Compiler Project 2

Name: Chaozu Zhang ID: 11712021

Project target

For this project, the main goal is to do semantic analysis based on the previous project.

Project implementation

Data structure

- For the syntax tree:

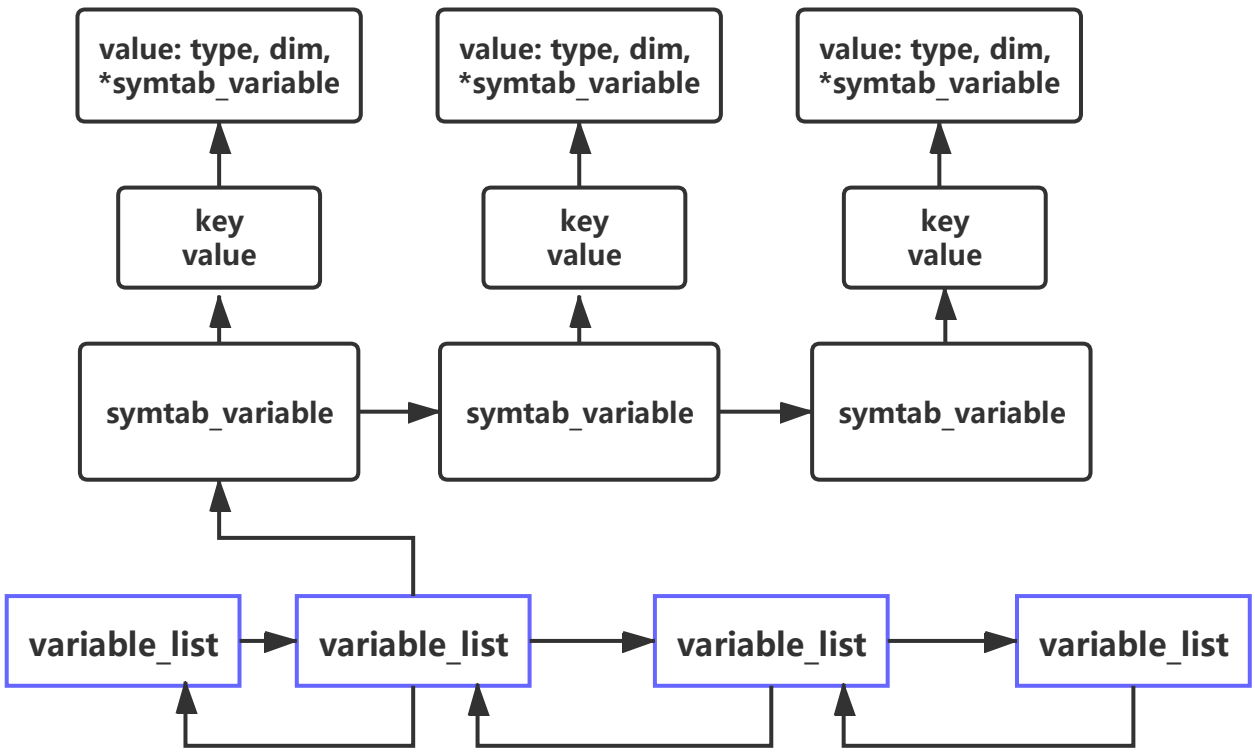
I use a struct named ast(abstract syntax tree) in this project, contained in *ast.c*, there are five fields in this struct:

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.

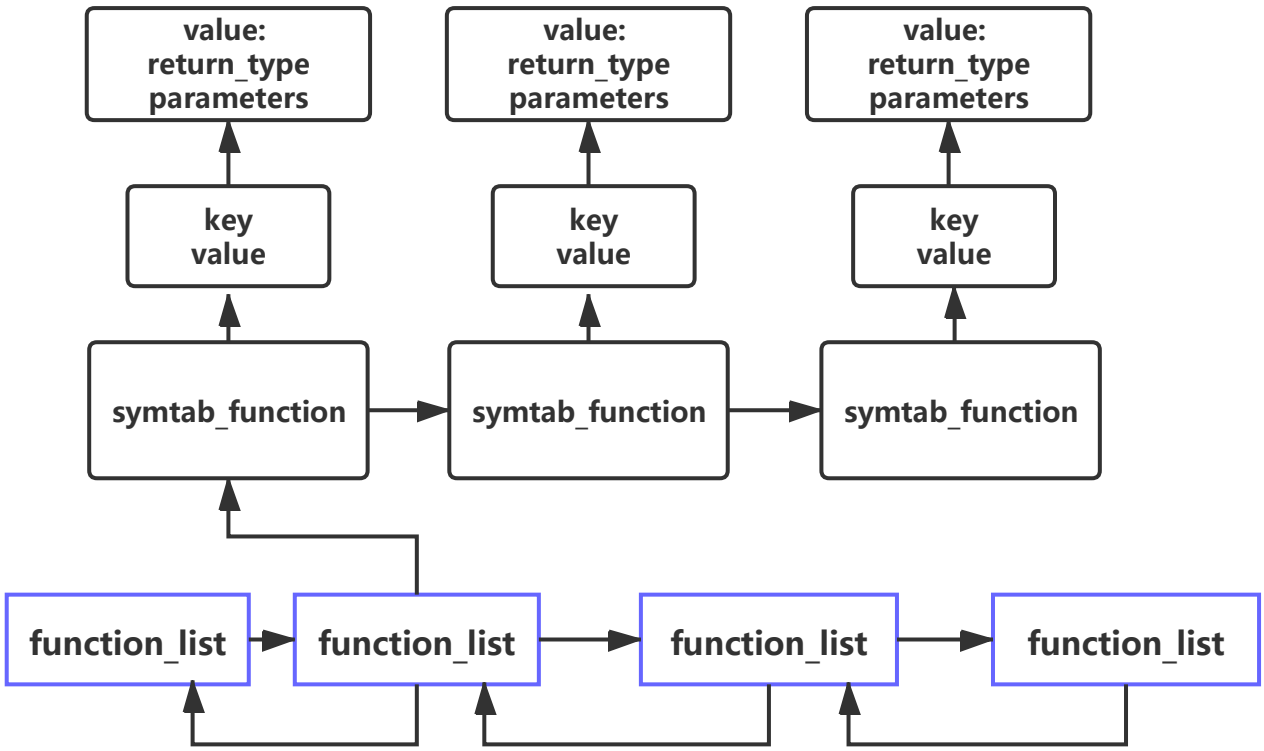
- For the variable (including struct) /function list:

I use *list* structure to implement semantic analysis, the main struct is shown beow:

Varaible(including struct):



Function:



For variable or function check, I implement on two main aspects: 1. scope checking, and 2. define checking. For the semantic analysis, I do it main on several steps:

> 1. global variable/function define
> 2. variable or variable define checking
> 3. local variable define
> 4. expression check

Main function in ast.c

- parse_tree: the entrance function for semantic.
- variable_insert: insert a new variable symtab list for scope checking.
- global_charge: handle the global checking, including the struct define, function define and global variable define.
- local_charge: handle the local checking, including the local variable define and assignment operation.
- check_exp_type: do semantic analysis for the expression.
- get_type: return the type of expressing, if not defined before, it will return "null_false".
- function_args_compare: compare and return the result of two function compare.
- array_check: check the array object, including the dimension and type checking.

Optional feature

1. scope checking: for different scopes, thr working field of variable is different.
2. char can not only appear on the assignment or function parameters, it can also work as function return type.