```
procedure LRParse( T, D )
where T is an LR Action Table (eg: an LR(0) or SLR(1) table), D
is a deque of tokens from the sentence with operations pop
and push operating at the front of the deque. D has one
or more $ markers at the back of the deque.

  S ← empty stack
  push state (0, emptyTree) onto S
  while ( |D| > 0 ) do (
    t ← D.front()
    if ( T[S.top.state][t] does not exist ) then (
      SYNTAX ERROR, terminate parse
    )
    action ← T[S.top.state][t]
    if ( action is SHIFT AND GO TO newState ) then (
      t ← D.pop()
      S.push((newState, t))
    ) else if ( action is REDUCE WITH p ) then (
      reduce the topmost elements of S with production
        rule p, push the non-terminal LHS result onto D
    ) else if ( action is REDUCE WITH p AND ACCEPT ) then (
      reduce the remaining elements of S with production rule p
      terminate parse accepting the token sequence
        from D as a valid sentence of the grammar
    )
  )
  SYNTAX ERROR, terminate parse
```