

Compilers

Phase 2

Parser

Due – September 23

30 Points

Use Bison to implement the parser for the Modula-2 language. Read Chapter 2 in *Modula-2: Language Description and Compiler* for more information.

You should do the following:

- Create a file named `m2.y` that contains the Bison description of the Modula-2 language. You should not attach any semantic actions to your grammar for this phase.
- Handle all command line options to the compiler. These are described in Chapter 2.
- Create a makefile or project for your compiler.

Create your parser using the command `bison -vd m2.y`. This command creates the following files:

- `m2.tab.h` – this file contains the definitions for the tokens and any other definitions you need. This file needs to be included in the `m2.l` file.
- `m2.tab.c` – this file contains the source code for the compiler. The parser function is named `yyparse()`. This file needs to be compiled to produce the compiler.
- `m2.output` – this file contains information produced by Bison about the parser. If you have conflicts (shift/reduce or reduce/reduce), you will find information about the conflicts in this file.

The software needed for a PC can be found on my home page. There are also links to other locations for additional software.

Files to turn in:

- `makefile` – created by you to compile your compiler.
- `m2.y` – created by you for this phase.
- `m2.tab.c` – created by Bison.
- `m2.tab.h` – created by Bison.
- `m2.l` – same file you used for the first phase.
- `main.c` – a stub for this file is available from the course home page.
- `README` – describes how to compile and run your compiler.

Hints:

- Make sure you use an ambiguous grammar for expressions. The ambiguities are resolved using the precedence specification in Bison: `left`, `right`, and `nonassoc`.
- You must use the `%union` in your Bison file. You must define the following fields:
 - `int_val` – an integer
 - `name_ptr` – a pointer to a character.
- When recursion is used in the rules, you may use either left or right recursion. I always use left recursion.

- It is sometimes helpful to add key words to the precedence rules. This may eliminates some conflicts.
- It is possible to get a linking error because of `alloca`. This can be solved by including `malloc.h` in the `m2.y` file.
- An outline of the main function is given on the course home page. Use it if you want to.