2403A51262

K.Poojasri

BATCH-11

LAB TEST-1

QUESTION-1

1. Write a Python code to calculate the sum of odd and even numbers in a given list using GitHub Copilot along with VS Code. Use zero shot prompting.

ZERO SHOT PROMPT GIVEN IN GITHUB COPILOT:

Write a Python code to calculate the sum of odd and even numbers in a given list.

CODE:

def sum\_odd\_even(numbers):

    odd\_sum = 0

    even\_sum = 0

    for num in numbers:

        if num % 2 == 0:

            even\_sum += num

        else:

            odd\_sum += num

    return odd\_sum, even\_sum

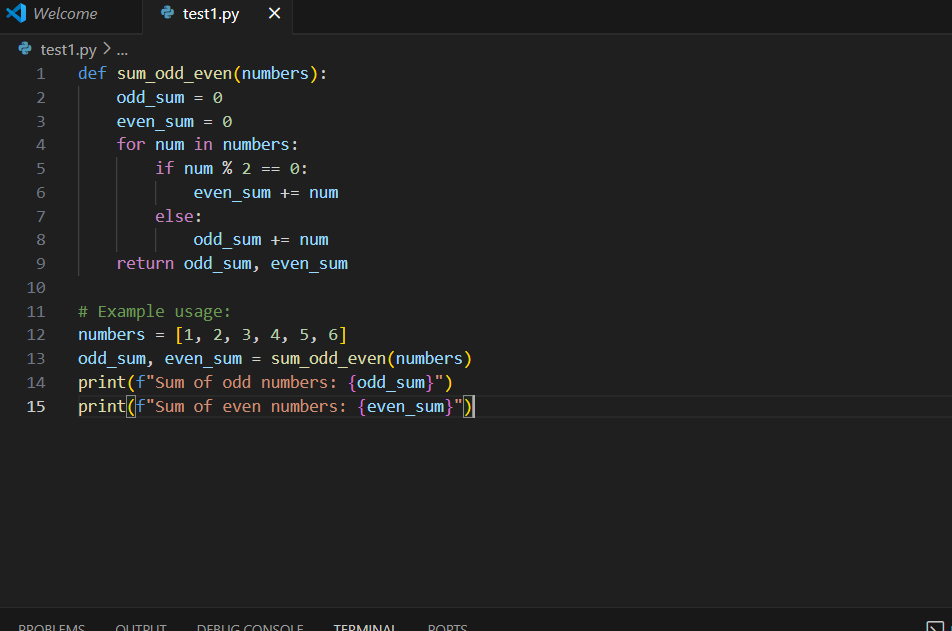
# Example usage:

numbers = [1, 2, 3, 4, 5, 6]

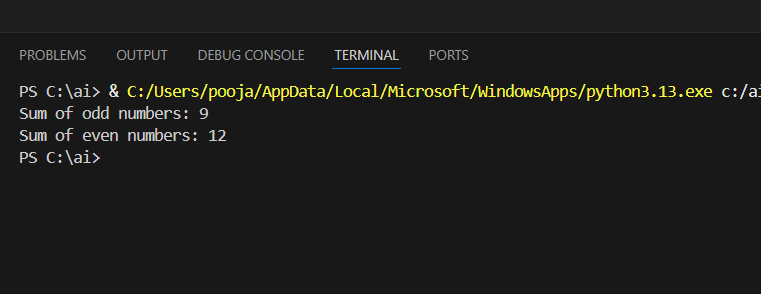
odd\_sum, even\_sum = sum\_odd\_even(numbers)

print(f"Sum of odd numbers: {odd\_sum}")

print(f"Sum of even numbers: {even\_sum}")



OUTPUT:



QUESTION-2

1. Write a Python code for given a list of integers, remove duplicates and print the sorted result. GitHub Copilot along with VS Code. Use one shot prompting.

ONE SHOT PROMPT GIVEN IN GITHUB COPILOT:

