Session 4: Detecting Simple Syntax Errors

Assignment:

- Syntax errors are very common in source programs. Suppose, a given C source program has been scanned, filtered, lexically analyzed and tokenized as that were done in earlier sessions. In addition, line numbers have been assigned to the source code lines for generating proper error messages. As the first step to Syntax Analysis, we now perform detection of simple syntax errors like duplication of tokens except parentheses or braces, Unbalanced braces or parentheses problem, unmatched ‘else’ problem, etc.

Sample code segment with numerous syntax errors

```
/* A program fragment*/

float     x1 = 3.125;;
/* Definition of function f1 */
double  f1(int int x)
{if(x<x1)
       double z;
  else z =          0.01+x*5.5;}}
     else   return z;
}
/* Beginning of 'main'     */
int main(void)
{
  int n1;  double z;
 {{ n1=25;  z=f1(n1);}
```

Recognized tokens in the lines of code

```
1
2
3  float id 1 = 3.125 ; ;
4
5  double id 2 ( int int id 3 )
6  { if ( id 3 < id 1 )
7  double id 4 ;
8  else id 4 = 0.01 + id 3 * 5.5 ; } }
9  else return id 4 ;
10  }
11
12  int id 5 ( void )
13  {
14  int id 6 ; double id 7 ;
15  { { id 6 = 25 ; id 7 = id 1 ( id 6 ) ; }
```

Sample errors detected:

Duplicate token at line 3, Misplaced ‘}’ at line 8, Unmatched ‘else’ at line 9, etc.

Note:

- ✓ Unbalanced braces or parentheses problem can be detected during tokenization in a simple way by counting the openings and closings: No prefix has more closing than openings.
- ✓ Unmatched ‘else’ problem in its simplest form may also be detected by counting ‘if’s and ‘else’s: For every ‘else’ there must be an ‘if’ that occurs earlier .
- ✓ Undeclared identifiers and duplicate identifier declarations in the same scope are detected during Symbol Table construction in a relatively easier way.
- ✓ Parsing with CFGs for checking correctness of constructs like expressions and statements will be required.