Parser is a simple intended to parse 32-bit ARM assembly code. It combs through the source code and creates objects depending on what it encounters. Once the source code is transformed into a list of objects, it can be more easily worked with to discover patterns. It uses Python’s type hints to be more transparent.

Locations are spots in the code that can be referenced and jumped to. Example: .LC0 and main.

IntegerLiterals are integers. In 32-bit syntax, these are prefaced with a “#”

StringLiterals are strings. These are prefaced by a location and .ascii.

Registers are ARM registers.

Instructions are assembly instructions. They are composed of the string representation and a list of the arguments, which are further parsed into IntegerLiterals, StringLiterals and Registers.

There are certain features that need to be added such as parsing memory locations, which are found inside straight brackets (“[“ and “]”). Since these are offset to register values or memory locations, this will take some work.

Two sample files secure.s and insecure.s were prepared with <https://godbolt.org/> using the 32 bit ARM gcc trunk compiler.

Useage: python3 Parser.py [target file]