```
def unify(expr1, expr2):
    def is_variable(x):
        return isinstance(x, str) and x.islower()
    def occurs_in(var, expr):
        if var == expr:
            return True
        elif isinstance(expr, list):
            return any(occurs_in(var, sub_expr) for sub_expr in expr)
        return False
    if is_variable(expr1):
        if expr1 == expr2:
            return None
        elif occurs_in(expr1, expr2):
            return "FAILURE"
        else:
            return [(expr2, expr1)]
    elif is_variable(expr2):
        if occurs_in(expr2, expr1):
            return "FAILURE"
        else:
            return [(expr1, expr2)]
    elif isinstance(expr1, str) and isinstance(expr2, str):
        if expr1 == expr2:
            return None
        else:
            return "FAILURE"
    if isinstance(expr1, list) and isinstance(expr2, list):
        if expr1[0] != expr2[0]:
            return "FAILURE"
        if len(expr1) != len(expr2):
            return "FAILURE"
        subst = []
        for arg1, arg2 in zip(expr1[1:], expr2[1:]):
            result = unify(arg1, arg2)
            if result == "FAILURE":
                return "FAILURE"
            elif result is not None:
                for sub in result:
                    expr1 = apply_substitution(expr1, sub)
                    expr2 = apply_substitution(expr2, sub)
                    subst.append(sub)
        return subst
    return "FAILURE"
def apply_substitution(expr, substitution):
    value, variable = substitution
    if isinstance(expr, str):
        return value if expr == variable else expr
    elif isinstance(expr, list):
        return [apply_substitution(sub_expr, substitution) for sub_expr in expr]
    return expr
# input the expression here
expr1 = ["P", "b", "f(g(c))"]
expr2 = ["P", "a", "b"]
result = unify(expr1, expr2)
print("Result:", result)
```

```
Result: [('a', 'b'), ('b', 'f(g(c))')]
```