

Hoc1 프로그래밍 보고서



학 과	컴퓨터공학과
학 번	201211704
이 름	김기홍
제 출 일	2015.10.03

목 차

1. hoc1 을 프로그래밍 하기 위한 전반 작업

- LINUX MINT 설치	-----	3
- LINUX MINT 설정	-----	4
- lex	-----	5
- yacc	-----	9
- make	-----	9
- gcc	-----	9

2. hoc1 프로그램 수정 작업

(Default)	-----	10
-----------	-------	----

% (modulus 연산자 추가)

+ (UNARYPLUS 추가)

(Option)	-----	10
----------	-------	----

- (UNARYMINUS 추가)

_ (number of divisors 연산자 추가)

Syntax Error 시 프로그램이 종료되는 버그 수정

1. hoc1 을 프로그래밍 하기 위한 전반 작업

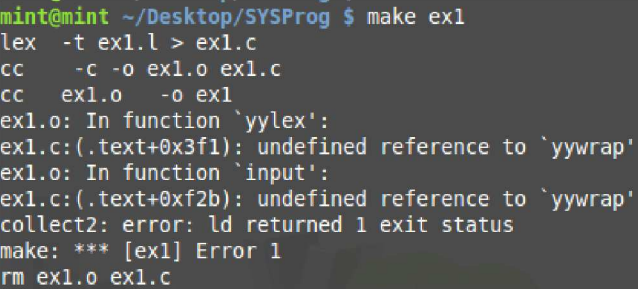
- LINUX MINT 설치

	Linux Mint iso 파일 수집
	Linux Mint iso 파일 마운트
	Start Linux Mint 를 선택하면 Linux Mint 를 설치한다.

- LINUX MINT 설정

	<p>전체 패키지를 검색하여 update 대상을 자동으로 update 한다</p>
	<p>패키지들에 대한 upgrade 를 한다</p>
	<p>Vim 패키지를 설치한다</p>
	<p>vim 의 설정을 변경한다</p>
	<p>문법 색 처리 줄번호 표시 자동 들여쓰기 등등 많은 설정이 있으므로 취향에 따라 설정한다</p>
	<p>절대경로없이 응용프로그램을 실행시킨다</p>

- lex

 <pre> 1 %% 2 . \n ECHO; 3 %% 4 5 main() { 6 yylex(); 7 } </pre>	<p>Ex1.l 의 코드이다</p>
 <pre> mint@mint ~/Desktop/SYSProg \$ make ex1 lex -t ex1.l > ex1.c cc -c -o ex1.o ex1.c cc ex1.o -o ex1 ex1.o: In function `yylex': ex1.c:(.text+0x3f1): undefined reference to `yywrap' ex1.o: In function `input': ex1.c:(.text+0xf2b): undefined reference to `yywrap' collect2: error: ld returned 1 exit status make: *** [ex1] Error 1 rm ex1.o ex1.c </pre>	<p>Make ex1 을 했을 경우 에러가 발생했다</p>
 <pre> 1 %% 2 . \n ECHO; 3 %% 4 5 int yywrap(void) { 6 return 0; 7 } 8 9 main() { 10 yylex(); 11 } </pre>	<p>Error message 에 따라 yywrap 함수를 작성해주었다</p>
 <pre> mint@mint ~/Desktop/SYSProg \$ make ex1 lex -t ex1.l > ex1.c cc -c -o ex1.o ex1.c cc ex1.o -o ex1 rm ex1.o ex1.c </pre>	<p>Make ex1 을 했을 경우 에러가 발생하지 않는 것을 볼 수 있다.</p> <p>※ 다른방법으로는 lex ex1.l → lex.yy.c gcc lex.yy.c -o ex1 -ll → ex1</p>

```
Thanks for flying Vim
mint@mint ~/Desktop/SYSProg $ ./ex1
lex
lex
ex1
ex1
lex ex1.1
lex ex1.1
```

Ex1 실행화면이다

Ex1 은 input 을 그대로
출력하는 프로그램이다

```
ex2.l (~/Desktop/SYSProg) - VIM
1 digit [0-9]
2 letter [a-zA-Z]
3 white [\t\n]
4
5 %%
6 .      ECHO;
7 {white} ECHO;
8 {letter}* printf("%s is string\n", yytext);
9 {digit}*  printf("%s is number\n", yytext);
10 %%
11
12 int yywrap(void) {
13     return 1;
14 }
15
16 int main(void) {
17     yylex();
18     return 0;
19 }
```

