Tutorial 5: MiniC Symbol Table and Semantic Checking

MiniC Symbol Table is presented in `SymbolTable.h`. It involves symbol table for function declarations and for variable declarations respectively. class `VerifyAndBuildSymbols` is inherited from `ASTVisitor`. It will visit each node in the same way as ast printer in A3. In assignment 4, mainly fill in `VerifyAndBuildSymbols.cpp` along with `Minic.g4` if needed. Free to change any files.

Agenda

- Semantic Errors Intro
- Implementation Tips

Semantic Errors Intro

E.g. Function has return type "bool", but the returned expression has type "int"! (13:8)

```
-- Var and Func Declaration error
  Definition of function "name()" with different return type!
  Definition of function "name()" with different number of parameters!
  Definition of function "name()" with different parameter type at position x!
  Redefinition of function "name()"!
  Redefinition of variable/parameter "name" in the same scope!
-- Func Declaration and return type mismatch
  The function "name()" need to return a value at its end!
  Function has void return type, but the return statement has a returned expression!
  Function has non-void return type, but the return statement has no returned expression!
  Function has return type "y" but the returned expression has type "x"!
-- Break stmt
  Break statement must appear inside a for statement!
-- Expr error
  Conditional expression in if statement has non-bool type!
  Conditional expression in for statement has non-bool type!
  Negate "-" opcode must have int operand!
  Not "!" opcode must have bool operand!
  "&&"/"||" opcode must have bool operand!
  "=="/"!=" opcode must have same primitive type operand!
  ">" opcode must have int type operand!
  "<" opcode must have int type operand!
  ">=" opcode must have int type operand!
  "<=" opcode must have int type operand!
  Variable name is not declared before use!
  Array index expressions must have int operand!
```

```
  Indexing an non-array variable!
  Variable and the assignment expression do not have the same type!
  Integer literal must be inside the range of int!
  "+" opcode must have int type operand!
  "-" opcode must have int type operand!
  "*" opcode must have int type operand!
  "/" opcode must have int type operand!
-- Function Call error
  Function name() is not declared before use!
  Function name() is declared with x parameters but called with y arguments!
  Function name() does not match the type of the call argument at position x!
```

Implementation Tips

- You could build symbol table along with checking errors at the same time.
- Minic language says "The operators of level 3 do not associate, so a==b==c is illegal." It will bot be tested in following assignments. But you're free to check the error. The sample solution will show the answer in comments.
- You don't need to care about the correction of line number and error position number. Just make sure you print it.
- **Do not think about too strange or too extreme case!** The tests will not really be very extreme or closely examine edge cases to allow for some personal interpretation and differences in the compiler design. E.g.

  ```
  int Hello(int a, int b, int c);
  bool Hello(bool a, int b, bool c, int d){ return 1; }
  ```

  Under general case, the errors are placed in different positions. Multiple errors should be reported in the order of visiting AST node.