**Name:** Noah Muncie

**Date Created:** 02/09/2025

**Program Description:**

The program is designed to scan email messages and determine the likelihood that the email is a spam message based off flagged keywords.

**Functions used in the Program (list in order as they are called):**

1. **Function Name:** main()

**Description:** This is the main function in the program which is responsible for calling the other functions and organizing the workflow.

**Parameters:** None

**Variables:**

1. emailText (string) – Holds the original email text as provided by the user
2. spamScore (integer) – The score value that determines the likelihood that the email was a spam message. The value is returned by calculate\_spam\_score().
3. foundWords (list) - The flagged words that were found in the provided message. The value is returned by calculate\_spam\_score().
4. likelihood (string) – A string value that displays the likelihood that the provided message was a spam message. The value is returned by evaluate\_spam\_score().

**Logical Steps:**

1. Prompt the user to enter an email message.
2. Pass the email text to calculateSpamScore() to determine its spam score and flagged words.
3. Pass the spam score to evaluateSpamScore() to determine the likelihood of spam.
4. Display the spam score, likelihood, and any flagged words to the user.

**Returns:** None

2. **Function Name:** calculate\_spam\_score()

**Description:** Scans the email message for spam-related keywords and assigns a spam score based on the number of occurrences.

**Parameters:**

1. emailText (string) – The email message to be analyzed.

**Variables:**

1. spamKeywords (list) – A predefined list of common spam-related words and phrases.
2. emailTextLower (string) – The email text converted to lowercase for case-insensitive comparison.
3. spamScore (integer) – A counter for the number of spam-related words found.
4. foundWords (list) – A list of flagged spam words found in the email.

**Logical Steps:**

1. Convert the email text to lowercase.
2. Iterate through spamKeywords and check if each word appears in the email text.
3. If a word is found, increase spamScore and add it to foundWords.
4. Return spamScore and foundWords.

**Returns:**

1. spamScore (integer) – The total spam score based on keyword occurrences.
2. foundWords (list) – The list of flagged spam words.

**3. Function Name:** evaluate\_spam\_score()

**Description:** Determines the likelihood of an email being spam based on the spam score.

**Parameters:**

1. score (integer) - The spam score calculated by calculate\_spam\_score().
2. spamWordsFound (list) - The spam words detected in the email.

**Variables:**

1. likelihood (string) – A string that categorizes the spam risk based on the spam score.

**Logical Steps:**

1. Compare the spam score to predefined thresholds:
   * 0 → "Not Spam"
   * 1-3 → "Unlikely Spam"
   * 4-7 → "Possibly Spam"
   * 8-12 → "Likely Spam"
   * 13+ → "Highly Likely Spam"
2. Return the appropriate likelihood string.

**Returns:**

1. likelihood (string) - The classification of the email based on the spam score.

**Link to repository:** https://github.com/NMHero1/COP2373