Key Word List: and, or, nah(not), Tru(True), False, import, from, class, def, if, ifel, else, for, while, return, try, except, int, String, char, bool, global, nonlocal

-Grammars for keywords, statements, and expressions overall syntax are evaluated

-A description of the semantics (some combination of the following for each grammar rule) for the following: Denotation Operational Attribute Grammar

-A list of data types and the operations related to them (each operation requiring a syntax rule) int, String, float (real number), char, bool

Identifiers:
$ - used to start an identifier
_ - used to separate string when initializing an identifier
Invalid types of Identifiers include:
sum of vals // since we can not initialize an identifier with spaces in the name
2013stats // since an identifier can not start with a digit
b - a // an operator should not be in the identifier
var-1 // we can not have a hyphen in the identifier
log2&log3 // we can not have a and sign as an identifier

Key Words - { if, ifel, else, while, for, try, except, class, bool, tru, false, return, def, global, and, or , nah, bool, switch, case, catch, int, String, float (real number), char, bool, double}
Operators - { +, -, /, %}
Literals which include:
String - {""}
Integer - {int} // ex int a, a = 10
Real - {float} // float result, result = 2.1458f
Boolean - {bool} // if true : return tru if false : return nah

Requirements:
- The variables must be declared before they are used
- The variables must have matching types when used in arithmetic expressions (this is a big part of semantic analysis called type checking that we'll cover separately)
- There should be no duplicate declarations (Pascal prohibits, for example, having a local variable in a procedure with the same name as one of the procedure's formal parameters)
- A name reference in a call to a procedure must refer to the actual declared procedure

- A procedure call must have the correct number of arguments and the arguments' types must match those of formal parameters in the procedure declaration
- Utilize the symbol table and check if the symbol exists in it

Info we need to collect about variables:
Name, category, and type. And we utilize the symbols to hold that information.

Information is stored:

symbols are stored in the symbol table by using its insert method.

If variables are not declared and stored in a variable table:
Return null or error.

For example:
Case 1   Case 2
new int a, b; | new int b;
a = 10 , b = 7; | a = 10 , b = 7;
return a; | return a;
Results in no Error |# Results in an Error since a was not declared