Target Language Syntax
CS 4300 – Spring 2017

program ::= variable_definitions function_definitions

function_definitions ::= function_head block
::= function_definitions function_head block

identifier_list ::= **ID**
::= **ID** [**INT_LITERAL**]
::= identifier_list **,** **ID**
::= identifier_list **,** **ID** [**INT_LITERAL**]

variable_definitions ::= ϵ
::= variable_definitions type identifier_list **;**

type ::= **INT**
::= **FLOAT**

function_head ::= type **ID** arguments

arguments ::= **(** parameter_list **)**

parameter_list ::= ϵ
::= parameters

parameters ::= type **ID**
::= type **ID** [ ]
::= parameters **,** type **ID**
::= parameters **,** type **ID** [ ]

block ::= **{** variable_definitions statements **}**

statements ::= ϵ
::= statements statement

statement ::= expression **;**
::= compound_statement
::= **RETURN** expression **;**
::= **IF** **(** bool_expression **)** statement **ELSE** statement
::= **WHILE** **(** bool_expression **)** statement
::= input_statement **;**
::= output_statement **;**

input_statement ::= **CIN**
::= input_statement **STREAMIN** variable

output_statement ::= **COUT**
::= output_statement **STREAMOUT** expression
::= output_statement **STREAMOUT STR_LITERAL**
::= output_statement **STREAMOUT ENDL**

compound_statement ::= **{** statements **}**

variable ::= **ID**
::= **ID** [expression]

expression_list ::= ϵ
::= expressions

expressions ::= expression
::= expressions **,** expression

expression ::= variable **ASSIGNOP** expression
::= variable **INCOP** expression
::= simple_expression

simple_expression ::= term
::= **ADDOP** term
::= simple_expression **ADDOP** term

term ::= factor
::= term **MULOP** factor

factor ::= **ID**
::= **ID** (expression_list)
::= literal
::= (expression)
::= **ID** [expression]

literal ::= **INT_LITERAL**
::= **FLT_LITERAL**

bool_expression ::= bool_term
::= bool_expression **OR** bool_term

bool_term ::= bool_factor
::= bool_term **AND** bool_factor

bool_factor ::= **NOT** bool_factor
::= (bool_expression)
::= simple_expression **RELOP** simple_expression

Where:

Entries in **boldface** are tokens
ASSIGNOP stands for the lexeme =
MULOP is one of * / %
ADDOP is one of + -
INCOP is one of += -=
RELOP is one of < > <= >= == !=
NOT stands for the lexeme !
OR stands for the lexeme ||
AND stands for the lexeme &&
FLT_LITERAL is a float constant without a sign
(at least 1 digit before and after decimal pt.; possible exponent)
INT_LITERAL is an integer constant without a sign
STR_LITERAL is a string enclosed in quotes (”), not longer than 1 line
STREAMIN is >>
STREAMOUT is <<
ID follows the usual rules for C++ identifiers, and may be any length
CIN, COUT, ELSE, ENDL, FLOAT, IF, INT, RETURN, and **WHILE**
are the keywords with those spellings
( ) [ ] { } ; and **,** are single-character tokens representing themselves

Additional lexical conventions:

Comments may be entered using either /* ... */ or //, as in real C++
Any line beginning with # (like, for instance, #include <iostream>)
is also considered a comment