**Technical Design Document**

**Name:** Henry Huitema

**Date Created:** June 8th, 2025

**Program Description:**

This program searches through a user-supplied message and checks it against a list of 30 phrases commonly found in spam emails. It then returns spam score (number of phrases found), the specific phrases found, and an estimate of how likely the message is to be spam based on the spam score.

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

1. **Function Name:** countPhrases

**Description:** This function takes a message and combs through it for specific phrases. It then stores the phrases and their frequency in a dictionary as a key value pair and returns the dictionary.

**Parameters:**

message (string) – User supplied contents of an email.

**Variables:**

phraseList (list) – List of 30 phrases/words commonly found in spam emails.

messageLower (string) – message parameter converted to an all-lowercase string, done so to allow the check to be case-insensitive.

phrasesAndCounts (dictionary with string keys and integer values) – Dictionary structured as “specific phrase”:integer count of times phrase appears in message.

**Logical Steps:**

1. Initialize phraseList list with 30 different phrases commonly found in spam emails.
2. Converts message into an all-lowercase string using .lower() method, stores result as messageLower.
3. Initialize empty phrasesAndCounts dictionary.
4. Iterate over phraseList, checking each phrase against messageLower. If a phrase is in messageLower, store the phrase and number of times it occurs as a key value pair in phrasesAndCounts.
5. Return phrasesAndCounts.

**Returns:** phrasesAndCounts

2. **Function Name:** getScore

**Description:** This function takes the dictionary returned by countPhrases, reads the values, and returns the sum of them as a score.

**Parameters:** phrasesFound (dictionary) – Dictionary returned by countPhrases.

**Variables:** score (integer) – Accumulator used to store the sum of the values.

**Logical Steps:**

1. Initialize score at 0.
2. Iterate over the values of phrasesFound, adding each value to the score variable.
3. Return score.

**Returns:** Spam score integer.

3. **Function Name:** getPhrases

**Description:** This function takes the dictionary returned by countPhrases, reads the keys, and returns a list containing each key.

**Parameters:** phrasesFound (dictionary) – Dictionary returned by countPhrases.

**Variables:** None used aside from parameters.

**Logical Steps:**

1. Convert dict\_keys object returned by the phrasesFound.keys() method into a list.
2. Return the resulting list.

**Returns:** List containing each specific phrase found.

4. **Function Name:** rateMessage

**Description:** This function evaluates the likelihood of the message being spam based on a provided spam score.

**Parameters:** score (integer) – Spam score of the message output by the getScore method.

**Variables:** percentFloored (integer) – 0 or (score – 1) \* 20, whichever value is greater.

**Logical Steps:**

1. Multiply (score – 1) by 20.
2. If previous calculation evaluates as -20, set it to 0. If not, continue.
3. Return the result of previous two steps or 100, whichever is lower.

**Returns:** (score – 1) \* 20, clamped between 0 and 100. This is a percentage chance the message is spam according to the program.

5. **Function Name:** main

**Description:** The main function takes a user-supplied message as a string input, passes it through the countPhrases

**Parameters:** This function takes no parameters.

**Variables:** hits (dictionary) – Used to store the output of countPhrases.

**Logical Steps:**

1. Take a user input.
2. Call countPhrases, store result in the hits variable.
3. Call getScore on hits within a formatted string to display the raw spam score of the message.
4. Call getPhrases on hits within a formatted string to display the specific phrases the program found within the message.
5. Call getScore again, using it as a parameter in rateMessage, which is called within a formatted string to display the estimated likelihood of the message being spam.

**Returns:** This function does not return anything.

**Link to your repository:** https://github.com/HenryH-SCF/COP2373

**Output Screenshot: (make sure big enough so I can see)**

