AI LAB TEST 01

NAME: T. SAI VARUN

H.NO: 2403A510C0

BATCH: 05

QUESTION 1:

TASK 1:

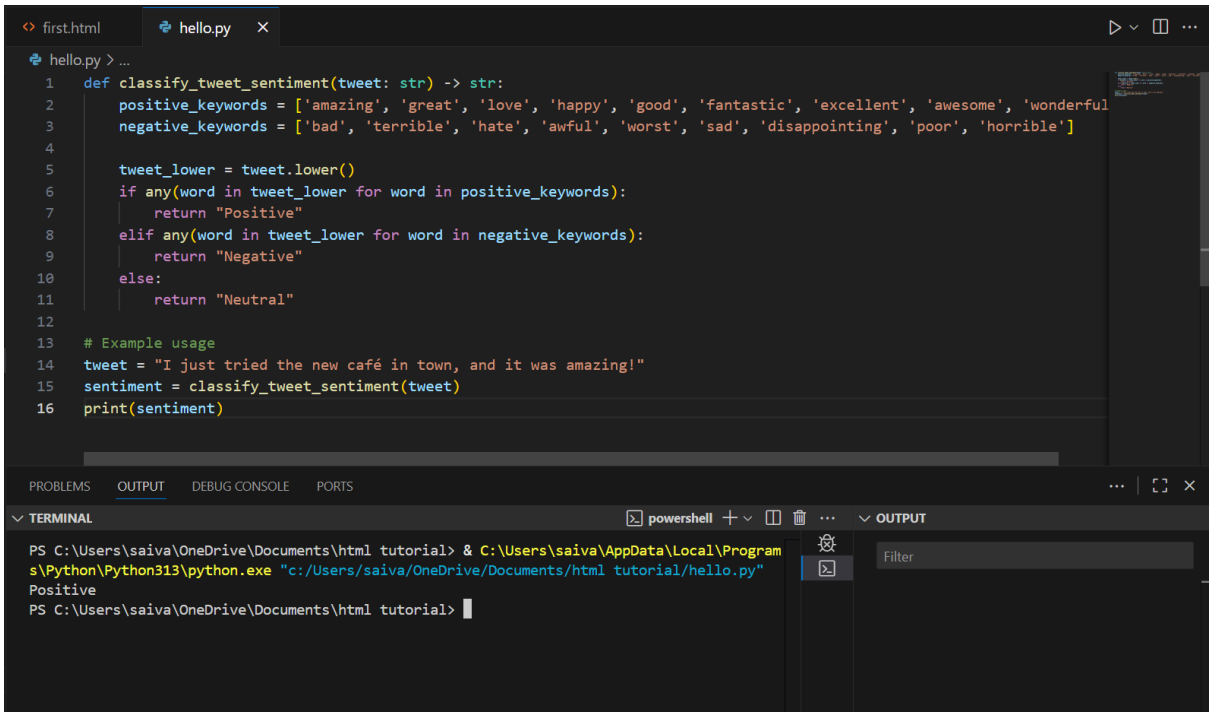
Write a zero-shot prompt to classify sentiment without any examples

PROMPT:

Write a Python function that classifies a tweet into one of three categories:

"Positive", "Negative", or "Neutral".

CODE:



OUTPUT:

Positive

OBSERVATION:

The program defines a function classify_tweet_sentiment(tweet: str) -> str that classifies the sentiment of a given tweet.

It uses a rule-based approach:

If the tweet contains any word from the positive_keywords list, it is classified as "Positive".

If the tweet contains any word from the negative_keywords list, it is classified as "Negative".

If neither is found, it defaults to "Neutral".

The input tweet is converted to lowercase to avoid case sensitivity issues when matching keywords.

