Compiler Project 2

11611006 邓韵杰

In this project, we should check whether input program has semantic error. I follow some rules and assumptions in `semantic.txt` to detect semantic error. Also, I add some extra rules to detect more error and remove some assumptions to make my compiler more flexible.

1. Required Rules(main Idea)

Declaration:

For each declared variable/function/struct, I add it to a variable/function/struct stable. When program use an ID(`ID`), compiler should ensure the ID is exist.

Usage:

For a declared variable, compiler should be able to check the expression(`Exp`) type to check whether assignment or operation to a variable is meaningful.

For a declared function, compiler should check whether its return type matches its declared type. Additionally, the function call matches its parameters and arguments.

2. Optional Rules

Same name variable defines in different scope is valid. Compiler maintain different variable table to separate the different scope. When a variable is declared or defined in current scope, compiler will only check current scope variable table to know whether it is existed. But for the usage of a variable, we should check current scope variable table and its father scope variable table.

For the equivalence between two struct, compiler will check whether the order and the type of each members is matched.

3. Bonus Rules

I add some bonus rules and provide test case and output.

Support more function:

- `break` can only be used in `while` and `for` loop
  - self_test_01.spl
- variable declaration in `for` loop
  - self_test_01.spl
- field names in struct are not necessary to be unique(relax assumption 6)
  - self_test_04.spl

Support more error check:

- type of condition(should be `int`) expression check in `if` and `while` (revoke assumption 3)
  - self_test_02.spl
- assignment/operation check between different type variable(revoke assumption 1, 2 and 4)
  - self_test_03.spl
- undefined struct type will be detected
  - self_test_05.spl