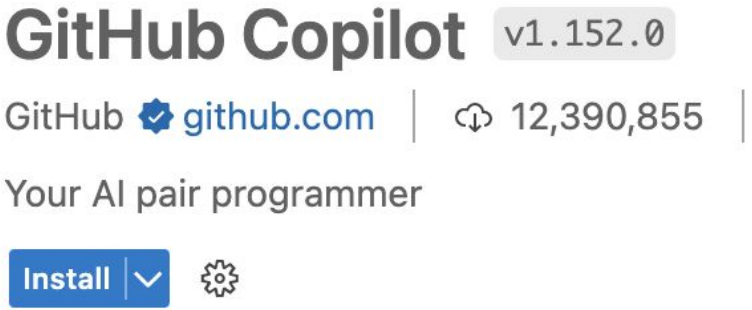
- A python environment that can be used by GPT-4
- GPT-4 is able to process output and errors, fix and re-run code
- Do not have internet access
- Does have some external libraries (ask ChatGPT to print installed libs in it's environment)

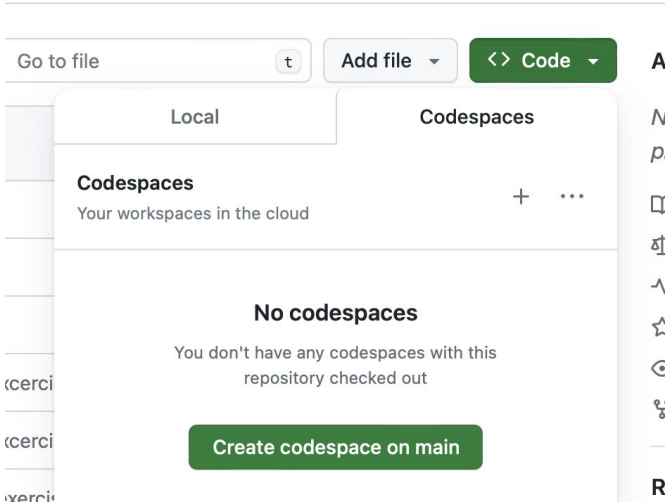


Exercises

#1 Set up your environment

- **Option 1:**
  - Go to my repository: https://github.com/Rexhaif/retreat-tutorial-2024/tree/main
  - Click on “Create codespace on main”
  - Install and set-up Copilot(or codeium extensions)
  - Install Python extensions (optional)





#1 Set up your environment

- **Option 2:**
  - Use your favorite IDE (PyCharm, NeoVim,, etc.)
  - Set up copilot or codeium extensions:
    - Copilot for neovim: https://github.com/github/copilot.vim
    - Codeium for neovim: https://github.com/Exafunction/codeium.nvim
    - Copilot for PyCharm: https://plugins.jetbrains.com/plugin/17718-github-copilot
    - Codeium for PyCharm: https://plugins.jetbrains.com/plugin/20540-codeium-ai-autocomplete-and-chat-for-python-js-ts-java-go-

#2 Simple bug fixing (ex1_1.py)

```python
from rich import print

def factorial(n):
    if n == 1:
        return 1
    else:
        return n * factorial(n - 1)

test_values = [0, 1, 5, 10]
factorials = [factorial(n) for n in test_values]
print(factorials)
```

#3 Advanced bug fixing (ex1_2.py)

```python
import math

def compute_tf(document):
    tf_dict = {}
    doc_len = len(document)
    for word in document:
        if word not in tf_dict:
            tf_dict[word] = 1
        else:
            tf_dict[word] += 1

    for word in tf_dict:
        tf_dict[word] = tf_dict[word] / doc_len
    return tf_dict

def compute_idf(documents):
    N = len(documents)
    idf_dict = {}
    all_words = set(word for document in documents for word in document)

    for word in all_words:
        idf_dict[word] = 0
        for document in documents:
            if word in document and document[word] > 0:
                idf_dict[word] += 1

    for word in idf_dict:
        idf_dict[word] = math.log(float(idf_dict[word]) / N)
    return idf_dict

def compute_tfidf(tf, idf):
    tfidf = {}
    for word, val in tf.items():
        tfidf[word] = val * idf[word]
    return tfidf
```

#4 Parser for NER data (ex2.py)

```
sentences = [
    "Yesterday, John#Person went to [New York]#Location.",
    "Вчера, Джон#Person поехал в [Нью-Йорк]#Location.",
    "ትናንት, ጆን#Person ወደ [ኒው ዮርክ]#Location ሄደ.",
    "The [Eiffel Tower]#Location is located in Paris#Location.",
    "[Эйфелева башня]#Location находится в Париже#Location.",
    "[ኤፌል ጣብያ]#Location በፓሪስ#Location ውስጥ ተገኝታለች።",
    "Apple#Organization was founded by Steve#Person.",
    "Apple#Organization была основана Стивом#Person.",
    "አፕል#Organization በስቲቭ#Person ተመስርታለች።",
    "He uses Python#Language for [data analysis]#Activity.",
    "Он использует Python#Language для [анализа данных]#Activity.",
    "እርሱ ፒውተን#Language እንደ [ዳታ አነልይሲስ]#Activity ይጠቀማል።"
]
```

Thanks!