**파싱테이블 만들기 Project**

**1. Lex 와 Yacc 를 사용하여 파서 만들기**

**-** 간단한 것부터 시작하여 복잡한 것까지 계속적으로 파싱테이블을 만들면서 프로젝트를 진행할 예정입니다. **최종 목표는 C 언어 전체의 파싱테이블을 만드는 것**입니다. 작은것부터 차근차근 하겠습니다.

**2. ANTLR 을 이용하여 만들기**

- 기존의 Lex 와 Yacc를 이용하는 것이 아닌 다른 방법으로도 시도해 보려고 합니다. 함께수업을 수강하는 다른 학생들에게도 또 다른 재밌고 유익한 프로그램이 많다는 것을 알려주고 함께 사용방법까지 숙지하고 싶습니다. ANTLR 은 간단한 문법을 통해 직관적인 오토마타를 생성해주고 그 오토마타의 규칙을 C# C++ JAVA 등의 파일로 생성시켜주는 프로그램입니다. 이 프로그램을 이용해서는 **C 언어 전체에 대해서 어휘분석**을 하려고 합니다. ANTLR 에 대해서 간단히 소개하겠습니다.

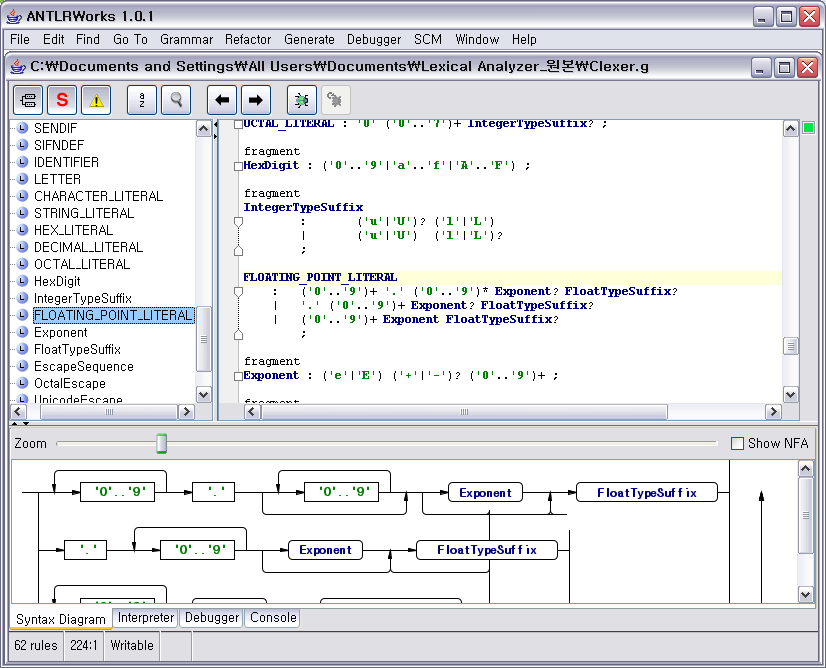
* **ANTLR이란 무엇인가? (**[**http://www.antlr.org**](http://www.antlr.org)**)**

- 인식기(recognizer), compiler, translator를 구성하기 위한 framework를 제공해주는 언어 인식툴

- 문장이 해당 언어에 적합한가 여부를 결정하는 프로그램을 생성하고, 중간 형식 트리 구성, 트리워킹, 번역과 자동 에러 검출, 보고 등에 대한 지원을 제공

- LL(k) 파서이면서 테이블 기반(Table-Driven)이 아닌, 사람이 작성한 것과 유사한(Recursive Decent Parser) 코드를 생성하기 때문에 생성된 코드를 쉽게 이해 및 수정

- 기본적으로 LL 파서이므로 left Recursion을 직접 제거해야하고 일부 문법에 Lookahead를 직접 지정해주어야 한다는 것이 단점



* **왜 ANTLR를 이용하는가?**

ANTLR은 손으로 했던 것을 정확하게 자동화하기 위한 목적에서 생겨났다. 즉 NFA 를 이용한 직관적 분석을 실시간으로 피드백을 받을 수 있어 그 논리가 옳은지 바로 확인 할 수 있고 쉽게 이해할 수 있다. 이는 수정 및 보완 역시 용의하게 해준다는 장점을 함께 가져다 준다.

ANTLR의 특징은 다음의 기본적인 요구를 만족한다.

1. 이해하기 쉬운 인터페이스
2. 문제를 해결하기 위한 강력한 툴,
3. 작업의 자동화
4. 쉽게 폴더화 하여 결과물을 생성

- ANTLR은 렉서 및 파서를 명세하여 주는 툴이다. ANTLR은 직관적으로 이해 가능한 NFA 구조를보여주어 보다 쉽고 정확하게 에러를 확인 하고, 수정 보완 할 수 있다.

- ANTLR은 Java, C#, C++, 등의 다양한 언어를 지원해준다.

- ANTLR은 다양한 언어와 호환할 수 있는 source code를 생성해 주고, 사용에 있어서 절대적으로 전혀 제약사항을 가지고 있지 않다.

위와 같은 장점 때문에 현재 ANTLR 은 이러한 분석 프로그램을 만드는데 많이 이용되고 있다.