Design and Implementation of Programming Languages
(CMSC 124 Project)

Phase 2:

Design the grammar of your own programming language and create a mini-compiler for it.

Requirements:

- Create programs that will be run to implement your Compile/Run button:
  - Create your own scanner for identifiers and numbers.
  - Create a recursive descent parser for your designed grammar.

  Notes:
  - When you compile the source code (written in your programming language), you will run your parser, and the parser calls for the scanner to supply one token at a time.
  - When you run the source code (written in your programming language), you will compile it first, if the source code has been modified and not yet recompiled. Otherwise, run immediately the compiled version of the source code.

- Provide the necessary basic functionalities:
  - Printing some texts on the screen like prompts for the user to respond. For example, "Input the 1st number: "
  - Accepting inputs from the keyboard like numbers, characters, or strings.
  - Performing mathematical expressions and displaying the results in the screen.

  Note:
  - Plus points for those who can add more operations like loop and if-statements.

- Apply a simple code generation by translating your source code into assembly language code (MIPS or 80x86 family), which can be fed into the appropriate assembler to produce the program's object code. No need for code optimization.

  Notes:
  - You may run the Assembler by feeding it with the assembly codes that was generated by code generation process at the DOS prompt, and then call the Linker to create the executable code.
  - Execute your program from the DOS prompt.
  - Alternatively, you may run the assembly code in any existing IDE.
  - Plus points for those who can run your source program inside your IDE.

Points:

40% grammar (group) – programming language design
20% parser (group) – recursive descent parsing implementation of your grammar
10% scanner (group) – implementation of your FA for identifiers and numbers
10% code generation (group) – generation of the assembly code of a source code
10% code execution accuracy (group) – output of running the code (based on your own language)
10% peer-evaluation (individual) – rate of member's contributions