%{

  #include<stdio.h>

  #include<string.h>

  #include<ctype.h>

  char lexema[255];

  void yyerror(char \*);

%}

// Especificamos los tokens

%token NUM MAS MENOS POR ENTRE

// Especificamos la gramatica

%%

exp: exp MAS term

   | exp MENOS term

   | term;

term: term POR NUM

    | term ENTRE NUM

    | NUM;

%%

void yyerror(char \*msg) {

  printf("error: %s", msg);

}

// Especificamos las reglas de los tokens

int yylex() {

  char c;

  while(1) {

    c = getchar();

    if(c == '\n') continue;

    if(isspace(c)) continue;

    if(c == '+') return MAS;

    if(c == '-') return MENOS;

    if(c == '\*') return POR;

    if(c == '/') return ENTRE;

    if(isdigit(c)) {

      int i = 0;

      do {

        lexema[i++] = c;

        c = getchar();

      } while(isdigit(c));

      ungetc(c, stdin);

      lexema[i] == 0;

      return NUM;

    }

    return c;

  }

}

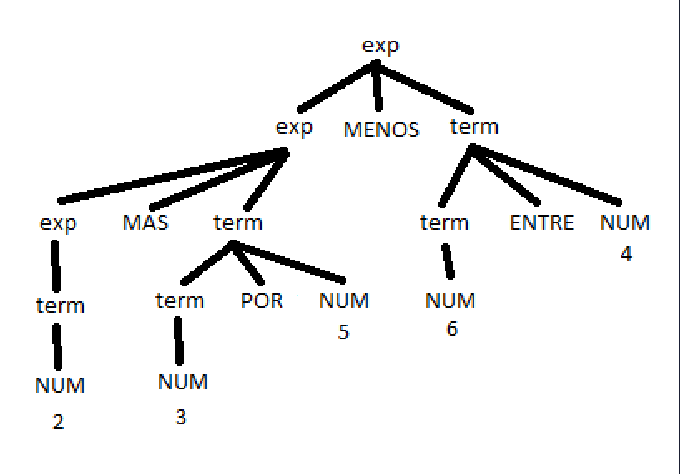
int main() {

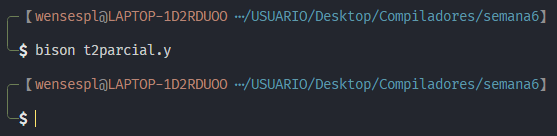
  if(!yyparse()) printf("\ncadena valida\n");

  else printf("cadena invalida\n");

  return 0;

}





No sale errores de ambigüedad y del grafico del árbol notamos que se respeta la precedencia

