

< 과제 3 : Yacc Programming >

1. 개요

Yacc Program 은 Lex Program 에서 특정부분을 분석하며 읽는 내용을 넘겨받아서 특정 이벤트를 발생시킬 수 있다. 이 기능을 이용하여 읽어 들인 부분이 어떤 기능을 하는지를 분석하고 각 기능들에 대한 카운트를 하여 출력하는 프로그램을 구현하도록 한다. ANSI C 문법을 바탕으로 구현한다.

2. 순서

- A. Yacc Program 을 실행 시 C 코드를 읽어오게 된다.
- B. 코드의 내용을 분석한다. (test.c 파일의 내용을 읽어 들임)
- C. 각 코드에 나오는 특성에 따라서 변수에 카운트를 한다.
- D. 카운트 된 값을 다음과 같이 화면에 출력한다.

```
function = 3
operator = 15
int = 5
char = 2
pointer = 1
array = 2
selection = 2
loop = 2
return = 1
```

3. 요구사항

함수 정의, 일반 변수 선언문, 포인터 변수 선언문, 배열 변수 선언문, 수식, 함수 호출, 선택문, 조건문, 반복문, 리턴문 카운팅 확인

A. Function

함수 정의, 함수 사용, printf 와 같은 내장 함수 사용 횟수를 카운팅.
테스트케이스에 함수가 선언만 되고 정의가 안된 경우는 없습니다.
함수가 전방선언 되고 뒷부분에 정의 된 경우 정의된 것 1 개만 카운팅. sizeof 는 함수가 아닙니다.

B. Operator

*, -, +, /, +=, -=, ||, &&, ^, & 등 이항 연산과 관련된 operator 들을 카운팅 하는 것입니다. a++; 의 경우 의미가 이항 연산자 a+=1 과 동일하므로 1 개로 카운팅.

operator.txt 에 적혀있는 것 외에 연산자는 카운팅하지 않습니다.

단항연산자(부호 +/- 등 ...)는 카운트 하지 않습니다.

? : 는 삼항연산자입니다. 카운팅하지 않습니다.

C. Int

Int 로 선언된 변수 개수. 파라미터로 선언된 Int 개수도 count 해야 합니다. Int a,b 는 int 개수 2 증가입니다.

D. Char

Char 로 선언된 변수 개수. 파라미터로 선언된 char 개수도 count 해야 합니다.

E. Pointer

Pointer 로 선언된 변수 개수.

Int * a; 는 int 개수, pointer 개수 모두 증가.

int **a 인 경우 포인터 개수 하나만 증가하면 됩니다.

파라미터로 선언된 pointer 개수도 count 해야 합니다.

포인터 함수 `int (*p)(int, int)` 는 포인터 개수, 함수 개수 모두 증가입니다. 이때 int 로 선언된 변수가 없으므로 Int 값은 증가하지 않습니다.

F. Array

배열로 선언된 변수 개수.

`Int a[3]`은 int 개수, 배열 개수 모두 증가.

또한 `int a[3] = {0}`과 같은 것도 처리가 가능해야 함.

2 차원 배열도 Array 1 증가

파라미터로 선언된 Array 개수도 count 해야 합니다.

G. selection

선택문 개수.

선택문이란, `If(){} , switch(){} ,` 단 `else` 문, `else if` 문 제외.

또한 `return a>b?a:b;`도 선택문이 아님.

H. 반복문

반복문 개수.

반복문이란 `While(){} , do{} while() , for(){} .`

I. 리턴문

`return` 의 개수.

단, `printf("return ;");`을 했을 때 카운팅 하면 안 됨.

문법상 `return` 역할을 할 때만 카운팅.

J. 변수의 선언 위치 자유

제공된 ANSI C 문법을 수정하여 기존의 변수의 전방 선언만 허용하는 문법을 변수의 선언이 자유롭도록 문법 수정.

