**컴파일러의 기초: Project #3 for Semantic Analysis**

**Due date: 12월 1일 수요일 23:59**

이번 프로젝트의 목적은 lex와 yacc utility를 이용하여 (C의 feature를 가감한; subc라 부른다)에 대한 semantic analysis를 구현하는 프로젝트이다.

프로젝트의 시작에 앞서 강의 교재(Ch 7. yacc, Ch 8. semantic analysis)와 주어진 subc문법에 대해서 철저하게 공부하는 것을 강력하게 권장한다. 이번 프로젝트에서 구현한 사항들이 프로젝트 4에서도 직접적으로 사용되기 때문에 전체적인 디자인이 매우 중요하다. 디자인에 대한 계획을 세웠다면 문서에 언급된 제한 사항을 만족하는 semantic analysis 를 구현하도록 한다. 구현 방법과 제한사항, 문법 등은 다음과 같다.

* **How to do Project #3?**

Environment: 이전 환경과 동일, flex, bison 사용

Makefile: 이전에 주어진 형태에서 바꿔서 사용

**semantic check code**

subc.y 문법의 각 terminal과 nonterminal 사이의 적절한 위치에 action (C코드)를 삽입해서 아래 주어진 제한 사항들을 체크하고, 만족하지 못할 경우 화면으로 출력하도록 한다.

**에러 메시지 포맷: {filename}:{line\_num}: error:{description}**

(Project #2에서 REDUCE할때 출력되던 메세지는 더이상 출력 X)

* **Restriction of this project**

이번 프로젝트는 다음과 같은 범위 정도에서 체크 및 구현하도록 한다.

각 제한사항에 대한 자세한 내용과 예시는 ppt 및 error message.docx를 참조하도록 한다.

* Undeclared Variables & Functions
* Redeclaration
* Type Checking
* Structure & Structure Pointer Declaration
* Function Declaration
* **Grammer**

프로젝트 2의 문법에서의 anonymous structure declaration, binary의 “\*”, “/”, “%” 연산 등이 제거되고 VOID 토큰, NULL 토큰, pre-increment/decrement 연산이 추가되는 등의 수정사항이 있다.

본인의 설계를 위해서 문법 구조를 변경해도 가능하나, 앞에서 언급된 제한 사항에 대해서는 모두 체크할 수 있도록 구현해야 한다.

**수정된 문법은 따로 grammar.txt 파일로 제공한다.**

* **제출 사항**

**소스코드**

subc.l, subc.y, subc.h, hash.c, Makefile 등 작성한 소스코드

readme 파일

**결과 보고서** (채점 시에 참고할 수 있도록 간단히 작성하면 됨. 4장 이내)

구현 방법, 구현 내용, 문제점(각 구현 제한사항 별로 언급바람)에 대해서 작성한다.

**project3\_학번.zip**으로 압축하여 eTL을 통해 제출한다.

**제출 형식을 지키지 않는 경우 각각 5% 감점 (파일 이름, 출력)**

* **TA Contact**
  + 조중하 (301동 819호)
  + E-mail: zoonghi@snu.ac.kr