```
        /* ---------------------------------------------------------
                              The grammar symbols
         --------------------------------------------------------- */

%token  WORD
%token  ASSIGNMENT_WORD
%token  NAME
%token  NEWLINE
%token  IO_NUMBER

/* The following are the operators mentioned above. */

%token  AND_IF    OR_IF    DSEMI
/*      '&&'      '||'     ';;'    */


%token  DLESS  DGREAT  LESSAND  GREATAND  LESSGREAT  DLESSDASH
/*      '<<'   '>>'    '<&'     '>&'      '<>'       '<<-'   */


%token  CLOBBER
/*      '>|'   */


/* The following are the reserved words. */


%token  If    Then    Else    Elif    Fi    Do    Done
/*      'if'  'then'  'else'  'elif'  'fi'  'do'  'done'   */


%token  Case    Esac    While    Until    For
/*      'case'  'esac'  'while'  'until'  'for'   */


/* These are reserved words, not operator tokens, and are
   recognized when reserved words are recognized. */


%token  Lbrace    Rbrace    Bang
/*      '{'       '}'       '!'   */


%token  In
/*      'in'   */
```

```
        /* ------------------------------------------------------
                                The Grammar
          ------------------------------------------------------ */


%start  complete_command
%%
complete_command : list separator
                 | list
                 ;
list             : list separator_op and_or
                 |                   and_or
                 ;
and_or           :                         pipeline
                 | and_or AND_IF linebreak pipeline
                 | and_or OR_IF  linebreak pipeline
                 ;
pipeline         :      pipe_sequence
                 | Bang pipe_sequence
                 ;
pipe_sequence    :                             command
                 | pipe_sequence '|' linebreak command
                 ;
command          : simple_command
                 | compound_command
                 | compound_command redirect_list
                 | function_definition
                 ;
compound_command : brace_group
                 | subshell
                 | for_clause
                 | case_clause
                 | if_clause
                 | while_clause
                 | until_clause
                 ;
subshell         : '(' compound_list ')'
                 ;
compound_list    :              term
                 | newline_list term
                 |              term separator
                 | newline_list term separator
                 ;
```

```
term            : term separator and_or
                |                and_or
                ;
for_clause      : For name linebreak                            do_group
                | For name linebreak in          sequential_sep do_group
                | For name linebreak in wordlist sequential_sep do_group
                ;
name            : NAME                     /* Apply rule 5 */
                ;
in              : In                       /* Apply rule 6 */
                ;
wordlist        : wordlist WORD
                |          WORD
                ;
case_clause     : Case WORD linebreak in linebreak case_list    Esac
                | Case WORD linebreak in linebreak case_list_ns Esac
                | Case WORD linebreak in linebreak              Esac
                ;
case_list_ns    : case_list case_item_ns
                |           case_item_ns
                ;
case_list       : case_list case_item
                |           case_item
                ;
case_item_ns    :     pattern ')'               linebreak
                |     pattern ')' compound_list linebreak
                | '(' pattern ')'               linebreak
                | '(' pattern ')' compound_list linebreak
                ;
case_item       :     pattern ')' linebreak     DSEMI linebreak
                |     pattern ')' compound_list DSEMI linebreak
                | '(' pattern ')' linebreak     DSEMI linebreak
                | '(' pattern ')' compound_list DSEMI linebreak
                ;
pattern         :             WORD         /* Apply rule 4 */
                | pattern '|' WORD         /* Do not apply rule 4 */
                ;
if_clause       : If compound_list Then compound_list else_part Fi
                | If compound_list Then compound_list           Fi
                ;
else_part       : Elif compound_list Then else_part
                | Else compound_list
                ;
while_clause    : While compound_list do_group
                ;
```

```
until_clause     : Until compound_list do_group
                 ;
function_definition : fname '(' ')' linebreak function_body
                 ;
function_body    : compound_command                /* Apply rule 9 */
                 | compound_command redirect_list  /* Apply rule 9 */
                 ;
fname            : NAME                            /* Apply rule 8 */
                 ;
brace_group      : Lbrace compound_list Rbrace
                 ;
do_group         : Do compound_list Done           /* Apply rule 6 */
                 ;
simple_command   : cmd_prefix cmd_word cmd_suffix
                 | cmd_prefix cmd_word
                 | cmd_prefix
                 | cmd_name cmd_suffix
                 | cmd_name
                 ;
cmd_name         : WORD                   /* Apply rule 7a */
                 ;
cmd_word         : WORD                   /* Apply rule 7b */
                 ;
cmd_prefix       :            io_redirect
                 | cmd_prefix io_redirect
                 |            ASSIGNMENT_WORD
                 | cmd_prefix ASSIGNMENT_WORD
                 ;
cmd_suffix       :            io_redirect
                 | cmd_suffix io_redirect
                 |            WORD
                 | cmd_suffix WORD
                 ;
redirect_list    :               io_redirect
                 | redirect_list io_redirect
                 ;
io_redirect      :           io_file
                 | IO_NUMBER io_file
                 |           io_here
                 | IO_NUMBER io_here
                 ;
io_file          : '<'       filename
                 | LESSAND   filename
                 | '>'       filename
                 | GREATAND  filename
```

```
                | DGREAT     filename
                | LESSGREAT  filename
                | CLOBBER    filename
                ;
filename        : WORD                      /* Apply rule 2 */
                ;
io_here         : DLESS     here_end
                | DLESSDASH here_end
                ;
here_end        : WORD                      /* Apply rule 3 */
                ;
newline_list    :              NEWLINE
                | newline_list NEWLINE
                ;
linebreak       : newline_list
                | /* empty */
                ;
separator_op    : '&'
                | ';'
                ;
separator       : separator_op linebreak
                | newline_list
                ;
sequential_sep  : ';' linebreak
                | newline_list
                ;
```