German University in Cairo
Media Engineering and Technology
Assoc. Prof. Haythem Ismail

Compilers Lab, Spring term 2019

Milestone 1

Lexical Analysis for Nim in ANTLR

Please read the following instructions carefully:

- Read Rules & regulations first
- It is **YOUR responsibility** to ensure that you have:
  - Submitted before the deadline (6th of April).
  - Submitted the correct file(s).
  - Submitted the correct file(s) names.
  - Submitted the correct output format that matches each task.
  - Submitted correct logic of the task as it will be tested both publicly & privately.
  - Submitted your code in the format TeamName_milestone_1.zip where TeamName is your team name for example compteam_milestone_1.zip google form link https://goo.gl/forms/GrSDwLrDZb3ZTB5o2.

  .
- Good luck! =D

1 Lexical Analysis for Nim using ANTLR

In this part, you are required to implement the Lexical Analysis for the language "Nim" using ANTLR4.

- Follow the exact file name:
- "milestone_1.py", should contain the code to tokenize the the input given.
- "milestone_1.g4", should contain the lexer rules for the grammar.
- "milestone_1_result.txt", should conatin the tokens.
- The ANTLR file (.g4) should contain the regular definitions needed for Tokenization.
- You should submit all files contains your python code for the solution.
- All files should have the extension ".py" & the "main" method should be in a file with the correct name.
- You should make sure that the output is produced in a text file with correct name.

```
// test.g4 file
grammar test;
...
VARIABLE : 'var';
IDENTIFIER : LETTER+ ('_'(LETTER | DIGIT))*;
COMMA : ',' ;
EQUALS_OPERATOR : '=';
DIGIT : [0-9] ;
...
```

The Python file should contain the code to tokenize the input given & outputs the tokens, for example "var x, y = 3" would be:

(All tokens & matching strings must be separated by space)

```
var VARIABLE
x IDENTIFIER
, COMMA
y IDENTIFIER
= EQUALS_OPERATOR
3 DIGIT
```

1. Create the regular definitions for the language Nim.
   Follow Nim's manual https://nim-lang.org/docs/manual.html#lexical-analysis to cover the following:

   - Indentation.
     - Where it will be represented by four spaces.
   - Comments
   - Multiline comments
   - Identifiers & Keywords
   - String literals
   - Triple quoted string literals
   - Raw string literals
   - Generalized raw string literals
   - Character literals
   - Numerical constants
   - Operators
   - Other tokens

2. You must follow the same naming convention as follows:

- AND
- VARIABLE
- ADDR
- AS
- ASM
- BIND
- BLOCK
- BREAK
- CASE
- CAST
- CONCEPT
- CONST
- CONTINUE
- CONVERTER
- DEFER
- DISCARD
- DISTINCT
- DIV
- DO
- ELIF
- ELSE
- END
- ENUM
- EXCEPT
- EXPORT
- FINALLY
- FOR
- FROM
- FUNC
- IF
- IMPORT
- IN
- INCLUDE
- INTERFACE
- IS
- ISNOT
- ITERATOR
- LET
- MACRO
- METHOD
- MIXIN
- MOD
- NIL
- NOT
- NOTIN
- OBJECT
- OF
- OR
- OUT
- PROC
- PTR
- RAISE
- REF
- RETURN
- SHL
- SHR
- STATIC
- TEMPLATE
- TRY
- TUPLE
- TYPE
- USING
- WHEN
- WHILE
- XOR

- YIELD
- IDENTIFIER
- LETTER
- DIGIT
- INT8_LIT
- INT16_LIT
- INT32_LIT
- INT64_LIT
- UINT_LIT
- UINT8_LIT
- UINT16_LIT
- UINT32_LIT
- UINT64_LIT
- FLOAT32_LIT
- FLOAT32_SUFFIX
- FLOAT64_LIT
- FLOAT64_SUFFIX
- FLOAT_LIT
- EXP
- INT_LIT
- HEX_LIT
- DEC_LIT
- OCT_LIT
- BIN_LIT
- HEXDIGIT
- OCTDIGIT
- BINDIGIT
- EQUALS_OPERATOR
- ADD_OPERATOR
- MUL_OPERATOR
- MINUS_OPERATOR
- DIV_OPERATOR
- BITWISE_NOT_OPERATOR
- AND_OPERATOR
- OR_OPERATOR
- LESS_THAN
- GREATER_THAN
- AT
- MODULUS
- NOT_OPERATOR
- XOR_OPERATOR
- DOT
- COLON
- OPEN_PAREN
- CLOSE_PAREN
- OPEN_BRACE
- CLOSE_BRACE
- OPEN_BRACK
- CLOSE_BRACK
- COMMA
- SEMI_COLON
- STR_LIT
- Character literals
- CHAR_LIT
- TRIPLESTR_LIT
- TRIPLESTR_ITEM
- RSTR_LIT
- GENERALIZED_STR_LIT
- GENERALIZED_TRIPLESTR_LIT