**Technical Design Document**

**Name:** Matthew Pocrnic  
 **Date Created:** 6/8/2025

## **Program Description:**

This program creates a spam detection system for email messages. The user enters an email message, and the program looks it over for the presence of 30 common spam-related words and phrases. The program calculates a "spam score" based on how many times these words are found. It then classifies the likelihood that the message is spam and displays the result along with the detected spam words and their counts.

## 

## **Functions used in the Program:**

## **1. Function Name: calculate\_spam\_score**

**Description:** Scans the input message for instances of known spam words and phrases and calculates a corresponding spam score.

**Parameters:**

* message *(str)* – The email message to be analyzed
* spam\_words *(list of str)* – A list of known spam-related words/phrases

**Variables:**

* message\_lower *(str)* – Lowercase version of the input message
* score *(int)* – Total count of spam word occurrences in the message
* found *(dict)* – Dictionary mapping spam words/phrases to their number of occurrences in the message

**Logical Steps:**

1. Convert message to lowercase for case-insensitive matching
2. Initialize score and found
3. For each word in spam\_words: Count how many times the word appears in the message, if found, add to found dictionary and update score
4. Return the total score and the found dictionary

**Returns:** Returns a tuple containing:

* score *(int)* – The total spam score
* found *(dict)* – Dictionary of detected spam words and their counts

### **2. Function Name: classify\_spam**

**Description:** Classifies the message based on the spam score.

**Parameters:**

* score *(int)* – The total spam score from calculate\_spam\_score

**Variables:** None

**Logical Steps:**

1. If score is 0, return "Not Spam"
2. If score is between 1 and 2, return "Low likelihood of spam"
3. If score is between 3 and 5, return "Moderate likelihood of spam"
4. If score is greater than 5, return "High likelihood of spam"

**Returns:** Returns a string representing the likelihood of spam

### **3. Function Name: main**

**Description:** Manages the user interface, collects input, processes the message, and displays the results.

**Parameters:** None

**Variables:**

* message *(str)* – Stores user input for the email message
* score *(int)* – Stores the spam score returned from calculate\_spam\_score
* found *(dict)* – Stores the dictionary of detected spam words/phrases
* classification *(str)* – Stores the classification returned from classify\_spam

**Logical Steps:**

1. Display a welcome banner
2. Prompt the user to enter an email message
3. Call calculate\_spam\_score() to compute the spam score and detected spam words
4. Call classify\_spam() to classify the likelihood of spam
5. Display the spam score and spam likelihood
6. If any spam words were found, display them and their counts; otherwise, display that no spam indicators were found

**Returns:** None

## **Logical Steps (Program Flow):**

1. Start the program by calling main()
2. In main(), prompt user to enter an email message
3. Call calculate\_spam\_score() with the input message and the spam word list
4. Call classify\_spam() using the returned spam score
5. Display results to the user:  
   1. Spam score
   2. Spam likelihood
   3. Detected spam words and counts (if any)
6. End program

**Link to my COP2373 repository:** [**here**](https://github.com/mpocrnic/COP2373)

Screenshot of output from running code

