COMPILER CONSTRUCTION PARSER IMPLEMENTATION WORK-PLAN

P15/137032/2019 Hamud Abdulrahman Abeid

P15/1894/2020 Emmanuel Mbithi Muthiani

P15/1915/2020 Mugo Staicy Nelima Muthoni

Group 17

Methodology chosen.

Recursive Descent

After a long discussion we have concluded it to be easier to understand and implement than LL parsing. As beginners we believe this method of parsing will make it easier to follow and debug the parsing process. It also makes it easier to modify and maintain the parser as the grammar evolves.

Workplan

Research on recursive-descent predictive parsing (2 days)

- Understand the basics of top-down parsing and recursive-descent predictive parsing
- Research different algorithms and techniques for implementing a recursive-descent parser

(done by all group members)

Define parsing functions (3 days)

- Design and implement the parsing functions for each non-terminal in the grammar
- Test and refine the parsing functions to ensure correct behaviour

(done by all group members)

Implement the parsing algorithm (7 days)

- Choose the appropriate parsing algorithm based on your research
- Implement the algorithm and integrate it with the parsing functions
- Test and refine the parser

(done by Hamud Abdulrahman Abeid)

Error handling (1 day)

- Define error handling strategies for common parsing errors (e.g., syntax errors)

- Implement error handling mechanisms within the parser
- Test and refine the error handling functionality

(done by Mugo Staicy Nelima Muthoni)

Optimization (1 day)

- Identify potential performance bottlenecks in the parser
- Optimize the parser to improve its speed and efficiency
- Test and refine the optimized parser

(done by Emmanuel Mbithi Muthiani)