```
procedure TERM
      begin
         FOUND := FALSE
         if FACTOR returns success then
            begin
               FOUND := TRUE
               while ({TOKEN = 18 {*}) or (TOKEN = 19 {DIV})
                  and (FOUND = TRUE) do
                     begin
                        advance to next token
                        if FACTOR returns failure then
                           FOUND := FALSE
                     end {while}
            end {if FACTOR}
         if FOUND = TRUE then
            return success
         else
            return failure
      end {TERM}


procedure FACTOR
      begin
         FOUND := FALSE
         if (TOKEN = 22 {id}) or (TOKEN = 23 {int}) then
            begin
               FOUND := TRUE
               advance to next token
            end {if id or int}
         else
            if TOKEN = 20 { ( } then
               begin
                  advance to next token
                  if EXP returns success then
                     if TOKEN = 21 { ) } then
                        begin
                           FOUND := TRUE
                           advance to next token
                        end {if )}
               end {if ( }
         if FOUND = TRUE then
            return success
         else
            return failure
      end {FACTOR}
```

(a)

Figure 5.17 *(cont'd)*