CSCI 4160 Project5

Due: see class calendar

Description:

In this project, you are required to use symbol table to perform every basic semantic error checking listed below:

- undefined type/variable/function names
- redefined type/variable/function names.

All other semantic errors will not be covered in this project.

Classes in this project

Several classes are defined in different namespace for various purposes. Here is a short explanation of each class.

- Class absyn::Absyn and all its children are used to define types of nodes in the abstract syntax tree.
- Class template symbol::SymbolTable provides the implementation of symbol tables used in this course.
- Class types::Type and all its children are used to represent types supported by tiger languages. We are not going to use them in this project. But we need them to make sure the syntax is correct, and no change is needed for the next project.
- Structure symbol::SymTabEntry provides information that should be tied with a variable/function/type name in the SymbolTable. The SymTabEntry contains three information: the level of associated name in the source program, a pointer to the type information of the name, and a pointer to the AST node that contains the variable/function/type.
- Class symbol::Env provides the compile environment for Tiger language. It has two member data of symbol::SymbolTable<SymTabEntry>, one to store variable/function information, and one to store type information as specified by Tiger language manual. Its constructor will insert all global functions like print, ord into variable symbol table, and built-in data types like int, and string into type symbol table.
- Class semantics::TypeChecking provides functions to check semantic errors at each node in the abstract syntax tree..

Your major task is to complete the implementation of the following class.

1. TypeChecking: this class performs the basic type checking using the symbol table

You should not modify other classes/files in the project.

Tips:

1. Tiger.tab.hh, Tiger.tab.cc, lex.yy.cc files are provided by instructor in SymbolTableProject. So you don't need to use your own tiger.ll or tiger.yy files in this project. Just leave them blank. When you compile the solution, please just build the MainDriver project.

2. Every time an item is inserted into the variable or type symbol table, a SymTabEntry object should be construct. In this project, please pass **nullptr** to the second parameter when constructing a SymTabObject. Let d be a pointer to an absyn::FunctionDec object, the following statement will insert it into the variable symbol table:

```
insertFunc( d->getName(),
            symbol::SymTabEntry(
                 env.getVarEnv()->getLevel(),
                 nullptr,
```

```
                d)
        );
```

where insertFunc function is provided in the class semantics::TypeChecking

Instructor provided files in the class repository

The following files are provided by the instructor:

- SymbolTableProject folder. This contains a sample Visual Studio 2010 project.
- Description5.pdf: this file
- Rubric5.doc: the rubric used to grade this assignment.
- example.txt. sample output

How to submit

1. Once you have finished, submit the project in the following way:
    - Copy the file projects/project5/rubric5.doc from the class repository to the project5 folder in your local repository of project5. Edit the file to put your name.
    - Commit the whole project5 folder to your local repository.
    - Push all the changes to master repository on ranger.
    - **Any commit of the project after the deadline is considered as cheating. If this happens, the latest version before the deadline will be graded, and you may receive up to 50 points deduction.**
2. You can also check your overall grade by update the rubric5.doc from the repository after receiving notification from the instructor.