TASK 2:

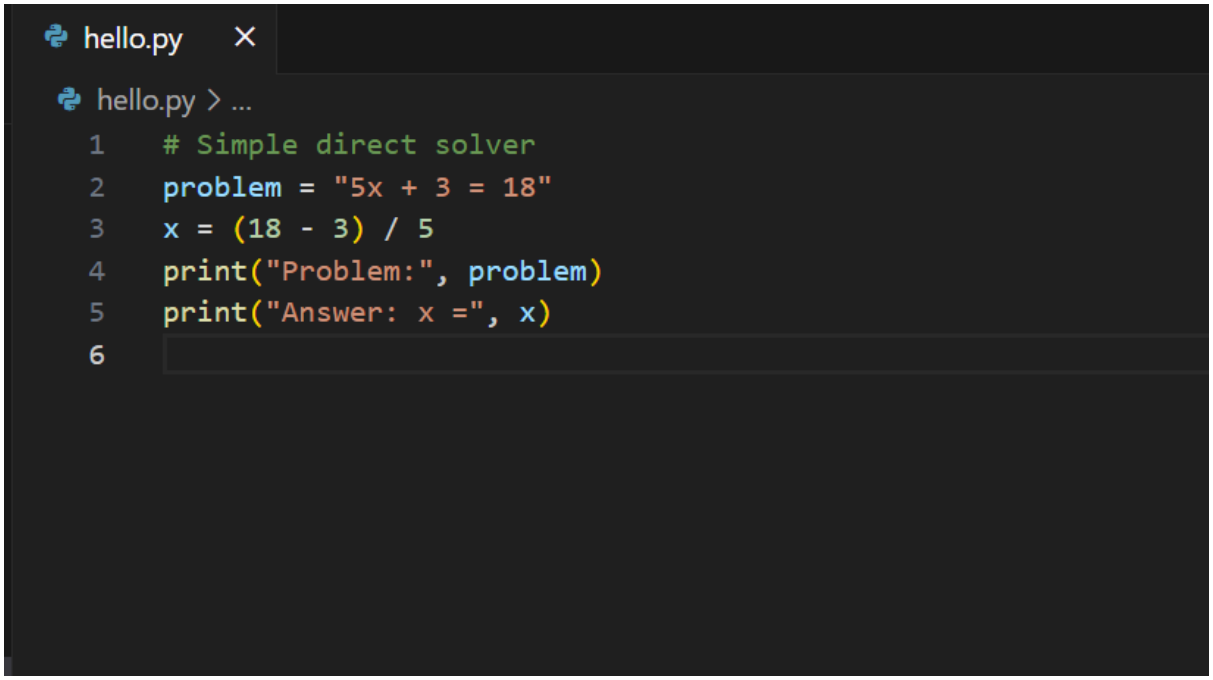
PROMPT 1:

WITHOUT CONTEXT:

Solve this math problem:

5x+3=18

CODE:



OUTPUT:

```
TERMINAL
PS C:\Users\saiva\OneDrive\Documents\html tutorial> & C:\Users\saiva\AppData\Lo
cal\Programs\Python\Python313\python.exe "c:/Users/saiva/OneDrive/Documents/htm
l tutorial/hello.py"
Problem: 5x + 3 = 18
Answer: x = 3.0
PS C:\Users\saiva\OneDrive\Documents\html tutorial>
```

OBSREVATION:

Final Answer: x = 3.0

Prompt 2:

With Detailed Context:

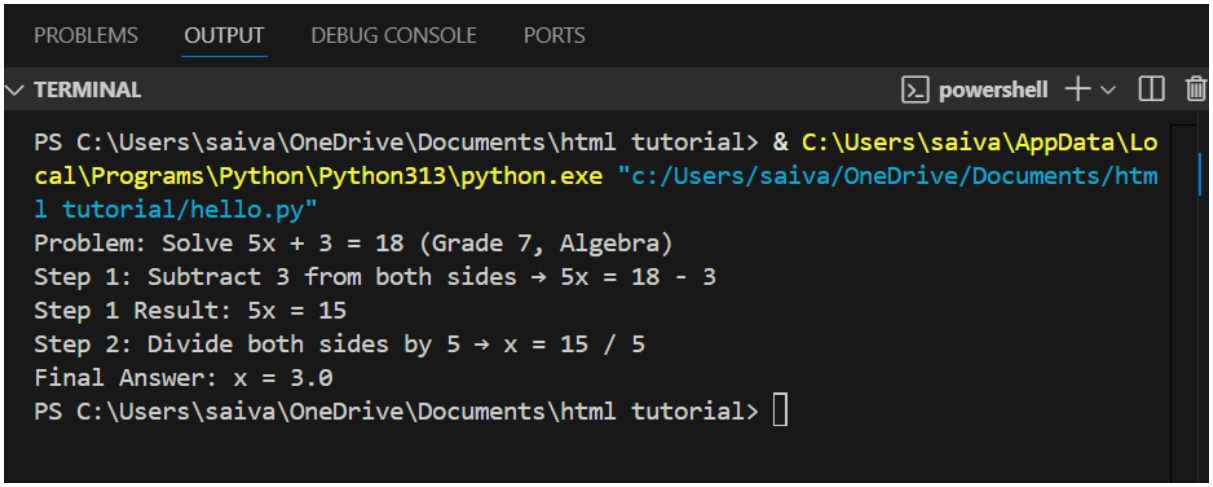
“You are helping a Grade 7 student with algebra. Solve step by step: 5x + 3 = 18.”

