```
Switch E {
        Case V1: others;
}

Clause -> case const : N1 M S N2
{
        Clause.tempList = newList(nextQuad)
        Clause.falseList = newList(nextQuad)
        emit(if __ != const.addr goto __ else goto M.quad)
        Clause.nextList = merge(N.nextList, S.nextList);
}

Clauses -> clause; M clauses1
{
        backpatch(clause.falseList, M.quad)
        Clauses.tempList = merge(clause.tempList, clauses1.tempList)
        Clauses.falseList = clauses1.falseList
        Clauses.nextList = merge(clause.nextList, clauses1.nextList)
        if(clause.defaultAddr)
                Clauses.defaultAddr  = clause.defaultAddr
        Else
                Clauses.defaultAddr  = clauses1.defaultAddr

}
//Assumption: semantic analysis makes sure at most one default case in the switch
S -> switch E {clauses}
{
        backpatch(clauses.tempList, E.addr)
        if(clauses.defaultAddr)
                backpatch(clauses.falseList, clauses.defaultAddr)
                S.nextList = clauses.nextList
        Else
                S.nextList = merge(clauses.falseList, clauses.nextList)
}

S -> break
{
        S.nextList = newList(nextQuad)
        emit(goto __)
}

S -> S1; M S2
{
        backpatch(S1.nextList, M.quad)
```

```
        S.nextList = S2.nextList
}

Clause -> default : M S N
{
        Clause.defaultAddr = M.quad
        Clause.nextList = merge(S.nextList, N.nextList)
}
Clauses -> clause
{
        Clauses.nextList = clause.nextList
        Clauses.falseList = clause.falseList
        Clauses.defaultAddr = clause.defaultAddr
}

S -> others {}
```