K. 주의사항

- 1) 이 과제는 문맥상의 오류를 찾는 과제.

`int a=4; int a=10;` 과 같은 문장은 실제 컴파일 때는 에러가 뜨지만 여기에서는 메모리 상의 문제가 아닌 문맥상의 오류만 찾으므로 에러가 뜨지 않아도 됩니다.

- 2) Overflow 나 `int a = 2.4;` 과 같은 에러는 잡지 않습니다.
- 3) 테스트 파일이 `#include<stdio.h>` 같은 헤더 파일로 시작할 수 있습니다.
- 4) 테스트 파일이 `#define a 5` 와 같이 `define` 처리가 가능해야 합니다. 이때 마지막 파라미터가 상수인 경우만 구현하시면 됩니다.
- 5) `auto, break, case, const, continue, default, double, enum, extern, float, goto, long, register, short, signed, sizeof, static, struct, typedef, union, unsigned, volatile` 처리가 가능해야 합니다.
- 6) 변수에 값을 대입할 때, 10 진수 뿐만 아니라 16 진수, 8 진수 또한 가능해야 합니다
- 7) 테스트 파일에서 `for, while, do while, if, switch` 등 에는 `{}`가 있을 예정입니다.
- 8) 구조체로 변수를 선언 할 수 있어야 합니다.
- 9) 테스트 파일에서 함수 선언은 기본 자료형인 `void, int, float ...` 으로 할 예정입니다. `struct` 와 같은 임의의 자료형으로 반환하는 함수는 없을 예정입니다. 단 파라미터에는 `struct` 로 선언된 자료형이 들어 갈 수 있습니다.
- 10) 테스트케이스에 주석문(`//, /* ... */`)이 있습니다.
- 11) 테스트케이스는 에러없이 실행 되는 C 코드입니다.
- 12) 통상적인 C 언어가 아닌 ANSI C 언어에 특화된 문법(`int a()[]()[]{...}` 등)은 테스트케이스에 없습니다.

- 13) for(i=0; i< 10; i++){ func() }의 경우 함수는 한번만 카운트 됩니다.
- 14) 자료형 변수의 경우 명시적으로 선언된것만 카운트 됩니다.다음과 같은 경우 int 는 2 로 카운트 됩니다.

```
typedef struct { int x, y; } EX; EX a, b;
```

4. 문서화

구현한 부분에 대한 **코드 분석** 및 **사용 이유** 등의 설명을 **자세히 문서화**하도록 한다. 문서 자체의 비중이 코드의 비중보다 더욱 큼을 유념.

5. 제출 결과물

A. 제출 파일

- 1) hw3.l hw3.y hw3.pdf hw3.tex, tex 에 첨부된 이미지 파일.

B. 보고서 내용

- 1) yacc 에 대한 설명 (동작 방식)
- 2) Lex, yacc 코드 옮겨 적은 후 주석 작성

C. 제출 마감 시간 및 장소

- 1) 제출 마감 시간 : 2 주

4 월 30 일 00 시(1,2 분반)

5 월 1 일 00 시(3,4 분반)

2) 제출 방법 : submit pem_ta hw3_

(_ == 1 분반 : a, 2 분반 : b, 3 분반 : c 4 분반 : d)

6. 유의 사항

A. 채점 기준

- 1) yacc 에 관해 어느정도 이해했는가?
- 2) 프로그램의 구동은 잘 되는가?

B. 감점 사항

- 1) 부정행위 발견 시 관련 학생 모두 0 점 처리하며, Latex 파일에서 과제에 대한 이해도 부족으로 부정행위로 간주될 경우 큰 감점
- 2) 프로그램의 출력이 제시된 출력과 다르면 0 점

C. 질문 사항

- 1) 제목 첫 글자에는 pl 추가
- 2) 질문 : pemta818@gmail.com
- 3) 수업시간에 여러번 알려 주었거나 강의록에 명시되어 있는 부분은 답변하지 않음.