HOMEWORK ASSIGNMENT #2

Instructor: Mr. Usama Arshad
Subject code: CS 424

Due Date: March 11, 2024
No Late submission allowed **[Total 20 marks]**

PLEASE NOTE: THIS is an **INDIVIDUAL** assignment and **NOT** a group assignment.

Objectives:
This assignment aims to deepen your understanding of syntax analysis in compiler design. It focuses on constructing a parser that can recognize and validate the syntax of the MiniLang programming language, preparing you for more advanced stages of compiler construction.

Title: Design and Implement a Parser for "MiniLang"

Scenario:
Imagine a small, new programming language called "MiniLang." MiniLang is designed to be simple yet powerful enough to demonstrate key programming concepts. It supports basic arithmetic operations, variable assignments, if-else conditions, and print statements. Your task is to design and implement a parser for MiniLang using C++ or python.

Requirements:

1. **Language Specifications:**
   - Use the token types identified by your scanner for MiniLang, which include integer and boolean data types, arithmetic and logical operators, keywords, identifiers, literals, and comments.
   - Define the grammar for MiniLang, which should include rules for arithmetic expressions, variable assignments, conditional statements (if-else), and print statements.
2. **Parser Implementation:**
   - Parser Type: Decide between implementing a top-down parser (such as a recursive descent parser) or a bottom-up parser (like an LR parser). Consider the complexity of your language and the ease of implementation.
   - Grammar Rules: Implement the grammar rules of MiniLang. Each rule should correspond to a construct in MiniLang, such as expressions, statements, and programs.
   - Syntax Tree: Your parser should build an abstract syntax tree (AST) that represents the hierarchical structure of the source code. Each node in the tree should represent a language construct.

- Error Handling: Implement error handling mechanisms to report syntax errors, providing meaningful messages and possibly suggestions for corrections.

3. **Documentation:**
    - Document your design decisions, the structure of your parser, and how to run your program.
    - Include test cases that demonstrate your parser's capabilities, including edge cases.

Deliverables:

- Source code for your parser. (10)
- A report documenting your scanner's design, implementation details, and test cases. (Submit report of 2 pages only as hard copy in class.) (5)
- A set of MiniLang example programs and their corresponding output as screenshots. (3)
- (Send as a zip file on email – **usama.arshad@giki.edu.pk**)
- Upload on GitHub and share link in email. (2 marks)

Evaluation Criteria:

- Correctness and Completeness: Your parser should correctly implement the grammar of MiniLang and accurately build ASTs for valid input programs.
- Error Handling: Your parser should effectively detect and report syntax errors with clear and helpful messages.
- Code Quality: Your code should be well-organized, commented, and adhere to best practices in the chosen programming language.
- Quality and thoroughness of the documentation and test cases.

========================== *to err is human* ==============================