```
(1)     procedure VISIT(v);
(2)     begin
(3)         root[v] := v; InComponent[v] := False;
(4)         PUSH(v, stack);
(5)         for each node w such that (v, w) ∈ E do begin
(6)             if w is not already visited then VISIT(w);
(7)             if not InComponent[w] then root[v] := MIN(root[v], root[w])
(8)         end;
(9)         if root[v] = v then
(10)            repeat
(11)                w := POP(stack);
(12)                InComponent[w] := True;
(13)            until w = v
(14)    end;
(15)    begin/* Main program */
(16)        stack := ∅;
(17)        for each node v ∈ V do
(18)            if v is not already visited then VISIT(v)
(19)    end.
```

Figure 1: Tarjan's algorithm detects the strongly connected components of graph $G = (V, E)$.