Republic of the Philippines
CAMARINES SUR POLYTECHNIC COLLEGES
Nabua, Camarines Sur
COLLEGE *of* COMPUTER STUDIES

Final Project and Final Exam

A requirement in the course

CSEC-311 Defensive Programming

(Camarines Sur Polytechnic Colleges)

Submitted by:

Horlador, Ken B.

Ruam, John Mark C.

Vargas, Regine B.

I. Project Description

The Python-based project that we will be modifying is the Python Spelling Checker & Corrector by Data Flair. Ensuring accurate written communication requires eliminating spelling errors. However, as humans, we naturally make mistakes. Recognizing this, Data Flair has developed this spell checker, which checks spelling and suggests corrections. Data Flair built the prototype of this spell checker using Python. They leveraged the Tkinter module to create a user-friendly interface for accepting user input and displaying output based on the typed spelling. The TextBlob module, on the other hand, handles the core functionality of spellc hecking.

This project builds upon the capabilities of the existing Python Spelling Checker & Corrector, presenting a modified version with enhanced functionalities and additional features. This advancement aims to provide individuals seeking to refine their written communication with a significantly more accurate and user-friendly experience. Our project aims to enhance the functionality of the existing Python-based spelling checker through the implementation of additional features. This includes the development of a more intuitive and user-friendly interface to improve user experience, adding a user input validation and error handling function. To ensure the best possible performance, we seek to increase the accuracy and efficiency of the spell-checking algorithms. We are optimistic that the improved version of the Python Spelling Checker & Corrector holds the potential to become a valuable tool for individuals dedicated to enhancing the accuracy of their written communication and furthering their communication skills.

II. What did you do? Original vs. Modified

The modified code improves upon the original in several ways:

1) Error Handling

The modified code implements comprehensive error handling. It checks for multiple scenarios like empty input, numerical input, multiple words, and special characters. This prevents crashes and provides user-friendly error messages.

2) **Explicit Return Types**

Functions are annotated with return types indicating the expected return values. This provides clarity on what each function should return or if they don't return anything (None).

3) **Variable Type Annotations**

Variables are explicitly annotated with expected types. This enhances code readability by specifying the expected data types of these variables, aiding understanding and maintenance.

4) **Readability**

The use of clear variable names, comments, and structured blocks of code improves readability. Each section is encapsulated in functions or descriptive comments, making it easier to follow the code's logic.

5) **Clarity and Maintainability**

The code is organized into functions, each handling a specific task. This modular approach makes it easier to maintain and modify the code in the future.

6) **Enhanced User Interface**

The modified code introduces clear labels, improved formatting, and a more user-friendly interface with distinct buttons for different actions. It provides better feedback to the user by updating the label with the corrected word or appropriate error messages.

7) **Clipboard Functionality**

The code adds a feature to copy the corrected word to the clipboard, enhancing the usability of the program.

8) **Consistent Styling**

The use of a consistent style convention for components (like colors, fonts, positions) improves the overall visual consistency of the application.

In summary, these comprehensive enhancements not only fortify the code against potential errors but also significantly improve its readability, maintainability, and usability. The modified version stands as a more robust and user-friendly solution for spell-checking, meeting higher standards of code quality and user experience.

III. Screenshots of the Original vs Modified.

OLD (Check Spelling Function)

```
#Function to check the spelling and show the corrected spelling
def checkSpelling():
    a = text.get() #Getting the word user entered
    b = TextBlob(a) #Getting the object for the word
    correctedText.set("The corrected word is: "+str(b.correct())) #Showing the corrected word
```

MODIFIED (Check Spelling Function)

```
# [BUTTON-Check]
# This function checks the word input and performs spelling correction
def checkSpelling() -> str | None:
    try:
        # [INPUT] Getting the word user entered
        input_text: str = text.get().strip()
        input_text_includes_character = (re.compile('[@_!#$%^&*()<>?/|}{~:]').search(input_text) is not None)
        input_text_is_numerical = input_text.isdigit() or isinstance(input_text, int) or isinstance(input_text, float)
        input_text_more_than_one_word = len(input_text.split(" ")) >= 2

        """ Implemented error handling to gracefully manage potential issues."""
        # [ERROR]
        # This error msg will be displayed if the user clicks the "Click Me" button without entering any text.
        if len(input_text) == 0:
            raise ValueError("Please enter a word before clicking.")

        # [ERROR]
        # Check if the user's input is numerical
        if input_text_is_numerical:
            raise ValueError("Invalid input. Please enter a word.")

        # [ERROR]
        # User inputs more than one word
        if input_text_more_than_one_word:
            raise Exception("This program only checks a single word")

        if input_text_includes_character:
            raise Exception("The word shouldn't include special characters")

        # Textblob
        blob_text: TextBlob = TextBlob(input_text)  # Getting the object for the word

        # Corrected word
        corrected_word: str = str(blob_text.correct())

        # [SUCCESS]
        # The user input and the suggested correct word is equal
        if input_text.lower() is corrected_word.lower():
            correctedText.set("The spelling is correct.")
            # [RETURN]
            return corrected_word

        # [SUCCESS]
        # The blob package successfully generated the correct word
        else:
            correctedText.set("The corrected word is: " + corrected_word)  # Showing the corrected word
            # [RETURN]
            return corrected_word

    except Exception as e:
        correctedText.set("Error: " + str(e))
```

OLD (Creating the Window)

```
#Creating the window
wn = Tk()
wn.title("DataFlair Spell Checker")
wn.geometry('500x250')
wn.config(bg='SlateGray1')

#Creating the variables to get the word and set the correct word
text=StringVar(wn)

correctedText =StringVar(wn)
```

MODIFIED (Creating the Window)

```
# Single Style Convention
component = {
    'primary-bg-color': 'SlateGray1',
    'secondary-bg-color': 'SlateGray4',
    'anchor': 'e',
    'header-font-family': 'Times',
    'header-font-weight': 'bold',
    'header-font-size': 20,
    'body-font-family': 'calibre',
    'body-font-size': 13,
    'body-font-weight': 'normal',
    'position-x': 20,
    'position-y': 10,
    'justify-position': LEFT
}

# Creating the window
tkinterWindow: Tk = Tk()
tkinterWindow.title("Modified Spell Checker (CS3A-G7)")
tkinterWindow.geometry('500x250')
tkinterWindow.config(bg=component['primary-bg-color'])

# Creating the variables to get the word and set the correct word
text: StringVar = StringVar(tkinterWindow)
correctedText: StringVar = StringVar(tkinterWindow)
```

OLD (Label)

```
#The main label
Label(wn, text='DataFlair Spell Checker',bg='SlateGray1',
  fg='gray30', font=('Times', 20,'bold')).place(x=100, y=10)

#Getting the input of word from the user
Label(wn, text='Please enter the word',bg='SlateGray1',font=('calibre',13,'normal'), anchor="e", justify=LEFT).place(x=20, y=70)

Entry(wn,textvariable=text, width=35,font=('calibre',13,'normal')).place(x=20,y=110)

#Label to show the correct word
opLabel = Label(wn, textvariable=correctedText, bg='SlateGray1',anchor="e",font=('calibre',13,'normal'), justify=LEFT).place(x=20, y=130)
```

MODIFIED (Label)

```
# [LABEL]
# Getting the input of word from the user
(Label(tkinterWindow,
       text='Please enter the word:',
       bg=component['primary-bg-color'],
       font=(component['body-font-family'], component['body-font-size'], component['body-font-weight']),
       anchor=component['anchor'],
       justify=component['justify-position'])
 .place(x=component['position-x'], y=component['position-y'] + 60))

Entry(tkinterWindow, textvariable=text, width=35, font=('calibre', 13, 'normal')).place(x=20, y=110)

# [LABEL]
# Label to show the correct word
(Label(tkinterWindow,
       textvariable=correctedText,
       bg=component['primary-bg-color'],
       anchor="e",
       font=(component['body-font-family'], component['body-font-size'], component['body-font-weight']),
       justify=component['justify-position'])
 .place(x=component['position-x'], y=component['position-y'] + 130))
```

OLD (Button)

```
#Button to do the spell check
Button(wn, text="Click Me", bg='SlateGray4',font=('calibre', 13),
    command=checkSpelling).place(x=230, y=190)
```

MODIFIED (Button)

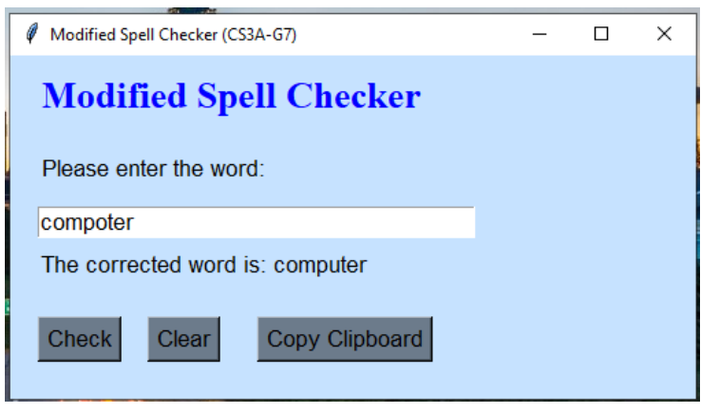
```
# [BUTTON]
# Button to do the spell check
(Button(tkinterWindow,
        text="Check",
        bg=component['secondary-bg-color'],
        font=(component['body-font-family'], component['body-font-size'], component['body-font-weight']),
        command=checkSpelling)
 .place(x=component['position-x'], y=component['position-y'] + 180))

# [BUTTON]
# Button to clear the input text
(Button(tkinterWindow,
        text="Clear",
        bg=component['secondary-bg-color'],
        font=(component['body-font-family'], component['body-font-size'], component['body-font-weight']),
        command=clearTextbox)
 .place(x=component['position-x'] + 80, y=component['position-y'] + 180))

# [BUTTON]
# Button to copy the text in the clipboard
(Button(tkinterWindow,
        text="Copy Clipboard",
        bg=component['secondary-bg-color'],
        font=('calibre', 13),
        command=clipboardCallback)
 .place(x=component['position-x'] + 160, y=component['position-y'] + 180))
```

IV. Results

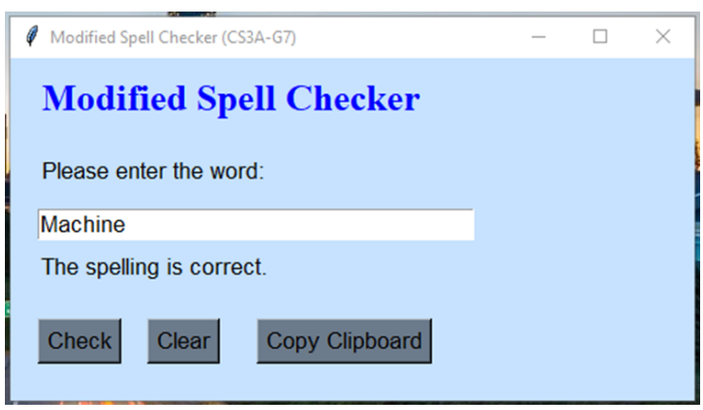
The enhanced usability of the Spelling Checker is evident. It incorporates error handling to adeptly manage potential issues and addresses words with special characters. The program now clearly defines how functions should behave and the types of variables it expects. It adheres to Python conventions, ensuring its clarity and consistency like eliminating squiggly lines. The design is now consistent with a single style. Two new features have been added: a clear and copy-to-clipboard function. Clear button, this function clears the input box and the label. Copy to Clipboard button, this function allows the user to copy corrected word directly to clipboard. The program also has helpful code documentation and comments for other programmers to understand the source code. As a result, this spelling checker and corrector has evolved into a user-friendly tool, clearly improved, and well-documented, ensuring accurate and efficient spell-checking capabilities.

The following are the additional features and function that we improved in the modified version:

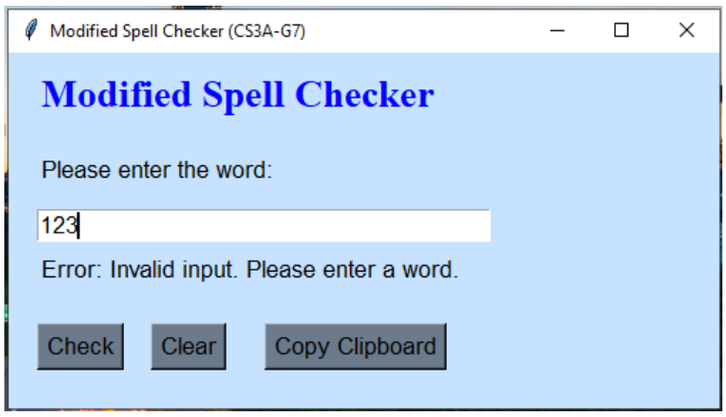
New GUI of the Modified Spell Checker



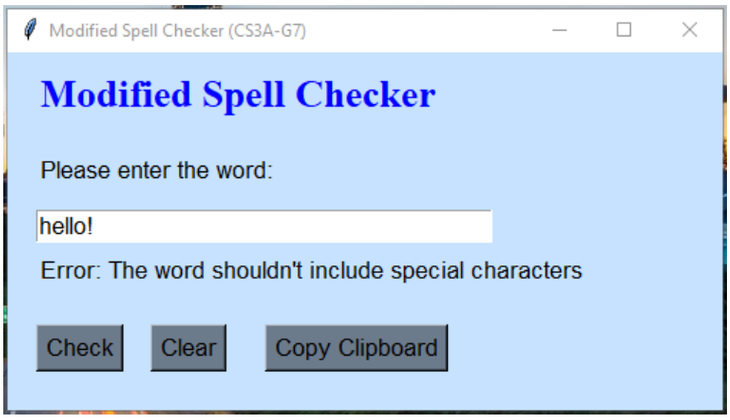
When the input word is already correct, the program will now inform the user that the spelling is correct.



