A common occurrence in the design of a programming language is the implementation of features of commonly recurring programming patterns used by programmers. Examples of some features include for loops, incrementing a variable by one, and data structures. The use of design patterns are one such common technique of many software engineers in the design of new systems. The problem is then to design a new programming language and a compiler for the language which implements features to simplify the use of software design patterns.

The parser will generate an abstract syntax tree. Each node will need to have corresponding code generated for it. Said code will be turned into llvm-ir code which can then be turned into an executable.

The only things necessary to run is a unix like environment, llvm, and llvmpy. We have 3 subsystems Lexer, Parser, and CodeGen.

We have implemented a problem language called CNatural.

- Make functions
- Do math operations on integer and floats
- Create arrays of integers or floats
- Output to console
- Use if and else to control the program
- Use while as an iterative control
- Make recursive function calls
- Use Boolean operators

The material presented in this poster is based upon the work supported by Leo Shao and Erik Edrosa. I am thankful to the help that I received from my group members, our mentor, Ziyuan Meng.