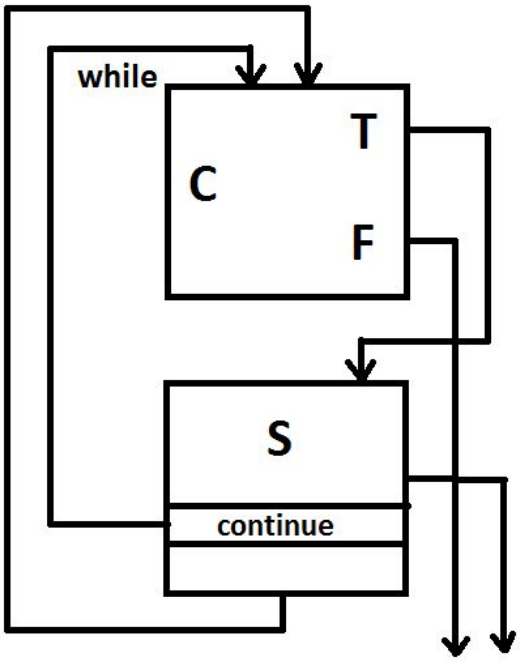
Syntax Directed Translation

Continue and Break



S→break;
{
S.nextlist = MakeList (NextQuad);
Gen (“goto__”);
}

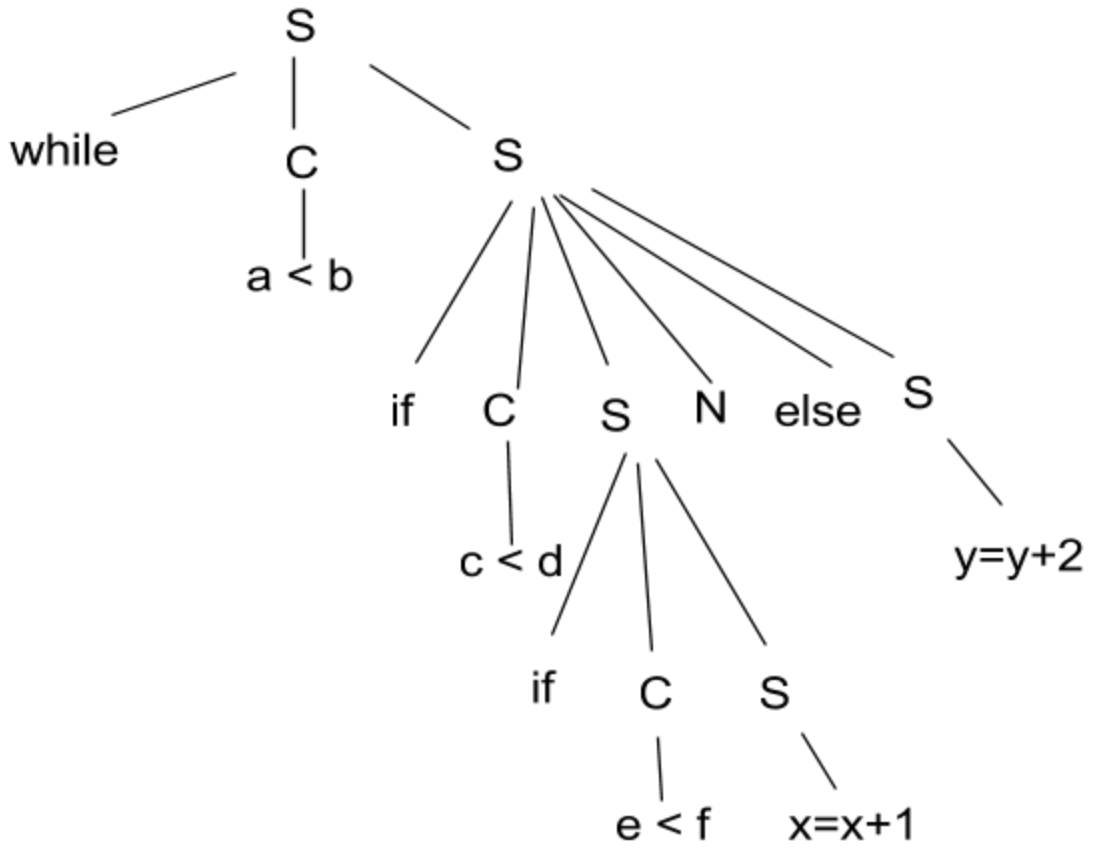
S→continue;
{
S.contList = MakeList (NextQuad);
Gen (“goto __”);
}

S→ while $M^{(1)}$ C $M^{(2)}$ $S^{(1)}$
{
Backpatch (C.TrueList, $M^{(2)}$.Quad);
S.nextList = Merge (c.FalseList , $S^{(1)}$.nextList);
Gen (“goto $M^{(1)}$.Quad ”);
Backpatch ($S^{(1)}$.contList , $M^{(1)}$.Quad);
}

Solved Example

```
while ( a<b ) {
        If ( c<d ) {
                If ( e<f ) {
                        x=x+1;
                }
                else{
                        y=y+2;
                }
        }
}
```

```
100 : if ( a<b ) goto  102
101 : goto __
102 : if ( c<d ) goto 104
103 : goto 108
104 : if ( e<f ) goto 106
105 : goto 100
106 : x=x+1
107 : goto 100          //For marker N
108 : y=y+2
109 : goto 100
```



Function Calls

Grammar for a simple function call statement is as follows:

S → call id(paramlist)
paramlist → paramlist , E
paramlist → E

S → call id(paramlist) {
for each p in paramlist do *//Consult symbol table to generate activation records*
{
GEN(param, p.place); //check if type of p matches with type expected by id
}
call id;
}

paramlist → paramlist, E {
put E.place in paramlist queue
}

paramlist → E {
put E.place in paramlist queue
}

Example

```
f()  {
       int x;
       x=5;        [base of activation record + 4] = 5
     }
```