CS 1723, Parser and Reverse Polish Generator, Wed Nov 25 1998

```
runner% cat arith0.c
/* arith0.c: simple parser -- no output
 * grammar:
 *   P ---> E '#'
 *   E ---> T {('+'|'-') T}
 *   T ---> S {('*'|'/') S}
 *   S ---> F '^' S
 *   F ---> char | '(' E ')'
 */
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
char next;
void E(void);
void T(void);
void S(void);
void F(void);
void error(void);
void scan(void);

void main(void)
{
   scan();
   E();
   if (next != '#') error();
   else printf("Successful parse\n");
}

void E(void)
{
   T();
   while (next == '+' || next == '-') {
      scan();
      T();
   }
}

void T(void)
{
   S();
   while (next == '*' || next == '/') {
      scan();
      S();
   }
}

void S(void)
{
   F();
   if (next == '^') {
      scan();
      S();
   }
}
void F(void)
{
   if (isalpha(next) || isdigit(next)) {
      scan();
   }
   else if (next == '(') {
      scan();
      E();
      if (next == ')') scan();
      else error();
   }
   else {
      error();
   }
}
void scan(void)
{
   while (isspace(next = getchar()))
      ;
}
void error(void)
{
   printf("\n*** ERROR ***\n");
   exit(1);
}

runner% arith0
(a + b*c)^(d/e - f)*g#
Successful parse
runner% arith0
a + b c #
*** ERROR ***
runner% arith0
( (a + b)*c #
*** ERROR ***
runner% arith0
a + b ) #
*** ERROR ***
```

CS 3323, Parser and Reverse Polish Generator, Fri Feb 12 1999

```
runner% cat rpn.c
/* rpn: output Reverse Polish Form */
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
char next;
void E(void);
void T(void);
void S(void);
void F(void);
void gen(char);
void error(void);
void scan(void);

void main(void)
{
   scan();
   E();
   if (next != '#') error();
   else printf("\n");
}
void E(void)
{
   char save;
   T();
   while (next == '+' || next == '-') {
      save = next;
      scan();
      T();
      gen(save);
   }
}
void T(void)
{
   char save;
   S();
   while (next == '*' || next == '/') {
      save = next;
      scan();
      S();
      gen(save);
   }
}
void S(void)
{
   F();
   if (next == '^') {
      scan();
      S();
      gen('^');
   }
}
void F(void)
{
   if (isalpha(next) || isdigit(next)) {
      gen(next);
      scan();
   }
   else if (next == '(') {
      scan();
      E();
      if (next == ')') scan();
      else error();
   }
   else {
      error();
   }
}
void scan(void)
{
   while (isspace(next = getchar()))
      ;
}
void error(void)
{
   printf("\n*** ERROR ***\n");
   exit(1);
}
void gen(char ch)
{
   putchar(ch);
}
runner% rpn
2 + 3 + 4 + 5 #
23+4+5+
runner% rpn
a^b^c^d#
abcd^^^
runner% rpn
(a + b*c)^(d/e - f)*g#
abc*+de/f-^g*
runner% rpn
a + b c #
ab+
*** ERROR ***
runner% rpn
( (a + b)*c #
ab+c*
*** ERROR ***
runner% rpn
a + b ) #
ab+
*** ERROR ***
```