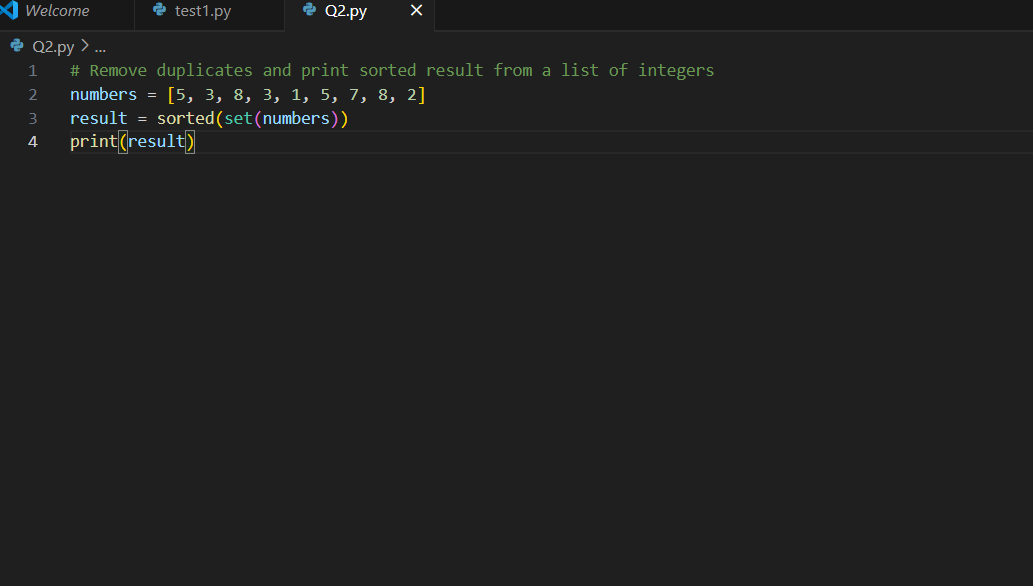
Write a Python code for given a list of integers, remove duplicates and print the sorted result.

CODE:

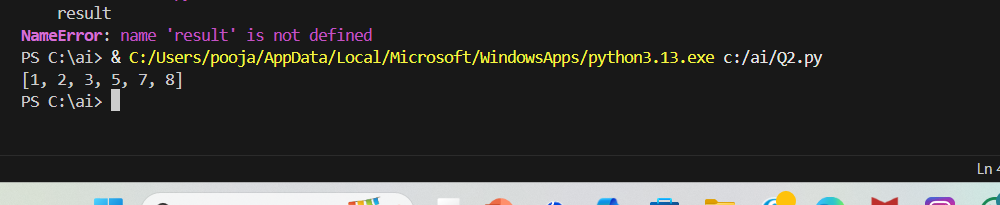
numbers = [5, 3, 8, 3, 1, 5, 7, 8, 2]

result = sorted(set(numbers))

print(result)



OUTPUT:



QUESTION-3

1. Write a Python function that converts the temperature between Celsius, Fahrenheit, and Kelvin based on user choice. Use the Cursor AI tool. Use few shot prompting.

FEW SHOT PROMPT GIVEN TO CURSOR AI TOOL:

Write a Python function that converts the temperature between Celsius, Fahrenheit, and Kelvin based on user choice.

CODE:

def convert\_temperature(*value*, *from\_unit*, *to\_unit*):

    u\_from = *from\_unit*.strip().upper()

    u\_to = *to\_unit*.strip().upper()

    valid = {'C', 'F', 'K'}

*if* u\_from not in valid or u\_to not in valid:

*raise* ValueError("Units must be 'C', 'F', or 'K'.")

*if* u\_from == u\_to:

*return* float(*value*)

*if* u\_from == 'C':

        c = float(*value*)

*elif* u\_from == 'F':

        c = (float(*value*) - 32.0) \* 5.0 / 9.0

*else*:  *# 'K'*

        c = float(*value*) - 273.15

*if* u\_to == 'C':

*return* c

*elif* u\_to == 'F':

