Technical Design Document

# Author

Joseph D Sullivan <[JSulli40@Student.SCF.edu](mailto:JSulli40@Student.SCF.edu)>

# Date

February 9, 2025

# Chapter

03

# Assignment

01

# Repository

<https://github.com/JosephDSullivan/COP2373/blob/main/src/chapter03/jsulli40_chapter03_assignment01.py>

# Program Description

This program is designed to detect spam in an email message text.

# Logic

1. Get email message from user.
2. Retrieve spam phrases.
3. Create output text to display results.
4. Retrieve spam counts for email message.
5. If there is at least one item in spam counts, then add that information to output text.
6. Calculate spam score from spam counts.
7. Calculate spam likelihood from spam score.
8. Add final score and likelihood to output text.
9. Display output text.

# Dependencies

typing

# Constant(s)

| Name | Type | Description |
| --- | --- | --- |
| SPAM\_MAX\_SCORE\_UNLIKELY | float | Maximum spam score allowed for category Unlikely. |
| SPAM\_MAX\_SCORE\_LOW | float | Maximum spam score allowed for category Low. |
| SPAM\_MAX\_SCORE\_MEDIUM | float | Maximum spam score allowed for category Medium. |
| SPAM\_MAX\_SCORE\_HIGH | float | Maximum spam score allowed for category High. |

# Variable(s)

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

# Class(es)

None

# Function(s)

## main

Entry function for when code is invoked directly.

### Logic

1. Get email message from user.
2. Retrieve spam phrases.
3. Create output text to display results.
4. Retrieve spam counts for email message.
5. If there is at least one item in spam counts, then add that information to output text.
6. Calculate spam score from spam counts.
7. Calculate spam likelihood from spam score.
8. Add final score and likelihood to output text.
9. Display output text.

### Parameter(s)

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

### Constant(s)

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

### Variable(s)

| Name | Type | Description |
| --- | --- | --- |
| email\_message | str | get\_email\_message() |
| spam\_phrases | List[str] | get\_spam\_phrases() |
| output | str |  |
| spam\_counts | List[Tuple[str, int]] | get\_spam\_counts() |
| spam\_phrase | str | spam\_counts[0] |
| count | int | spam\_counts[1] |
| spam\_score | float | get\_spam\_score() |
| spam\_likelihood\_text | str | get\_spam\_likelihood\_text() |

### Return(s)

|  | Type | Description |
| --- | --- | --- |
|  |  |  |

## get\_email\_message

Retrieves email message text.

### Logic

1. Create prompt message text.
2. Retrieve email message text from user and return it.

### Parameter(s)

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

### Constant(s)

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

### Variable(s)

| Name | Type | Description |
| --- | --- | --- |
| prompt | str |  |
| email\_message | str | input() |

### Return(s)

|  | Type | Description |
| --- | --- | --- |
|  | str | Email message text |

## get\_spam\_phrases

Retrieves words / phrases commonly found in spam email.

### Logic

1. Clean each phrase in phrases by explicitly converting into a string, trimming white space, and converting to lowercase.
2. Return cleaned phrases, sorted alphabetically.

### Parameter(s)

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

### Constant(s)

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

### Variable(s)

| Name | Type | Description |
| --- | --- | --- |
| phrases | List[str] | List of words and phrases commonly found in spam email. |

Note: List was compiled from multiple sources. Please see the following files for details:

* src/chapter03/source\_activecampaign.com.pdf
* src/chapter03/source\_lix-it.com.pdf
* src/chapter03/source\_zerobounce.net.pdf
* src/chapter03/spam\_phrases.xlsx

### Return(s)

|  | Type | Description |
| --- | --- | --- |
|  | List[str] | A sorted list of common spam phrases. |

## get\_spam\_counts

Calculates counts of each spam phrase found in email message.

### Logic

1. Initialize list containing found phrases and their count.
2. Convert email message to lowercase so search is case-insensitive.
3. Iterate through each spam phrase and count occurrences in email message.
4. If any spam phrase is found in email message, then add that phrase and the number of times that phrase was found in email message to spam phrases found list.
5. Return spam phrases found.

### Parameter(s)

| Name | Type | Description |
| --- | --- | --- |
| email\_message | str |  |
| spam\_phrases | List[str] | List of words and phrases commonly found in spam email. |

### Constant(s)

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

### Variable(s)

| Name | Type | Description |
| --- | --- | --- |
| spam\_phrases\_found | List[Tuple[str, int]] | List of tuples where each tuple contains a spam phrase and its count in the email message. |
| email\_message\_lower | str | Lower case of email message. |
| spam\_phrase | str | element of spam\_phrases |
| count | int |  |

### Return(s)

|  | Type | Description |
| --- | --- | --- |
|  | List[Tuple[str, int]] | A list of tuples where each tuple contains a spam phrase and its count in the email message. |

## get\_spam\_score

Calculates spam score given a list of spam phrase counts.

### Logic

1. Initialize spam score for the email message.
2. Iterate through spam counts and add each count to spam score.
3. Return spam score.

### Parameter(s)

| Name | Type | Description |
| --- | --- | --- |
| spam\_counts | List[Tuple[str, int]] | A list of tuples where each tuple contains a spam phrase and its count in the email message. |

### Constant(s)

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

### Variable(s)

| Name | Type | Description |
| --- | --- | --- |
| spam\_score | float | Spam score for the email message. |
| count | int |  |

### Return(s)

|  | Type | Description |
| --- | --- | --- |
|  | float | Total spam score. |

## get\_spam\_likelihood\_text

Calculates spam likelihood text based on spam score and categories created via module constants SPAM\_MAX\_SCORE\_\*.

### Logic

1. Verify spam score is numeric. If not, raise error.
2. Verify spam score is nonnegative. If not, raise error.
3. Spam score is valid. Find the category based on constants and return name.

### Parameter(s)

| Name | Type | Description |
| --- | --- | --- |
| spam\_score | float | Spam score to calculate likelihood text for. |

### Constant(s)

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

### Variable(s)

| Name | Type | Description |
| --- | --- | --- |
|  |  |  |

### Return(s)

|  | Type | Description |
| --- | --- | --- |
|  | str | Spam likelihood text. |