**Project Report**

Hui Hao & HHao@my.harrisburgu.edu

Harrisburg University of Science and Technology

I used a python to write a simple compiler. It can handle, numerical inputs, basic operations. Logical statements (If then, While) Comments, Print texts and numbers.

## Compiler Overview

The compiler has the following steps. First, given the input code of any format, it will first check/recognize the syntax and tokenize the input. Second, it will *parse* the tokens and use given grammar to make sure they are in an order and no grammatical errors that is allowed in our language. Last the Emitter will compile the final C code. And save it as ‘input\_filename.c’

## Parser Overview

The parser is the component that will make sure the code follows the correct syntax. It does this by looking at the tokens, one at a time, and deciding if the ordering is legal as defined by our language.

## Grammar Overview

Grammar regulates the statements logics allowed.

## Emitter Overview

In each function of the parser, emitter is evoked to produce the appropriate C code. The emitter is effectively just appending a bunch of strings together while following along the parse tree. For each grammar, we will figure out how it should map to C code.

The driver code is to have an input code to print out the first n elements of a Fibonacci sequence.

Driver code:

Python main.py input.txt

**GitHub**

https://github.com/graciehao25/toy\_compiler

**Reference**

1. Austin Z. Henley, Let’s make a Teeny Tiny complier, <http://web.eecs.utk.edu/~azh/blog/teenytinycompiler1.html>; <https://github.com/AZHenley/teenytinycompiler>
2. Marcelo Andrade, <https://blog.usejournal.com/writing-your-own-programming-language-and-compiler-with-python-a468970ae6df>