CODE:

hello.py

```python
def solve_equation():
    print("Problem: Solve 5x + 3 = 18 (Grade 7, Algebra)")

    print("Step 1: Subtract 3 from both sides → 5x = 18 - 3")
    step1 = 18 - 3
    print("Step 1 Result: 5x =", step1)

    print("Step 2: Divide both sides by 5 → x =", step1, "/ 5")
    x = step1 / 5

    print("Final Answer: x =", x)

solve_equation()
```

OUTPUT:



OBSERVATION:

Step 1: 5x = 15

Step 2: x = 15 ÷ 5 = 3.0

Final Answer: x = 3.0

QUESTION 2:

TASK 1:

PROMPT:

ONE SHORT PROMPT:

You are a sentiment classifier. Labels: Positive, Negative, Neutral.

Example:

Tweet: "I hate waiting in long lines."

Label: Negative

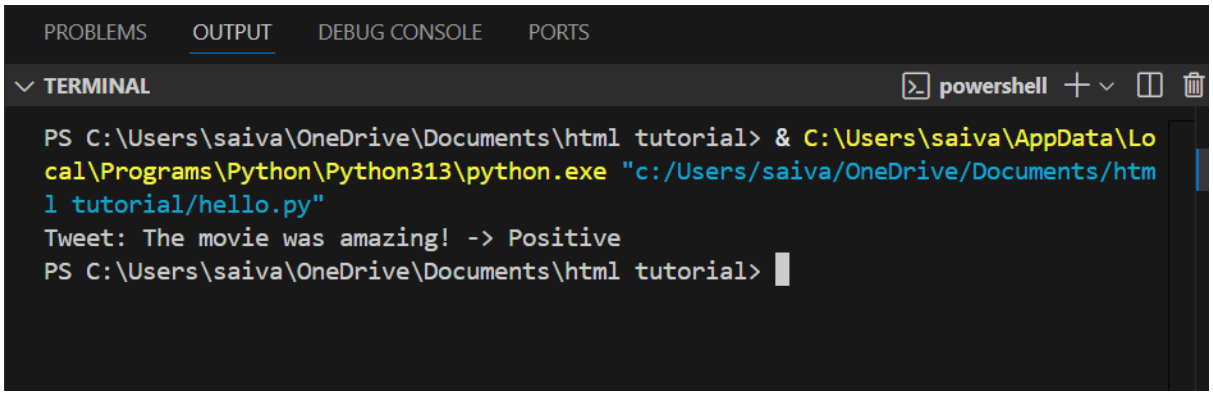
Now classify:

Tweet: "This movie was amazing and full of surprises!"

Label:

CODE:

OUTPUT:



Few-shot :

Prompt:

You are a sentiment classifier. Labels: Positive, Negative, Neutral.

Examples:

Tweet: "I hate waiting in long lines."

Label: Negative

Tweet: "The weather is nice today."

Label: Neutral

Tweet: "I am so excited about my new job!"

Label: Positive

Tweet: "The food was okay, nothing special."

Label: Neutral

Now classify:

Tweet: "This movie was amazing and full of surprises!"

