Homework 7 (10 points)

Symbol table

1. Extend the Yacc parser with semantic actions for creating the symbol table that stores variables and constants using the `entry` data structure below.
2. Update the corresponding references in the syntax tree by using the new `node` data structure below.

```
typedef enum { _CONST=0, _SCALAR, _ARRAY } entry_type;
typedef enum { _BOOL=0, _INT, _REAL } data_type;


typedef struct _entry {
      entry_type etype ;
      data_type dtype ;
      union {
            int int_val ;
            float real_val ;
            char *scalar ;
            void *array[2] ;  /* identifier, index */
      } symbol;
      struct _entry *next ;   /* collision list */
    } entry ;


typedef enum { PROGRAM, ASSIGN, IF, WHILE, STATEMENT, CONST, VAR, TYPE,
     EXPR, INT_CONST, REAL_CONST, BOOL_CONST, STRING_CONST, IDENTIFIER, OP
     } node_type;


typedef enum { PLUS, MINUS, MUL, DIV, MOD, LT, LE, GT, GE, EQ, NE, AND, OR
     } operator;


typedef struct _node {
      node_type type ;
      union {
            operator op ;
            entry *symbol;
            /* list of BNF right-hand side symbols of nonterminal type */
            struct _node *body ;
      } ;
      struct _node *next ;          /* decl-list, io-list, stmt-list */
} node ;
```