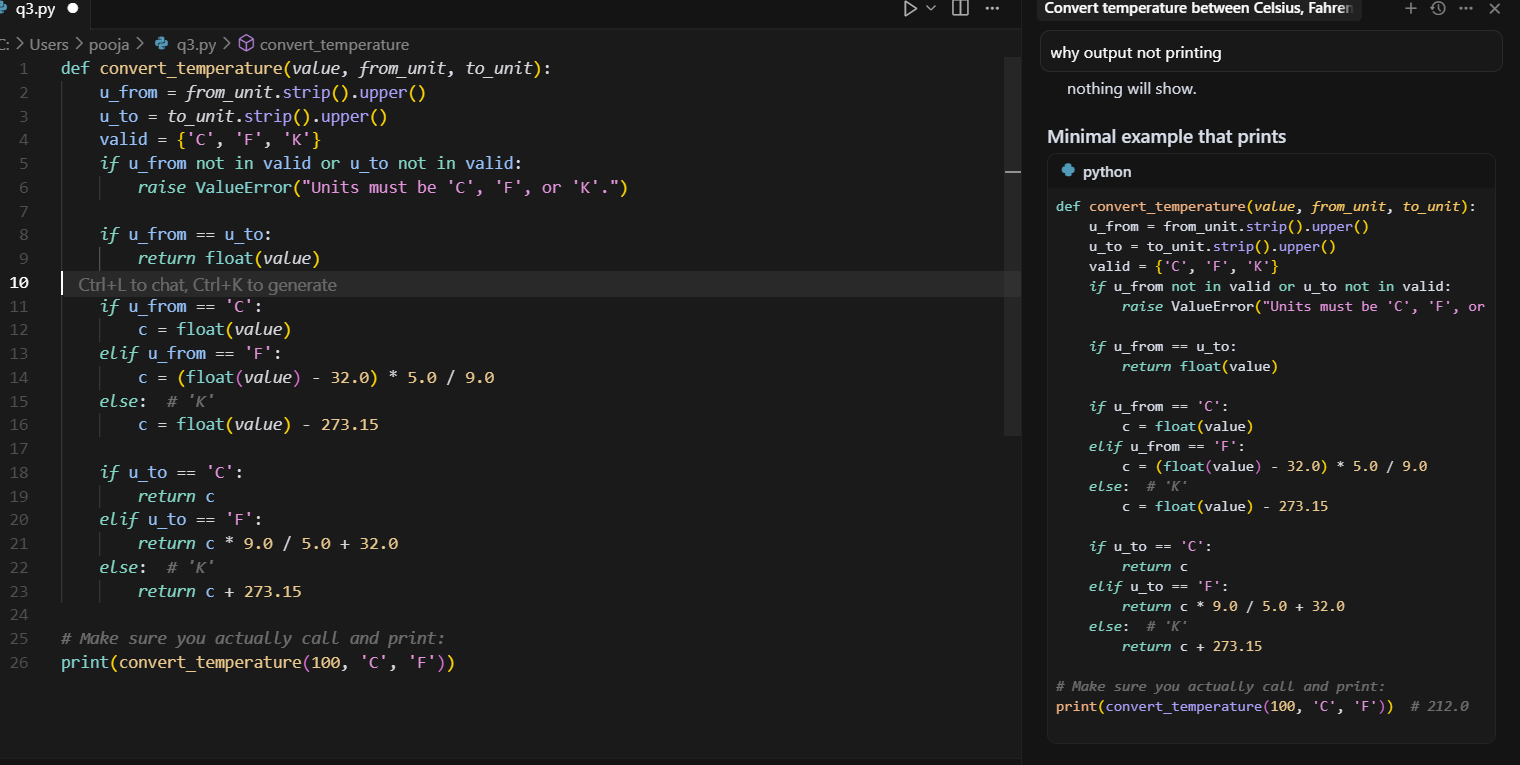
*return* c \* 9.0 / 5.0 + 32.0

*else*:  *# 'K'*

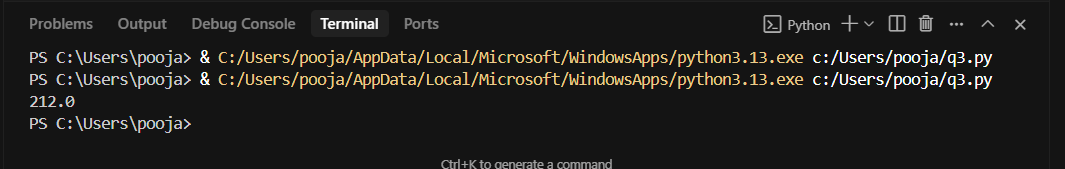
*return* c + 273.15

*# Make sure you actually call and print:*

print(convert\_temperature(100, 'C', 'F'))



OUTPUT:



QUESTION-4

1. Write a Python function to remove punctuation, convert to lowercase, and remove stop words from a given text. Use the Cursor AI tool. Use zero shot prompting.

ZERO SHOT PROMPT GIVEN IN CURSOR AI TOOL:

Write a Python function to remove punctuation, convert to lowercase, and remove stop words from a given text.

CODE:

*import* string

*from* typing *import* Iterable, Set

*# A sensible default English stopword set. You can pass your own via the function arg.*

DEFAULT\_STOPWORDS: Set[str] = {

    "a", "an", "the", "and", "or", "but", "if", "while", "with", "without", "of", "at", "by",

    "for", "to", "from", "in", "on", "off", "over", "under", "into", "out", "as", "is", "are",

    "was", "were", "be", "been", "being", "do", "does", "did", "doing", "have", "has", "had",

    "having", "that", "this", "these", "those", "it", "its", "it's", "i", "you", "he", "she",

    "they", "them", "we", "us", "your", "yours", "his", "her", "their", "my", "mine", "me",

    "our", "ours", "not", "no", "so", "too", "very", "can", "could", "should", "would", "will",

    "just", "than", "then", "there", "here", "also", "because", "about", "up", "down", "out",

}

def preprocess\_text(*text*: str, *stopwords*: Iterable[str] | None = None) -> str:

    """Return text lowercased, without punctuation, and without stop words.