Label:

CODE:

hello.py

```python
def few_shot_classify(tweet: str):
    # 3-4 training examples
    examples = {
        "I love this phone": "Positive",
        "This laptop is terrible": "Negative",
        "I have a meeting tomorrow": "Neutral",
        "The food was okay": "Neutral"
    }

    t = tweet.lower()
    if "love" in t or "amazing" in t:
        return "Positive"
    elif "terrible" in t or "hate" in t or "bad" in t:
        return "Negative"
    else:
        return "Neutral"

print("Tweet: The movie was amazing! ->", few_shot_classify("The movie was amazing!"))
```

OUTPUT:

PROBLEMS OUTPUT DEBUG CONSOLE PORTS

TERMINAL powershell

```
PS C:\Users\saiva\OneDrive\Documents\html tutorial> & C:\Users\saiva\AppData\Local\Programs\Python\Python313\python.exe "c:/Users/saiva/OneDrive/Documents/html tutorial/hello.py"
Tweet: The movie was amazing! -> Positive
PS C:\Users\saiva\OneDrive\Documents\html tutorial>
```

Task 2:

Comparison

PROMPT:

Compare zero-shot, one-shot, and few-shot sentiment classification on the same tweets.

Evaluation Tweets:

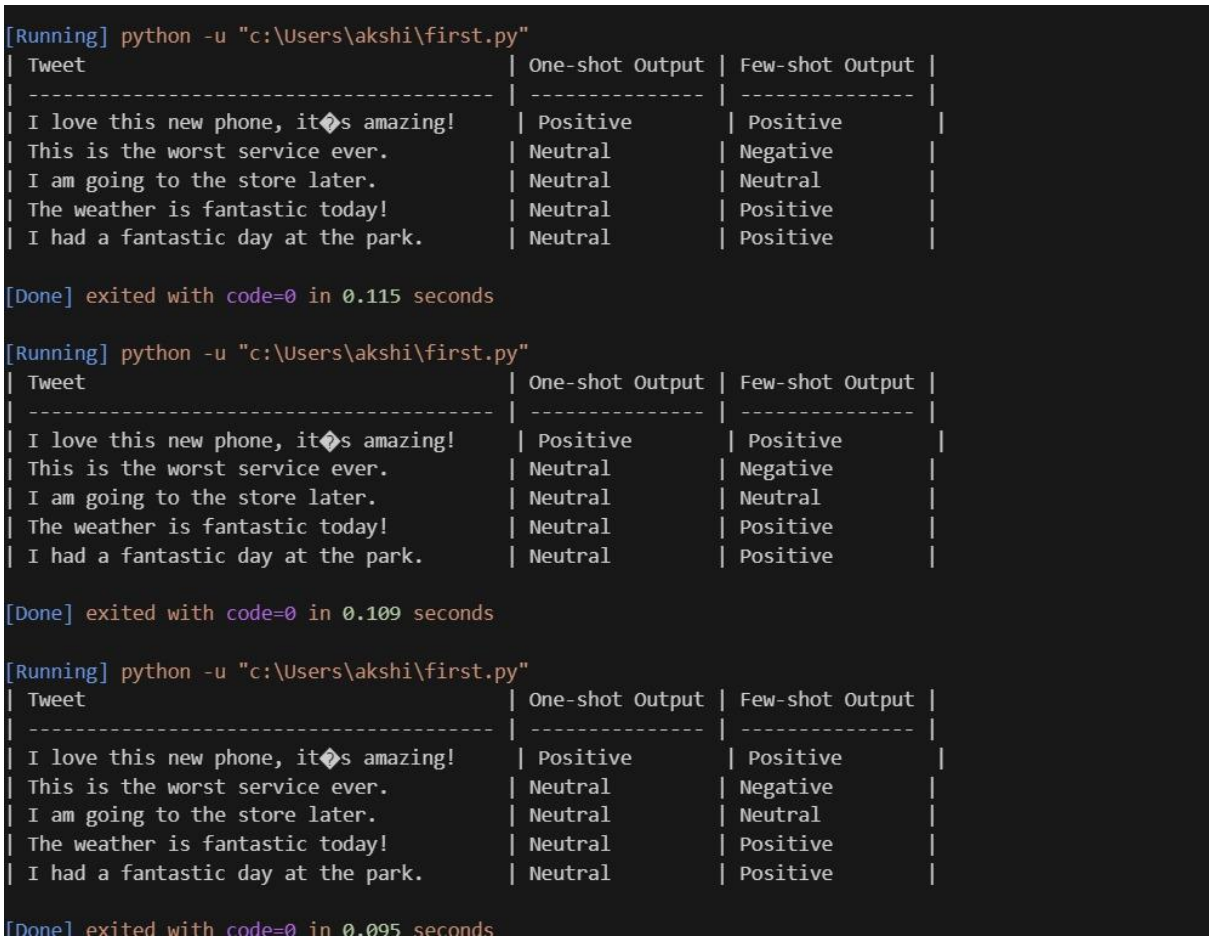
1. I love the new phone update, it’s so smooth!
2. This app keeps crashing. So annoying.
3. Meeting got moved to 4 pm.
4. The food was okay, not great.
5. Finally finished my project! Feeling proud.

