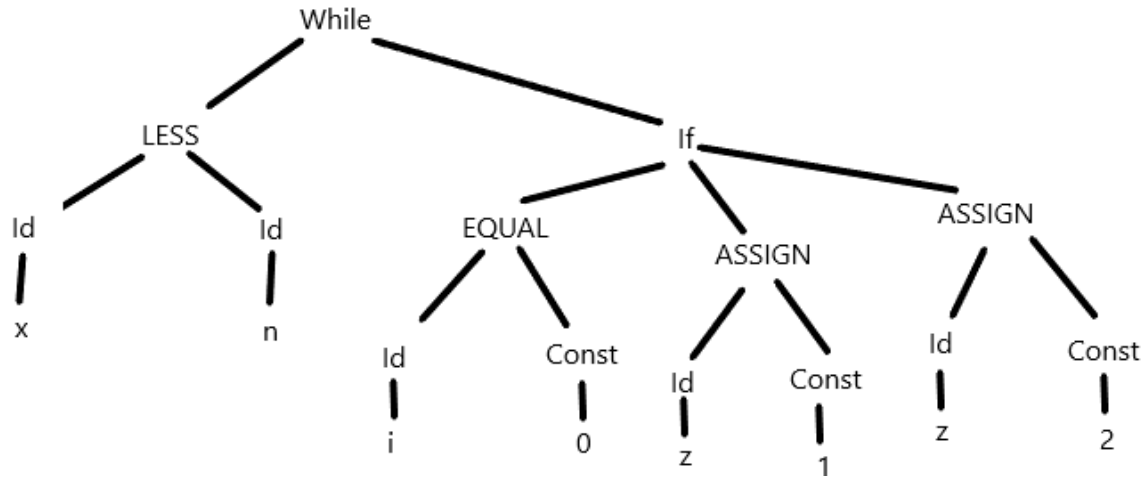
Q26



Q34

VO :

```
var VT symb := next_symb(w);
push(S);
repeat
        top := top_of_stack();
        if top = $ or top ∈ VT then
                if top = symb then pop();
                        symb := next_symb(w);
                else syntax_error();
        else if top = X and δ[top, symb] = X → Y1 · · · Yn
                then pop(X);
                        push(Yn); · · · push(Y1)
                else syntax_error() ;
until not empty_stack();
```

VF :

```
Var VT symb = next_symb(w);
Empiler(S);
repeter
        top := sommet_pile();
        si top = $ ou top ∈ VT alors
                si top = symb alors depiler() ;
                                        symb := next_symb(w) ;
                sinon erreur_syntaxe() ;
        sinon si top = X et δ[top, symb] = X -> Y1 … Yn
                alors depiler(X) ;
                        empiler(Yn) ; …empiler(Y1)
                sinon erreur_syntaxe() ;
tant que pas pile_vide() ;
```

Q49

S -> E	
E -> E1 + T	E.noeud = new nœud(‘+’, E1.noeud, T.noeud)
E -> T	E.noeud = T.noeud
T -> T1 x F	T.noeud = new nœud (‘*’, T1.noeud, F.noeud)
T -> F	T.noeud = F.noeud
F -> (E)	F.noeud = E.noeud
F -> const	F.noeud = new nœud(const)

Q51

T0 = -b
T1 = int(rac)
T0 = T0 + T1
T1 = 2*a
T0 = T0 / T1
x1 = T0

Q53 I -> Tant que E faire I1

Début = newLabel()
E.vrai = newLabel()
E.faux = I.suiv
I1.suiv = début
I.code = (début" :") + E.code
I.code += (E.vrai" :") + I1.code
I.code =+ (" aller à" début)

Q55 Optimiseur à lucarne :

- Examen d'une petite séquence (lucarne) d'instruction à la fois
- Remplacer si possible ces instructions pas une séquence plus courte/efficace
- Déplacer la lucarne d'une instruction vers le bas sur tout le code
- Recommencer tant qu'il y a de nouvelles optimisations.