function Unify(x, y, θ) returns a substitution to make x and y identical

inputs: x, a variable, constant, list, or compound expression
        y, a variable, constant, list, or compound expression
        θ, the substitution built up so far (optional, defaults to empty)

if θ = failure then return failure
else if x = y then return θ
else if VARIABLE?(x) then return Unify-Var(x, y, θ)
else if VARIABLE?(y) then return Unify-Var(y, x, θ)
else if COMPOUND?(x) and COMPOUND?(y) then
    return Unify(xARGS, yARGS, Unify(xOp, yOp, θ))
else if LIST?(x) and LIST?(y) then
    return Unify(xREST, yREST, Unify(xFIRST, yFIRST, θ))
else return failure

function Unify-Var(var, x, θ) returns a substitution

if {var / val} ∈ θ then return Unify(val, x, θ)
else if {x / val} ∈ θ then return Unify(var, val, θ)
else if Occur-Check?(var, x) then return failure
else return add {var / x} to θ

Figure 9.1 The unification algorithm. The algorithm works by comparing the structures of the inputs, element by element. The substitution θ that is the argument to Unify is built up along the way and is used to make sure that later comparisons are consistent with bindings that were established earlier. In a compound expression such as F(A, B), the Op field picks out the function symbol F and the Args field picks out the argument list (A, B).