```
# Step 1:
import pandas as pd

# Define the path to your CSV file
file_path = ‘PATH TO YOUR FILE.csv'

# Load the CSV file into a pandas DataFrame
df = pd.read_csv(file_path)

# Display the first few rows to inspect the contents
print(df.head())

#------------ The code
from docx import Document
import pandas as pd

# Load the CSV file and show a confirmation message
file_path = 'PATH TO YOUR FILE.csv'

# Load the CSV file into a pandas DataFrame
try:
    df = pd.read_csv(file_path)
    print("All is done great with the CSV file.")  #
Print success message
except Exception as e:
    print(f"Error loading CSV file: {e}")


# Step 4 (Revised): Function to save multiple sentences
to Word document with aligned glosses
def save_sentences_to_word(sentences_data):
    # Create a new Word document
    doc = Document()

    # Loop through each sentence data (English words,
glosses, translation)
    for sentence_data in sentences_data:
        english_words, glosses_list, translation =
sentence_data

        # Create a paragraph for the English words
        english_line = " ".join([word.ljust(15) for word
in english_words])
```

```
        doc.add_paragraph(english_line)  # Add English
words with fixed-width formatting

        # Create a paragraph for the glosses aligned
below the English words
        gloss_line = " ".join([gloss.ljust(15) for gloss
in glosses_list])
        doc.add_paragraph(gloss_line)  # Add glosses with
fixed-width formatting

        # Add the translation on a new line
        doc.add_paragraph(f'"{translation}"')

    # Save the document
    output_path = 'THE PATH WHERE YOUR OUTPUT IS SAVED/
sentences_glossing_output.docx'
    doc.save(output_path)
    print(f"Glossing saved to {output_path}")


# Step 4: Main logic for processing multiple sentences
def process_multiple_sentences_glossing(df):
    # Ask the user how many sentences they want to gloss
    num_sentences = int(input("How many sentences would
you like to gloss? "))

    # Validate the number of sentences
    if num_sentences < 1:
        print("Please enter at least one sentence.")
        return

    sentences_data = []  # To store data for all
sentences

    # Loop through each sentence
    for s in range(num_sentences):
        print(f"\nProcessing sentence {s + 1} of
{num_sentences}:")

        # Ask the user for the full sentence in Arabic
(space-separated words)
        arabic_sentence = input("Please enter the full
Arabic sentence: ")
```

```
        # Split the sentence into individual words
        arabic_words = arabic_sentence.split()

        # Initialize lists to store the English words,
glosses, and translations for each sentence
        english_words = []
        glosses_list = []
        translations_list = []

        # Loop to collect each Arabic word and retrieve
its gloss
        for arabic_word in arabic_words:
            # Search for the row where 'arabic_word'
matches the input
            result = df[df['arabic_word'] == arabic_word]

            # Check if the word is found
            if not result.empty:
                english_words.append(result.iloc[0]
['english_word'])
                glosses_list.append(result.iloc[0]
['glosses'])
                translations_list.append(result.iloc[0]
['translation'])
            else:
                print(f"Word '{arabic_word}' not found in
the database.")
                return  # Exit if any word is not found

        # Join translations for a full sentence
        full_translation = " ".join(translations_list)

        # Add the current sentence's data to the list
        sentences_data.append((english_words,
glosses_list, full_translation))

    # Print and save all sentences
    save_sentences_to_word(sentences_data)


# Run the multiple sentence glossing function
process_multiple_sentences_glossing(df)
```