Internal Document

Simple Parser

src/main.cpp

```
int main(int argc, char **argv) {
  // (...생략...)
  Lexer lexer(argv[1]);
  SymbolTable symbol_table;
  Parser parser(std::move(lexer), &symbol_table);
  // (...생략...)

  // 1. parse하여 parse_tree 생성
  Tree *parse_tree = parser.parse();

  // 2. ID, CONST, OP 개수 세기 및 정의되지 않은 식별자 사용 여부 확인
  parse_tree->analyze();

  // 3. 모든 Statement 평가
  parse_tree->evaluate();

  // 4. symbol_table에 담긴 변수들의 값 출력
  symbol_table.print_result();

  // (...생략...)
}
```

include/lexer.h

Terminal

```
enum Terminal {
  CONST,          // any decimal numbers
  IDENT,          // any names conforming to C identifier rule
  ASSIGNMENT_OP,  // :=
  SEMI_COLON,     // ;
  ADD_OP,         // + | -
  MULT_OP,        // * | /
  LEFT_PAREN,     // (
  RIGHT_PAREN,    // )
  END_OF_FILE,    // EOF
  UNKNOWN,        // unknown token
};
```

Lexer

```
// 어휘 분석을 해주는 class
class Lexer {
  ifstream file;

  char next_char;

  Terminal next_token;
  string token_string;

  public:
  Lexer(const char* filename);
  bool is_open();               // 파일 열기에 성공했는지

  void lexical();               // 어휘 분석
  Terminal get_next_token();    // 어휘 분석 후 인식 된 token를 반환
  string get_token_string();    // 어휘 분석된 string 반환
};
```

include/symbol_table.h

Container

```
// 변수의 값을 표현하는 class. Unknown과 정수형 표현 가능
class Container {
  bool is_unknown;
  int value;

  public:
  Container() : is_unknown(true), value(0) {}                // Unknwon 값을 가진 값 생성
  Container(int value) : is_unknown(false), value(value) {} // value 값을 가진 값 생성

  void add(Container container);
  void sub(Container container);
  void mult(Container container);
  void div(Container container);
  void print();                    // 값 출력
};
```

SymbolTable

```
// 변수들의 값을 담는 class
class SymbolTable {
  map<string, Container> table;

  public:
  bool is_exist(string ident);                          // 해당 이름을 가진 변수가 있는지 확인
  void add_ident(string ident);                         // 새로운 변수 추가
  Container get_value(string ident);                    // 해당 변수의 값 반환
  void set_value(string ident, Container container);    // 해당 변수의 값 변경
  void print_result();                                  // 모든 변수의 값 출력
};
```

include/parser.h

Parser

```
// lexer로부터 token들을 얻어와 parse_tree를 생성하는 class
class Parser {
  Lexer lexer;
  SymbolTable *symbol_table;

  // 각 non-terminal들을 파싱한 후 tree를 생성하여 반환
  Program *program();
  Statements *statements();
  Statement *statement();
  Expression *expression();
  TermTail *term_tail();
  Term *term();
  FactorTail *factor_tail();
  Factor *factor();

  public:
  Parser(Lexer lexer, SymbolTable *symbol_table);
  Tree *parse();  // 전체 코드 parsing
};
```