REPORT
Assignment 3

Symbol Table:

* For each ID, with their value and scope, a hash key is generated. IDs are saved in ‘**strTable’** hash map.
* A struct ‘**tyepEntry’** is used to store additional information about different IDs(variable & function). It has symbol_type, data_type, param & param_types array.
* The IDs are divided into two categories; variable & function. For variable, name, scope, struct ‘typeEntry’ is extended for data_type, stmbol_type. For function, ‘**typeEntry’** is extended for param & param_types array.

Error:

1. Undeclared variables and undefined functions.

If a variable or function is not found in strTable, it is considered undeclared or undefined. An error message will be generated if one is found.

2. Multiply declared variables and multiply defined functions.

After a variable or function is already declared, ST_lookup will scan strTable and provide an error message if a recently declared function or variable is already defined.

3. Function declaration/call mismatch

In function declaration, no of function parameter & the parameter types are saved in ‘**strTable**’ hash table. In function call expression, a temporary struct ‘**typeEntry**’ is created with the function call parameter. Then it is matched with the saved ones in ‘**strTable’.**

If function call parameters are more than function declaration parameter, then “Too many arguments provided in function call.” error generated. If less, then “Too few arguments provided in function call.” error generated. If all the function declaration parameter data types do not match with the function call parameter data types, then “Argument type mismatch in function call.” error generated.

4. Indexing an array variable with a non-integer type

5. Indexing an array with an out-of-bounds integer literal

6. Type mismatch in assignments

In assignment statement, left-hand side data type & right-hand side data types are found by traversing the partial AST tree created up to the point. Then they are checked whether they match or not.

If left-hand side data type is ‘int’, valid right-hand side data types are integer, integer constant, character & character constant. So if right-hand side data types includes ‘void’, then its assignment type mis-match error. Similarly for ‘char’, valid right-hand side data types are character & character constant. For ‘void’, void, integer & integer constant.

Compile & Test:

* Go to the assgn3 folder. Run 'make clean' & then 'make'.
* To run example test, give the name of the test file in place of 'input_file'.

```
./obj/scanner –ast –sym input_file.mC
Example: ./obj/scanner –ast –sym  test/cases/test.mC
```