**Technical Design**

**Name:** Livia Augusto Razera

**Date Created:** June 8, 2025

**Program Description:**

This program prompts user to enter an email message, it then analyzes it for common span wors and phrases. It checks words from a list of 30 spam related words/phrases. For each match, the program increases the scam score by 1. Based on the score, it rates the likelihood that the message is spam, it also shows users what words/phrases triggered it.

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

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

**Description:** Handles user input, calls both other functions, and prints the final results.

**Parameters:** none

**Variables:**

Email\_message(string): the message the user entered.

Score(int): number of spam words found.

Matched\_words(list of strings): spam words/phrases found in the email.

Rating(string): likelihood of spam returned by rate\_spam.

**Logical Steps:**

1. Prompt user for email message.
2. Stores user input.
3. Call analyze\_email() to get spam score and matched words.
4. Call rate\_spam() to determine how likely the message is spam.
5. Print the spam score and rating.
6. Display which spam words were found (if any).

**Returns:** none

2. **Function Name:** analyze\_email

**Description:** scans and counts the email for spam words/phrases.

**Parameters:** text(string): email being scanned.

**Variables:**

spam\_score (int): counts the number of spam words found.

matched (list of strings): stores the spam keywords.

text (string): lowercase version of the input message for comparison.

**Logical Steps:**

1. Convert input text to lowercase for case-insensitive comparison.
2. Loop through each spam keyword in the list SPAM\_WORDS.
3. If the keyword is in the message, increase spam\_score and add the word to matched.

**Returns**: spam\_score (int), and matched.

**3. Function Name:** rate\_spam

**Description:** based on the spam score, returns a string describing the likelihood that the message is spam.

**Parameters:** score (int): the number of spam keywords found.

**Variables:** none

**Logical Steps:**

1. If score is 0, return "Not spam".
2. If score is 1–3, return "Unlikely spam".
3. If score is 4–6, return "Possible spam".
4. If score is 7–10, return "Likely spam".
5. If score is over 10, return "Very likely spam".

**Returns:** string such as "Not spam", "Likely spam", etc.

**Logical Steps:**

1. The program starts by calling the main() function.
2. Inside main, the user is prompted to enter a message.
3. analyze\_email(email\_message) is called to get the spam score and matched keywords.
4. rate\_spam(score) is called to convert the score into a spam likelihood description.
5. The program prints the spam score, rating, and matched keywords (if any).

**Link to your repository:** <https://github.com/liviaar/COP2373.git>

**Output Screenshot:**

**A computer screen with white text

AI-generated content may be incorrect.**