# HYU-ELE4029, Compilers

# 1. Scanner

컴퓨터소프트웨어학부

2018008013 김영중

1. Compilation method and environment

- Env: Windows 10 WSL – Ubuntu 18.04
- GNU Make 4.1. gcc 7.4.0
- make 명령어를 통해 scanner_cimpl, scanner_flex 실행파일 생성

2. Scanner implementation – C

TINY 컴파일러는 globals.h에서 Token을 정의, scan.c에서 scanner를 구현한다.

Scanner의 경우 LL(1) Grammar의 predictive-parsing을 기반으로 하고, input character에 따라 START, DONE, IN token으로 구별된다.

START token은 가장 시작에 설정되는 token으로 지금 읽은 character가 token의 시작임을 나타낸다. 이에 따라 숫자이면 NUM, 영어이면 ID, 이외에 single character symbol "+-*(){}[];,"과 multi-character symbol "==, !=, <=, >=, /**/"로의 진입점 INEQ, INNE, INLT, INGT, INOVER로 분기된다.

이렇게 읽어드린 character는 tokenString에 한 글자씩 저장된다. multi-character symbol의 처리 과정에서는 두 번째 character가 적절치 않아 single character "=<>/"로 인식되었을 때, tokenString에 저장하지 않고 stream을 한 칸 복원한다.

이렇게 각각의 token이 인식되면 state를 DONE으로 바꿔 token 처리를 종료한다. ID인 경우 reserved keyword "if, else, when, return, void, int"인지 확인하는 루틴을 거쳐 최종 반환하게 된다.

주석 /**/의 경우에는 /의 인식에 따라 INOVER로 전이, *가 확인되면 INCOMMENT로 전이하고, 아니면 single character symbol "/"로 인식하여 루틴을 마무리한다. INCOMMENT 상태에서는 tokenString을 작성하지 않고 continue 하다가, *이 발견되면

INCOMMENT_로 전이한다. INCOMMENT_는 /를 발견하면 주석의 종료를 확인하여 START로 전이, 처음부터 다시 심볼을 찾아 나가고, 발견하지 못하면 다시 INCOMMENT로 전이하여 주석 상태를 유지한다.

3. Scanner implementation – Flex

Flex를 활용한 케이스는 "./1_Scanner/lex/cminus.l"을 통해 확인할 수 있다. Tiny.l을 대부분 차용하였으며, C-minus의 spec에 맞게 keyword 6개와 symbol 19개를 parsing할 수 있게 두었고, 주석 /**/의 경우는 /*를 parsing하여 callback에 */에 맞춰 종료할 수 있게 구성하였다.

4. Samples

"./1_Scanner/samples"에 test.cm, test2.cm, test3.cm을 두었다. 처음 두 개는 제시된 sample이며, 모든 token에 대한 parsing 여부를 검증하기 위해 부족한 token으로 세 번째 test3.cm을 구성하였다.

scanner_cimpl의 결과는 "./1_Scanner/samples/result*.txt", scanner_flex의 결과는 "./1_Scanner/result_flex*.txt"에서 확인할 수 있다.