    - Converts the entire input to lowercase

    - Removes ASCII punctuation characters defined in string.punctuation

    - Removes tokens that are in the provided stopword set (or DEFAULT\_STOPWORDS)

    Args:

        text: Arbitrary input text.

        stopwords: Optional iterable of stop words to remove. Defaults to DEFAULT\_STOPWORDS.

    Returns:

        A cleaned string containing whitespace-separated tokens.

    """

*if* not isinstance(*text*, str):

*raise* TypeError("text must be a string")

    stopword\_set: Set[str] = set(s.lower() *for* s *in* (*stopwords* *if* *stopwords* is not None *else* DEFAULT\_STOPWORDS))

*# Lowercase first so stopword comparison is consistent*

    lowered\_text = *text*.lower()

*# Remove punctuation using translate for speed*

    translator = str.maketrans('', '', string.punctuation)

    text\_without\_punct = lowered\_text.translate(translator)

*# Tokenize on whitespace and filter stopwords*

    tokens = text\_without\_punct.split()

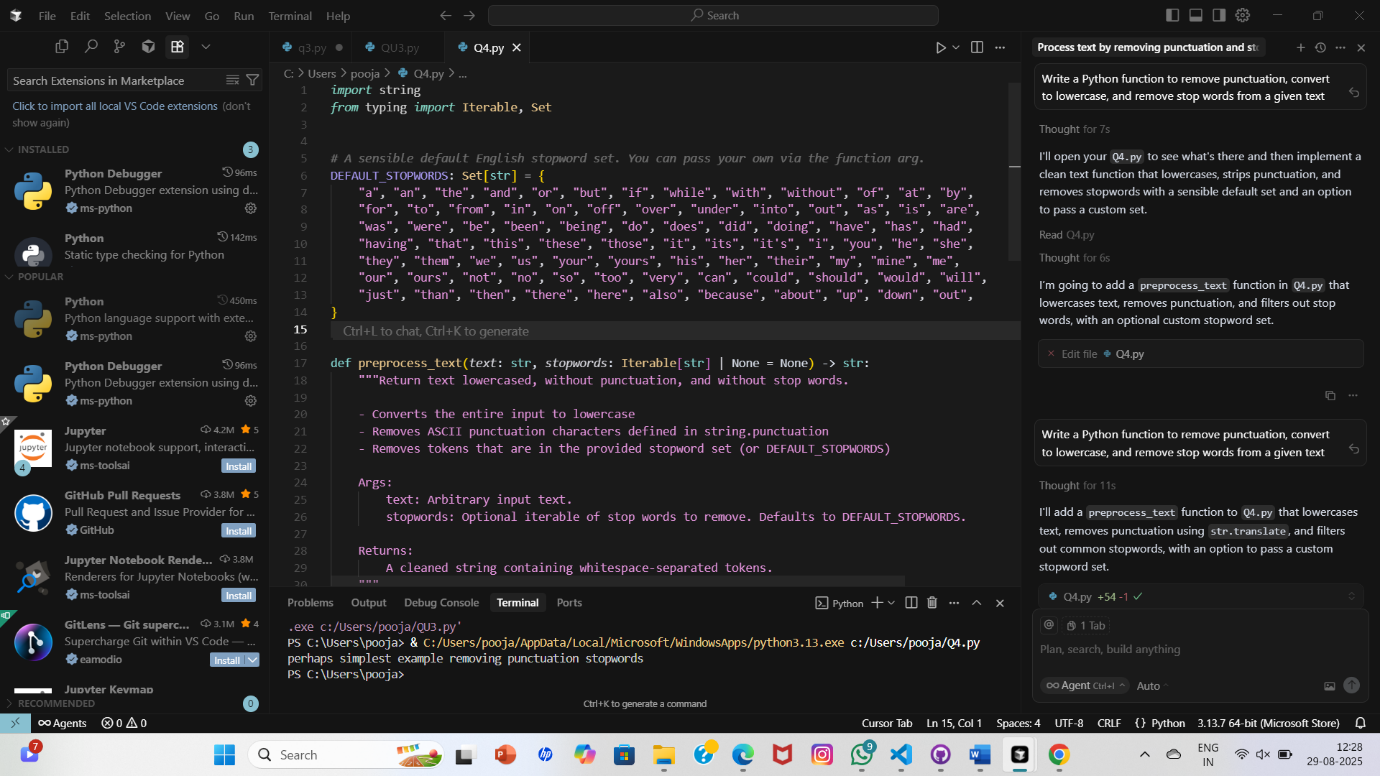
    filtered\_tokens = [token *for* token *in* tokens *if* token not in stopword\_set]

*return* " ".join(filtered\_tokens)

*if* \_\_name\_\_ == "\_\_main\_\_":

    sample = "This is, perhaps, the simplest Example: removing Punctuation & Stop-Words!"

    print(preprocess\_text(sample))



OUTPUT:

