202100377 – Bhavin Baldota
202100381 – Bramha Nimbalkar

Phase 2: Error Handling

Project Overview

Phase 2 of the System Programming Project focuses on the integration of robust error handling mechanisms within the Python-based tool using Tkinter. This phase ensures the efficient detection, identification, and handling of various errors that might occur during the lexical analysis of the provided C code.

Error Detection and Notification

1. **Header Errors Handling**: The system identifies and notifies users about incorrect or missing headers within the C code, ensuring compliance with the required header files.

```
# Header Errors Handling
if header_file not in header:
    messagebox.showerror("Syntax Error", "Invalid Header File")
    raise SyntaxError("Invalid Header File")
```

2. **Syntax Errors Notification**: It detects and notifies users of incorrect uses of **#define**, improper structures for **#include** statements, and missing semicolons, ensuring adherence to proper syntax.

```
# Syntax Errors Notification
if line.startswith('#define') and len(line.split()) != 3:
    messagebox.showerror("Syntax Error", "Invalid #define Usage")
    raise SyntaxError("Invalid #define Usage")

# Semicolon Missing Error Detection
if not line.endswith(';'):
    messagebox.showerror("Syntax Error", "Semicolon Missing")
    raise SyntaxError("Semicolon Missing")
```

3. **Function Type Errors Identification**: The system detects errors such as the absence of the **main()** function or incorrect usage of function types, providing relevant notifications to the user.

```
# Function Type Errors Identification
if ad_c_code[0][0] != 'int' or ad_c_code[0][1] != 'main()':
    messagebox.showerror("Syntax Error", "Invalid Function Type or main() Absent")
    raise SyntaxError("Invalid Function Type or main() Absent")
```

4. **Bracket Balance Errors**: It identifies unbalanced brackets (unequal counts of opening and closing brackets) and raises a notification for the user's attention.

```
# Bracket Balance Errors
if (content.count('(') != content.count(')')) or (content.count('{') !=
content.count('}')):
    messagebox.showerror("Syntax Error", "Unbalanced Brackets")
    raise SyntaxError("Unbalanced Brackets")
```

5. **Symbol Declaration Errors**: The system detects undeclared symbols or missing data types within the code, raising informative error notifications for rectification.

```
# Symbol Declaration Errors
if line[0] in types and line[1] not in symbol_table:
    messagebox.showerror("Syntax Error", "Undeclared Symbol")
    raise SyntaxError("Undeclared Symbol")
```

6. **Printf Statement Syntax Errors Identification**: It identifies incorrect syntax or missing data types within **printf** statements, providing detailed error notifications.

```
# Printf Statement Syntax Errors Identification
if opening_index == -1 or closing_index == -1:
    messagebox.showerror("Syntax Error", "Invalid Printf Statement Syntax")
    raise SyntaxError("Invalid Printf Statement Syntax")
```

Error Handling Procedure

Upon detection of errors, the system triggers the **messagebox.showerror** function, displaying informative error messages to alert users. Each identified error raises a **SyntaxError** exception, enabling precise identification and location of the error within the code for immediate rectification.

Significance and Conclusion

Phase 2's meticulous error handling mechanisms offer an essential layer of support to developers, ensuring that potential errors are promptly identified and detailed notifications provided for rectification. By detecting missing semicolons, the phase emphasizes strict adherence to proper syntax and structural correctness, further enhancing the quality and integrity of C programming code.