Ex2.1 의 코드이다

```
Thanks for flying Vim
mint@mint ~/Desktop/SYSProg $ ./ex2
Hi
Hi is string

1234
1234 is number

Hello1234
Hello is string
1234 is number
```

Ex2 실행화면이다

Ex2 는 문자로만 이뤄지면 string 을
출력하고 숫자로만 이뤄지면
number 을 출력하는 프로그램이다

```

1  %
2  [\t]+
3  rain |
4  rose | {printf("%s!! That is my favorite.\n", yytext);}
5  love |
6  story | {printf("%s!! I like a love story. ^_~\n", yytext);}
7  "potato chip" |
8  potato |
9  chip | {printf("%s...!! now and then, I have a potato chip.\n", yytext);}
10
11 [A-Za-z]+ {printf("%s..., what is that?\n", yytext);}
12
13 .|\n {ECHO;}
14
15 %
16
17 int yywrap(void) {
18     return 0;
19 }
20
21 main() {
22     yylex();
23 }
"simple.l" 23L, 381C 1,1 All

```

Simple.l 의 코드이다

```

mint@mint ~/Desktop/SYSProg $ ./simple
potato chip
potato chip...!! now and then, I have a potato chip.

love7
love!! I like a love story. ^_~
7
distress
distress..., what is that?

lovestory
lovestory..., what is that?

love story
love!! I like a love story. ^_~
story!! I like a love story. ^_~

```

Simple 의 실행화면이다

Simple 은 lex 를 활용하여
문자열에 따라 출력을 다르게
하는 프로그램이다

```
cat_number.l (~/.Desktop/SYSProg) - VIM
1 1 {
2   enum {
3       NONE = 0,
4       ONE,
5       TWO,
6       THREE,
7       FOUR,
8       FIVE,
9       SIX,
10      SEVEN
11  };
12
13  int number;
14  void action();
15  %}
16 %}
17 oneone {number=SEVEN; action();}
18 one {number = ONE; action();}
19 ONE {number = ONE; action();}
20 two {number = TWO; action();}
21 TWO {number = TWO; action();}
22 three {number = THREE; action();}
23 THREE {number = THREE; action();}
```

cat_number 의 코드이다

```
cat_number.l (~/.Desktop/SYSProg) - VIM
24 four {number = FOUR; action();}
25 FOUR {number = FOUR; action();}
26 five {number = FIVE; action();}
27 FIVE {number = FIVE; action();}
28 six {number = SIX; action();}
29 SIX {number = SIX; action();}
30 [a-zA-Z]+ {number = NONE; action();}
31 %}
32
33 void action()
34 {
35     if(number != NONE)
36         printf("%s is %d\n", yytext, number);
37     else
38         printf("%s?? I don't learn it.\n", yytext);
39 }
40 int yywrap(void) {
41     return 0;
42 }
43 main()
44 {
45     yylex();
46 }
```

```
Thanks for flying Vim
five
five is 5

four7
four is 4
7
SIX
SIX is 6

ten
ten?? I don't learn it.

oneone
oneone is 7

twotwo
twotwo?? I don't learn it.

two two
two is 2
two is 2
```

cat_number 의 실행화면이다

cat_number 은 input 에 해당하는
숫자를 출력하고
미리 설정해놓은 문자열이
아니라면
I don't learn it 을 출력하는
프로그램이다

- yacc

yacc 는 통상 lex 와 함께 쓰이며 lex 의 구조와 아주 유사하다.

lex 와 yacc 를 한번 알아보자면

예를 들어 if(1) printf(“hello\n”); 라는 문장이 있다고 하자.

여기서 어휘 분석을 하면

```
if ( 1 ) printf ( “hello\n” ) ;
```

이것을 구문 해석을 한다.

어휘분석 프로그램 lex 이고 구문해석 프로그램(파서)가 yacc 이다.

- make, gcc

컴파일을 하는 명령어로서 make 는 makefile 을 찾아보고 없다면 파일을 만들 수 있는 파일을 찾아서 만들어냄 예를들면

make main 이라고 하면 main.c 를 찾아 gcc -o main main.c 를 수행한다.

