컴파일러 (2023,1학기)

Project 3 (Modified Mini-C Parser) Due 6/1일(목) 24시

(Modified) Mini C를 위한 Parser를 과제2에서 작성한 scanner프로그램을 이용하여 scanning한 token을 bison (YACC)을 사용하여 구문 분석하라.

[제출 내용]

1) **전체 소스코드를 Zip 하여 제출**

   Souce codes: main.c, parser.y, scanner.l, symtable.c, reporterror.c, tn.h, glob.h 등.
   (주의사항: lex.yy.c 와 parser_tab.c 도 제출)

2) **입출력 내용**: (test1.mc, 출력 결과 1), (test2.mc, 출력결과 2), ...
   문법 오류 없는 것(4 개이상), 문법 오류 있는 것(4 개이상) **입출력을 Zip 하여 제출**
   <*** Output 의 경우 과제물에 명확히 정의되지 않은 부분이 있어 본인이 정의한 것이 있을 경우 그 내용을 기술하고, 해당 output 부분에 표시. ***>

[과제 내용]

1) scanner 와 parser 에서 발생한 모든 error 에 대하여 가능한 한 의미 있는 error 메시지로 출력하고, 프로그램 종료 전 전체 에러의 개수를 출력하시오.

예) line 3, 10 에서 error 발생했다고 가정했을 때,

```
***MiniC parsing begins
3                    line 3 에서 발생한 error type 에 따른 message 출력
10                   line 3 에서 발생한 error type 에 따른 message 출력
Parsing ends.***

2 error(s) detected
```

만일 error 가 하나도 없이 구문 분석이 끝났을 경우는, 다음이 출력된다.

```
***MiniC parsing begins
Parsing ends.***

0 error(s) detected
```

2) 변수(즉, Identifier)가 선언되었을 때, 선언된 위치의 라인 넘버를 HStab(해쉬 심볼테이블)에 저장하라. 구문분석이 끝나면, HStab 에 저장된 모든 identifer 에 대한 속성(identifier 이름, identifier 의 유형(**scalar 변수/array 변수 등, 함수 이름, 함수 파라메터), identifier 가 함수 이름인 경우에는 반환 값,** line number)을 과제 1 에서처럼 출력하라.

- **출력 양식은 자유.**
- Identifier 의 정의는 과제 2 와 동일함.

- scanner.l 에서 ID 인 경우 string 을 HStab 에 저장(즉, ST 의 인덱스를 저장)하고, **Parser.y 에서 identifier 의 유형을 좀더 세밀히 분류하여 HStab 의 해당 Identifier 의 속성을 업데이트 함.**
  (예, ID 인 경우 => integer scalar 변수, integer array 변수, void 함수명, integer 함수명, ...함수의 ..type 의 파라메터 등)

  출력 예) HASH TABLE 출력 양식은 자유이다.

  [[HASH TABLE]]
  Hash Code 10: (abc: integer scalar variable, line3)
  (bca: integer array variable, line6)
  Hash Code 20: (f: function name, return type = void, line2)

[과제3의 채점을 위한 주요 평가기준]

1) **일반적인 사항 (10점)**
- 프로그램 source file이 분리되어 있는가? (th.h glob.h, symtable.c, main.c reporterror.c scanner.l parser.y ... 파일이름은 약간 달라도 됨.)
- 입출력 file 빠짐없이 제출하였는가?
  - MiniC 문법 오류 없는 경우 4개 **MiniC 프로그램**과 그에 대한 출력
  - MiniC 문법 오류 있는 경우 4개 **MiniC 프로그램**과 그에 대한 출력
- 코딩 스타일이 적절한가? (각 파일의 앞부분에 파일이 담고 있는 프로그램에 대한 주석, 각 서브 function에 대한 주석 등, 적절히 blank line, Indentation을 넣었는가?)
- MiniC 문법 오류 없는 경우 4개 **MiniC 프로그램**과 그에 대한 출력이 맞는가?
- MiniC 문법 오류 있는 경우 4개 **MiniC 프로그램**과 그에 대한 출력이 맞는가?

2) **내용 구문분석 결과가 정확한가?** (→학생이 제출한 source code를 직접 실행한 결과로써 채점)**(10점)**
- MiniC 문법 오류가 없는 프로그램 (예, prime.mc)
  - Error 0로 표시
  - HS symbol Table 출력 내용
- MiniC 문법 오류가 있는 프로그램 (예, prime.mc에 적절한 error를 넣을 것임)
  - 적어도 하나의 Error를 detect하고 메시지를 source line과 함께 출력하는가?
  - 정상적으로 프로그램 종료가 되는가? (즉, core dump등 비정상적으로 종료되면 안됨)
  - HS Symbol Table 내용은 check하지 않음.

3) 주의사항
- 본 과제는 **Modified MiniC** 임. (즉, 변경된 내용에 맞추어 문법을 디자인해야 함. 필요하다면 키워드 추가 등)
- 토큰명 모두 대문자로
- **참고: 과제 1 의 경우 token type 을 enumerated type 으로 define 하였으나, 과제 2 의 경우는 parser.y 에서 자동적으로 생성되는 parser_tab.h 를 이용하여야 한다.**

<UNIX: YACC>	<BISON>
$ yacc -d parser.y	C:₩> bison -d parser.y
y.tab.c, **y.tab.h → cp y.tab.h tn.h**	parser_tab.c, **parser_tab.h → copy parser_tab.h tn.h**