function Unify\((x, y, \theta)\) 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
\(\theta\), the substitution built up so far (optional, defaults to empty)

if \(\theta = \text{failure}\) then return \(\text{failure}\)
else if \(x = y\) then return \(\theta\)
else if VARIABLE\((x)\) then return Unify-Var\((x, y, \theta)\)
else if VARIABLE\((y)\) then return Unify-Var\((y, x, \theta)\)
else if COMPOUND\((x)\) and COMPOUND\((y)\) then
    return Unify\((\text{op} x, \text{op} y, \theta)\)
else if LIST\((x)\) and LIST\((y)\) then
    return Unify\((\text{rest} x, \text{rest} y, \theta)\)
else return \(\text{failure}\)

function Unify-Var\((\text{var}, x, \theta)\) returns a substitution

if \(\{\text{var}/\text{val}\} \in \theta\) then return Unify\((\text{val}, x, \theta)\)
else if \(\{x/\text{val}\} \in \theta\) then return Unify\((\text{var}, \text{val}, \theta)\)
else if OCCUR-CHECK\((\text{var}, x)\) then return \(\text{failure}\)
else return add \(\{\text{var}/x\}\) to \(\theta\)

Figure 9.1 The unification algorithm. The algorithm works by comparing the structures of the inputs, element by element. The substitution \(\theta\) 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 \((A, B)\).