CSE4120 Fundamentals of Compiler Construction (Fall 2022)
Homework 2: LALR(1) Parser

Handed out: Nov 20, Due: Dec 14, 11:59PM (KST)

1. Problem Description

In this programming assignment, you will build a LALR(1) Parser for the C- language, defined in Appendix A of the textbook.

2. Your task and requirements (Read Carefully!)

(1) You should first read carefully Chap5.5-6 (pages 226-245) and Appendix A.2 (pages 492-496) for this programming assignment. Next, your task is to modify `global.h`, `scan.h`, `util.h`, `util.c` and `tiny.y` (shown in Appendix B) to build your LALR(1) parser for C- language. Note that you can use your own scanner which was built in the first assignment. Note also that all files for TINY parser will be given.

(2) In addition to the program, you should also write a **Makefile**. The TA will build your code by running "make". It should create the binary files, **`hw2_binary`**.

(3) Your program should read the source code and output the result to files. **If the input file name is `test1.c`, then your output file should be named `test1_20200001.txt`.** The red numbers should be changed to **your student ID**.

(4) Two example codes and corresponding results will be given.

(5) You have to do this assignment on the Unix or Linux system. Also, you should use C, Flex, and Bison to implement the LALR(1) Parser.

NOTE: If you do not have Unix or Linux system, I encourage you to use **Google Colab** instead. You can easily install Flex in your Colab project using the following commands:

```
%%shell
apt-get install bison
```

3. Submission

You should submit your codes and the Makefile. You do not need to submit compiled binary files. Combine your files into a zip file named cse4120_hw2_**20200001**.zip. The red numbers should be changed to **your student ID**. Submit your files on the cyber campus.

4. Evaluation Criteria

(1) Your LALR(1) parser should generate the abstract syntax tree if there is no syntactic error in the source code.

(2) Your LALR(1) parser should be able to handle syntactic errors correctly.

5. Notes

- You must write your own code. You may discuss ideas with other students but must not copy their work. Similarly, you can find ideas on the Internet, but must not copy the code from the sources obtained from the Internet.

- Our department uses a software that detects duplicate codes. Since the software uses an assembly code level detection, just changing the variable names will not bypass the software. Make sure you write your own code from the given file set. Then, you will have no problem.

- Your program should be able to compile and run on the Linux system. Make sure you check before you submit your work. The TA will download your code, run “make” and execute your binary files to check if they produce correct outputs.

6. Late Policy

No late turn-ins will be accepted for this assignment.