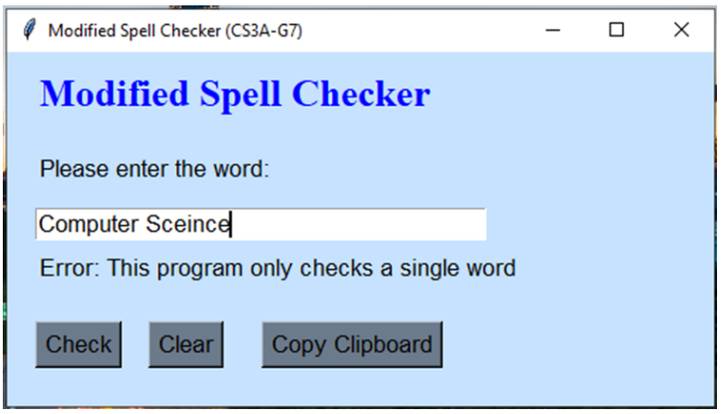
Error handling function where if the user input numerical, it will display the error message 'Invalid input. Please enter a word.'



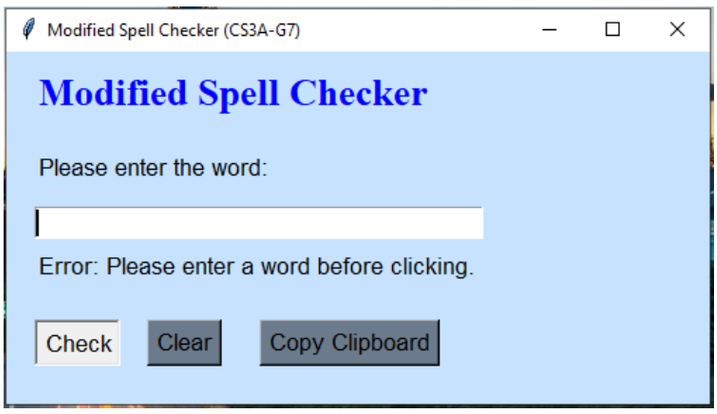
Error handling function where if the user enters a word with special characters, it will display the error message 'The word shouldn't include special characters.'



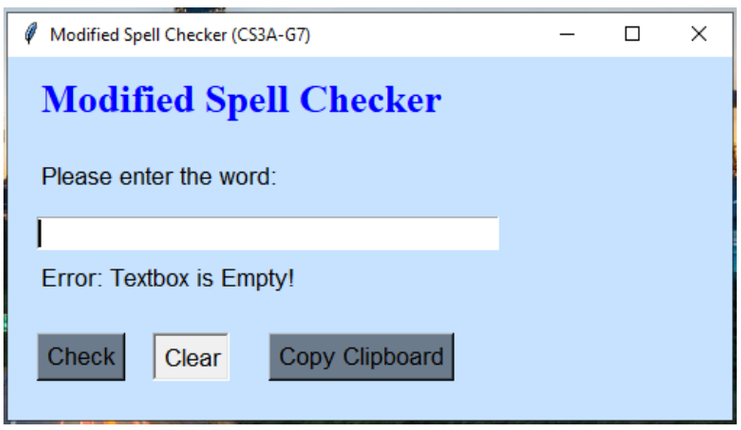
Error handling function where if the user enters two words, it will display the error message 'This program only checks a single word'.



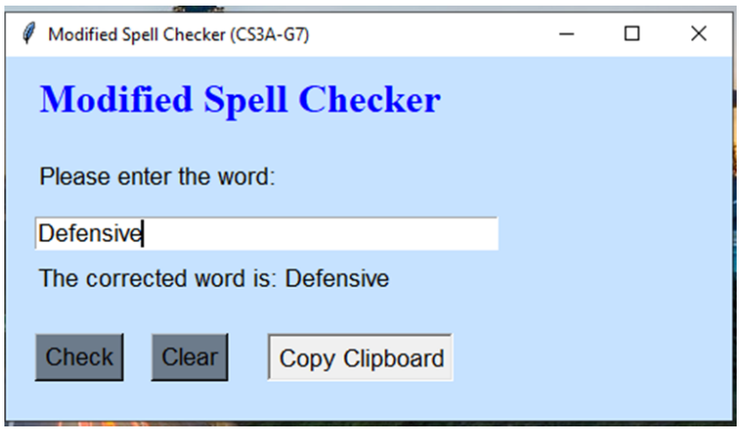
Error handling function where if the user tries to click check with entering a word, it will display the error message 'Please enter a word before clicking.'



Clear button is the added feature which the user can clear the entered word in the textbox. It has also an error handling where if the user tries to clear without input word, it will display the error message 'Textbox is empty!'.



Copy to clipboard button is also the added feature, the corrected word can now be copied directly to clipboard.



V. Reference

Python Spelling Checker & Corrector with Source Code. https://data-flair.training/blogs/python-spell-checker-correction-project/