```
<stmt> ::= {any alternative}

                {no code-generation action}


<write> ::= WRITE (<id-list>)

                generate [ +JSUB    XWRITE]
                record external reference to XWRITE
                generate [  WORD    LISTCOUNT]
                for each item on list do
                    begin
                        remove S(ITEM) from list
                        generate [  WORD    S(ITEM)]
                    end
                LISTCOUNT := 0


<for> ::= FOR <index-exp> DO <body>

                pop JUMPADDR from stack {address of jump out of loop}
                pop S(INDEX) from stack {index variable}
                pop LOOPADDR from stack {beginning address of loop}
                generate [  LDA  S(INDEX)]
                generate [  ADD  #1]
                generate [  J    LOOPADDR]
                insert [    JGT  LOCCTR]   at location JUMPADDR


<index-exp> ::= id := <exp>1 TO <exp>2

                GETA (<exp>1)
                push LOCCTR onto stack {beginning address of loop}
                push S(id) onto stack {index variable}
                generate [   STA   S(id)]
                generate [   COMP S(<exp>2)]
                push LOCCTR onto stack {address of jump out of loop}
                add 3 to LOCCTR  {leave room for jump instruction}
                REGA := null


<body> ::= {either alternative}

                {no code-generation action}
```

Figure 5.20 (*cont'd*)