Compiler Project 1

Name: Chaozu Zhang ID: 11712021

Project target

For this project, there are two goals we have:

- Write regular expression using flex to match the input stream and generate tokens stream.
- Write token parsing production using bison to parse the token stream and generate the parse tree.

Project implementation

Data structure

I use a struct named ast(abstract syntax tree) in this project, contained in *ast.c*, there are five fields in this struct:

1. name: the name or type of the token.
2. value: the value of the token if there is .
3. lineno: line number of the token coming.
4. next_layer: the next child node it point to.
5. next_neighbor: the next node at the same level.

Main function in ast.c

1. new_node: to generate the new node whenever a legal token is recognised in flex, a new node will generate as the value of the token.
2. new_ast: to generate the new ast struct whenever a grammer is matched in bison.
3. parsetree: traverse the AST from the given node, print the parse tree.

Optional feature

1. HEX_INT/HEX_INT_WRONG recoginsed.
2. CHAR/CHAR_WRONG recognised.
3. Single-line comments, for example *// int a = 1;* will berecognised in flex.
4. Multiple-line comments, for example */* int a = 1;*/* will berecognised in flex.