2. hoc1 프로그램 수정 작업

(Default)

<pre> '+' expr %prec UNARYPLUS { \$\$ = \$2; } '-' expr %prec UNARYMINUS { \$\$ = -\$2; }</pre>	<p>UNARYPLUS 와 UNARYMINUS 연산을 문법에 추가한다</p>
<pre> expr '%' expr { if(\$1 - (int)\$1 == 0 && \$3 - (int)\$3 == 0) \$\$ = (int)\$1 % (int)\$3; else yyerror("Reinsert as an integer"); }</pre>	<p>Modulus 연산자를 추가한다</p> <p>\$1, \$3 이 정수가 아니라면 yyerror()함수를 호출한다</p>

(Option)

<pre>while(getchar() != '\n'); yyparse();</pre>	<p>버퍼의 내용을 모두 지우고 yyparse()를 호출함으로서 Error Message 이후에도 프로그램이 종료되지 않는다</p>
<pre> '_' expr { int n = \$2; int count=0; int num=1; int i; for(i=2; i<=\$2; i++) { while(n%i == 0) { count++; n /= i; } num *= ++count; count = 0; if(n == 1) break; } \$\$ = num; }</pre>	<p>'_' 은 약수의 개수를 계산하는 연산이다 $p^m * q^n$ (p,q 는 소수 m,n 은 정수) 일 때 $(m+1)*(n+1)$을 계산하는 방법으로 약수의 개수를 구한다</p>

```

| '_' expr {
    int number = $2; int count=2; int i;

    for(i=2; i*i<=number; i++)
    {
        if(i*i == number)
            count++;
        else if(number % i == 0)
            count +=2;
    }

    $$ = count;
}

```

또는 count 변수에 2를 저장해놓고(1과 number 자신)
number의 제곱근까지 반복하면서 나눠지면 count에 2를 더해준다(i와 number/i)
number이 제곱수라면 2대신 1을 더해준다

전체 코드

```

1  /*
2  #include <stdio.h>
3  #include <ctype.h>
4  #define YYSTYPE double
5  */
6
7  %token NUMBER
8  %left '+' '-'
9  %left '*' '/' '%' /* for operating mod */
10 %left UNARYPLUS UNARYMINUS '_' /* for expressing unary plus or minus and '\\' '_' is an operator which can get number of divisors\\ */
11
12
13 list:
14     list '\n'
15     list expr '\n' { printf("\t%.8g\n", $2); }
16
17
18 expr:
19     NUMBER { $$ = $1; }
20     expr '+' expr { $$ = $1 + $3; }
21     expr '-' expr { $$ = $1 - $3; }
22     expr '*' expr { $$ = $1 * $3; }
23     expr '/' expr { $$ = $1 / $3; }
24     expr '%' expr {
25         if($1 - (int)$1 == 0 && $3 - (int)$3 == 0) /* if the inputs are not integer, display an error message */
26             $$ = (int)$1 % (int)$3;
27         else
28             yyerror("Reinsert as an integer");
29     }
30     | '_' expr {
31         int number = $2; int count=2; int i;
32
33         for(i=2; i*i<=number; i++)
34         {
35             if(i*i == number)
36                 count++;
37             else if(number % i == 0)
38                 count +=2;
39         }
40
41         $$ = count;
42     }
43     | '(' expr ')' { $$ = $2; }
44     | '+' expr %prec UNARYPLUS { $$ = $2; }
45     | '-' expr %prec UNARYMINUS { $$ = -$2; }
46
47
48
49 char *programe;
50 int lineno=1;
51
52 main(argc, argv)
53     char *argv[];
54 {
55     programe = argv[0];
56     yyparse();
57 }
58
59 yylex()

```

```

60     yyparse();
61 }
62
63 yylex()
64 {
65     int c;
66
67     while((c = getchar()) == ' ' || c == '\t')
68         ;
69     if(c==EOF)
70         return 0;
71     if(c=='.' || isdigit(c))
72     {
73         ungetc(c, stdin);
74         scanf("%lf", &yylval);
75         return NUMBER;
76     }
77
78     if(c == '\n')
79         lineno++;
80
81     return c;
82 }
83
84 yyerror(s)
85     char *s;
86 {
87     warning(s, (char *)0);
88 }
89
90 warning(s, t)
91     char *s, *t;
92 {
93     fprintf(stderr, "%s: %s", progname, s);
94     if(t)
95         fprintf(stderr, " %s", t);
96     fprintf(stderr, " near line %d\n", lineno);
97     puts("");
98     while(getchar() != '\n');    /* for continuing the program without the remains of buffer after occuring an syntax error */
99     yyparse();
100 }

```