```python
def unify(x, y, subst={}):
    if subst is None:
        return None
    elif x == y:
        return subst
    elif isinstance(x, str) and x.islower():
        return unify_variable(x, y, subst)
    elif isinstance(y, str) and y.islower():
        return unify_variable(y, x, subst)
    elif isinstance(x, list) and isinstance(y, list):
        if len(x) != len(y):
            return None
        for xi, yi in zip(x, y):
            subst = unify(xi, yi, subst)
        return subst
    else:
        return None

def unify_variable(var, x, subst):
    if var in subst:
        return unify(subst[var], x, subst)
    elif x in subst:
        return unify(var, subst[x], subst)
    elif occurs_check(var, x):
        return None
    else:
        subst[var] = x
        return subst

def occurs_check(var, x):
    if var == x:
        return True
    elif isinstance(x, list):
        return any(occurs_check(var, arg) for arg in x)
    return False

def test_unification():
    terms = [
        (['P', 'x', 'f(y)'], ['P', 'a', 'f(g(x))']),
        (['P', 'X', 'b'], ['P', 'a', 'b']),
        (['P', 'X', 'b'], ['P', 'f(X)', 'b']),
        (['P', 'a', 'b'], ['Q', 'a', 'b']),
        (['P', 'a'], ['P', 'a', 'b']),
    ]

    for t1, t2 in terms:
        subst = unify(t1, t2)
        print(f"Unify({t1}, {t2}) -> {'Unification Possible' if subst else 'Unification Not Possible'}")

test_unification()
```

```
Unify(['P', 'x', 'f(y)'], ['P', 'a', 'f(g(x))']) -> Unification Possible
Unify(['P', 'X', 'b'], ['P', 'a', 'b']) -> Unification Possible
Unify(['P', 'X', 'b'], ['P', 'f(X)', 'b']) -> Unification Not Possible
Unify(['P', 'a', 'b'], ['Q', 'a', 'b']) -> Unification Not Possible
Unify(['P', 'a'], ['P', 'a', 'b']) -> Unification Not Possible
```