Compilers Project

Simple Programming Language using Lex and Yacc

Number of Students per Group

3-4 Students

Requirements

It is required to design and implement a programming language using the Lex and Yacc compiler generating package.

The Project comprises the following

1. Design a suitable programming language; you may use an existing one. The important constructs to be considered are:
   - Variables and Constants declaration.
   - Mathematical and logical expressions.
   - Assignment statement.
   - If-then-else statement, while loops, repeat-until loops, for loops, switch statement.
   - Block structure (nested scopes where variables may bedeclared at the beginning of blocks).
   - Functions
   - Enums
   - Exception handling
     - Just implementing the try catch finally logic. Exception classes and methods (printStackTrace, message, etc.) are optional.
   - Classes:
     - Constructors (Called using “new” keyword).
     - Data members.
     - Member functions.
     - Private and public access modifiers.
2. Design a suitable and extensible format for the symbol table.
3. Implement the lexical analyzer using Lex.
4. Design suitable action rules to produce the output quadruples and implement your parser using YACC.

5. Implement a proper syntax error handler.
6. Build a simple semantic analyzer to check for the following:
   - Variable declaration conflicts. i.e. multiple declaration of the same variable.
   - Improper usage of variables regarding their type.
   - Variables used before being initialized and unused variables.
   - The addition of type conversion quadruples to coupe with operators' semantic requirements, i.e. converting integer toreal, etc.
7. Implement a simple GUI.

Project Phases

Phase I: In this phase, it is required to do the following:

- Implement the lexical analyzer using Lex.
- Implement the parser using YACC.
- Deliver your Lex and YACC files.

Phase II: In this phase, it is required to modify your implementations to include the following:

- Design a suitable and extensible format for the symbol table.
- Design suitable action rules to produce the output quadruples.
- Implement a proper syntax error handler.
- Build a simple semantic analyzer.

Deliverables

1. Source code of your project.
2. A Document that contains the following:
   - Project Overview
   - Tools and Technologies used
   - A list of tokens and a description of each.
   - A list of the language production rules.
   - A list of the quadruples and a short description of each. e.g. :

Quadruple	Description
JMP L	Unconditional jump to Label L
NEG V1, V2	V2= -V1

Program evaluation

1. The program is to be fed by a source code file containing your language and produce the corresponding quadruples.
2. Display the syntax errors that exist in your program.
3. Display the semantic errors that exist in your program.
4. Display the symbol table.

Evaluation Criteria

1. The correctness of your quadruples.
2. The Syntax error handling.
3. The Semantic error handling.
4. Project understanding for the whole team.
5. The document.

Notes

1. Anything listed as optional will be considered a bonus.
2. Everything else mentioned is mandatory.
3. Any other semantic checks than the ones mentioned above will be considered a bonus.
4. Fancy GUIs will be considered a bonus, but GUI with just input file path and output file path is mandatory

Due Dates:

Phase I delivery: week 10 – 26 May 2020
Phase II delivery: week 14 – 23 June 2020