**Technical Design Document**

Name: Delisma Desir

Date Created: 02/9/25

**Program Description:**

This program scans an email message for common spam-related words and phrases. It calculates a "spam score" based on the number of detected keywords and categorizes the likelihood that the message is spam. The user inputs an email message, and the program returns the spam score, likelihood of spam, and the words/phrases found within the message.

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

**1. Function Name: scan\_for\_spam**

**Description:**  
This function scans the input email message for common spam keywords and phrases. It identifies all occurrences of the spam keywords and stores them in a list.

**Parameters:**

* message (string): The email message provided by the user to scan for spam keywords.

**Variables:**

* spam\_hits (list): A list that holds the keywords/phrases found in the message.
* keyword (string): A single keyword/phrase from the spam\_keywords list that is checked against the input message.

**Logical Steps:**

1. Loop through each keyword in the predefined list of spam keywords.
2. For each keyword, check if it appears in the message (case insensitive).
3. If a keyword is found, add it to the spam\_hits list.
4. Return the spam\_hits list containing all the detected spam keywords.

**Returns:**

* spam\_hits (list): A list of spam keywords/phrases found in the message.

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

**Description:**  
This function calculates the "spam score" by counting the number of detected spam keywords in the input message.

**Parameters:**

* spam\_hits (list): The list of spam keywords/phrases detected in the message.

**Variables:**

* spam\_score (int): The total count of spam keywords detected.

**Logical Steps:**

1. Count the number of elements in the spam\_hits list.
2. Return the length of the spam\_hits list as the spam score.

**Returns:**

* spam\_score (int): The number of spam keywords found in the message.

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

**Description:**  
This function assesses the likelihood that a message is spam based on the spam score.

**Parameters:**

* spam\_score (int): The spam score calculated from the previous function.

**Variables:**

* spam\_likelihood (string): The categorization of the message's spam likelihood.

**Logical Steps:**

1. Check the spam score to determine the likelihood of the message being spam.
2. If the score is 0, return "Likely not spam."
3. If the score is between 1 and 5, return "Low likelihood of being spam."
4. If the score is between 6 and 10, return "Moderate likelihood of being spam."
5. If the score is between 11 and 15, return "High likelihood of being spam."
6. If the score is 16 or more, return "Very high likelihood of being spam."

**Returns:**

* spam\_likelihood (string): A message indicating the likelihood that the email is spam.

**4. Function Name: check\_email\_for\_spam**

**Description:**  
This is the main function that ties everything together. It prompts the user to input an email message, scans the message for spam keywords, calculates the spam score, and displays the result.

**Parameters:**

* None (it collects input directly from the user).

**Variables:**

* message (string): The email message entered by the user.
* spam\_hits (list): List of detected spam keywords.
* spam\_score (int): Calculated spam score.
* spam\_likelihood (string): Categorized likelihood that the message is spam.

**Logical Steps:**

1. Prompt the user to input an email message.
2. Call the scan\_for\_spam function to detect spam keywords in the message.
3. Call the calculate\_spam\_score function to get the spam score.
4. Call the rate\_spam\_likelihood function to determine the likelihood of spam based on the score.
5. Print the spam score, likelihood, and the list of spam words/phrases found.

**Returns:**

* None (it prints the results directly to the console).

**Logical Steps (in order):**

1. **check\_email\_for\_spam** is called by the user to start the program.
2. **scan\_for\_spam** is called within check\_email\_for\_spam to identify spam keywords in the email message.
3. **calculate\_spam\_score** is called within check\_email\_for\_spam to compute the spam score based on the detected spam keywords.
4. **rate\_spam\_likelihood** is called within check\_email\_for\_spam to evaluate the likelihood of the message being spam based on the spam score.
5. The results are printed to the console, including the spam score, likelihood of spam, and the spam keywords found.

**Link to Repository:**

**https://github.com/DeeDesir/Programming-Exercise-2.git**