

Symbol Tables and the csym Compiler Framework

EECS 665 Compiler Construction

Dr. Kulkarni

Marianne Jantz

Symbol Table

- The data structures that are used by compilers to hold information about source-program constructs
- Information is collected incrementally by analysis phases and used by synthesis phases
- Entries contain information about an identifier
 - Character string (or lexeme)
 - Type
 - Position in storage

Scope

- Scope of declaration – the portion of a program to which the declaration applies
- Symbol tables need to support multiple declarations of the same identifier within the program
 - Ex: x and i – common variable names

Most-Closely Nested Rule

- If blocks can be nested, several declarations of the same identifier can appear within a single block
- Most-Closely Nested rule: an identifier x is in scope of the most-closely nested declaration of x
- Examine blocks inside-out

Block Statements Example

```
1.  { int x; int y;  
2.    { int w; bool y; int z;  
3.      ... w ... ; ... X ... ; ... y ... ; ... Z ... ;  
4.    }  
5.      ... w ... ; ... X ... ; ... y ... ;  
6.  }
```

Understanding Lab 07's Sample Output

```
| int int_glob;  
| double d_glob;  
| | main()  
| | | {  
| | |     int loc1, loc2;  
| | |     double d_arr[10], d_loc;  
| | |     d_arr[0] = 3;  
| | |     loc1 = 0;  
| | |     if(loc1 == 10)  
| | | | {  
| | | |     int b_loc1;  
| | | |     double bd_loc2;  
| | | | }  
| | | }  
| | }
```