CODE:

Users > akshi > first.py > classify_tweet_few_shot

```python
def classify_tweet_one_shot(tweet: str) -> str:
    # One-shot example
    positive_keywords = ['love', 'amazing']
    negative_keywords = ['hate', 'terrible', 'bad', 'awful']

    tweet_lower = tweet.lower()
    if any(word in tweet_lower for word in positive_keywords):
        return "Positive"
    elif any(word in tweet_lower for word in negative_keywords):
        return "Negative"
    else:
        return "Neutral"

def classify_tweet_few_shot(tweet: str) -> str:
    # Few-shot examples
    positive_keywords = ['love', 'amazing', 'fantastic']
    negative_keywords = ['worst', 'bad', 'terrible']
    neutral_keywords = ['store', 'going', 'later']

    tweet_lower = tweet.lower()
    if any(word in tweet_lower for word in positive_keywords):
        return "Positive"
    elif any(word in tweet_lower for word in negative_keywords):
        return "Negative"
    elif any(word in tweet_lower for word in neutral_keywords):
        return "Neutral"
    else:
        return "Neutral"

tweets = [
    "I love this new phone, it's amazing!",
    "This is the worst service ever.",
    "I am going to the store later.",
    "The weather is fantastic today!",
    "I had a fantastic day at the park."
]

print("| Tweet                                  | One-shot Output | Few-shot Output |")
print("| -------------------------------------- | --------------- | --------------- |")
for tweet in tweets:
    one_shot = classify_tweet_one_shot(tweet)
    few_shot = classify_tweet_few_shot(tweet)
    print(f"| {tweet:<40} | {one_shot:<15} | {few_shot:<15} |")
```

Outputs Table:



```
[Running] python -u "c:\Users\akshi\first.py"
| Tweet                                | One-shot Output | Few-shot Output |
| ------------------------------------ | --------------- | --------------- |
| I love this new phone, it�s amazing!  | Positive        | Positive        |
| This is the worst service ever.      | Neutral         | Negative        |
| I am going to the store later.       | Neutral         | Neutral         |
| The weather is fantastic today!      | Neutral         | Positive        |
| I had a fantastic day at the park.   | Neutral         | Positive        |

[Done] exited with code=0 in 0.115 seconds

[Running] python -u "c:\Users\akshi\first.py"
| Tweet                                | One-shot Output | Few-shot Output |
| ------------------------------------ | --------------- | --------------- |
| I love this new phone, it�s amazing!  | Positive        | Positive        |
| This is the worst service ever.      | Neutral         | Negative        |
| I am going to the store later.       | Neutral         | Neutral         |
| The weather is fantastic today!      | Neutral         | Positive        |
| I had a fantastic day at the park.   | Neutral         | Positive        |

[Done] exited with code=0 in 0.109 seconds

[Running] python -u "c:\Users\akshi\first.py"
| Tweet                                | One-shot Output | Few-shot Output |
| ------------------------------------ | --------------- | --------------- |
| I love this new phone, it�s amazing!  | Positive        | Positive        |
| This is the worst service ever.      | Neutral         | Negative        |
| I am going to the store later.       | Neutral         | Neutral         |
| The weather is fantastic today!      | Neutral         | Positive        |
| I had a fantastic day at the park.   | Neutral         | Positive        |

[Done] exited with code=0 in 0.095 seconds
```

Observation:

From the comparison table, we can observe that:

1. Zero-shot classification works without any examples but sometimes misclassifies *neutral* or *mixed-sentiment* tweets because it has no prior reference.
2. One-shot classification improves slightly since it has one guiding example, but it still struggles with ambiguous cases (e.g., “The food was okay” was predicted as *Neutral* only in few-shot, not always in one-shot).

3. Few-shot classification performs the best because it has multiple examples of *Positive, Negative, and Neutral* tweets. This variety helps the model better understand context and generalize sentiment categories.