```
# Lab 1: Math and String shtuff
# Whew, this was tedious.
# Andrea Smith
# CSCI 1913

def left(e):
    return e[0]

def op(e):
    return e[1]

def right(e):
    return e[2]

def isInside(v,e): # v is a variable, e is an equation
    if v == e:
        return True
    elif type(e) == str:
        return False
    else:
        return isInside(v,left(e)) or isInside(v,right(e))

def solving(v,q):
    if v == left(q):
        return q
    elif op(left(q)) == "+":
        return solvingAdd(v,q)
    elif op(left(q)) == "-":
        return solvingSubtract(v,q)
    elif op(left(q)) == "*":
        return solvingMultiply(v,q)
    elif op(left(q)) == "/":
        return solvingDivide(v,q)

def solve(v,q):

    if isInside(v, left(q)):
        return solving(v,q)

    elif isInside(v, right(q)):
        return solving(v, [q[2], q[1], q[0]])

    else:
        return None

def solvingAdd(v,q):
    a = left(left(q))       # gets farthest left var within the left
```

```python
    expression (i.e. if left is x + b, will give x)
    b = right(left(q))
    c = right(q)

    if isInside(v,a):
        newEquation = (a, "=", (c, "-", b))
        return solving(v,newEquation)
    else:
        newEquation = (b, "=", (c, "-", a))
        return solving(v,newEquation)

def solvingSubtract(v,q):
        a = left(left(q))       # gets farthest left var within the left expression (i.e. if left is x + b, will give x)
        b = right(left(q))
        c = right(q)

        if isInside(v,a):
            newEquation = (a, "=", (c, "+", b))
            return solving(v,newEquation)
        elif isInside(v,b):
            newEquation = (b, "=", (a, "-", c))
            return solving(v,newEquation)
        else:
            return None

def solvingMultiply(v,q):
        a = left(left(q))       # gets farthest left var within the left expression (i.e. if left is x + b, will give x)
        b = right(left(q))
        c = right(q)

        if isInside(v,a):
            newEquation = (a, "=", (c, "/", b))
            return solving(v,newEquation)
        elif isInside(v,b):
            newEquation = (b, "=", (c, "/", a))
            return solving(v,newEquation)
        else:
            return None

def solvingDivide(v,q):
        a = left(left(q))       # gets farthest left var within the left expression (i.e. if left is x + b, will give x)
        b = right(left(q))
        c = right(q)
        if isInside(v,a):
            newEquation = (a, "=", (c, "*", b))
            return solving(v,newEquation)
        elif isInside(v,b):
            newEquation = (b, "=", (a, "/", c))
            return solving(v,newEquation)
        else:
            return None

#
#  TESTS. Test the equation solver for CSci 1913 Lab 1.
#
#    James Moen
#    10 Sep 18
#
#  Every test is followed by a comment which shows what must be printed if your
#  code works correctly. It also shows how many points the test is worth, for a
#  total of 35 possible points.
#

print(isInside('x', 'x'))                           #  True   1 point
print(isInside('x', 'y'))                           #  False  1 point
print(isInside('x', ('x', '+', 'y')))               #  True   2 points
print(isInside('x', ('a', '+', 'b')))               #  False  2 points
print(isInside('+', ('a', '+', 'b')))               #  False  2 points
print(isInside('x', (('m', '*', 'x'), '+', 'b')))  #  True   2 points

print(solve('x', (('a', '+', 'x'), '=', 'c')))
#  ('x', '=', ('c', '-', 'a'))  2 points

print(solve('x', (('x', '+', 'b'), '=', 'c')))
#  ('x', '=', ('c', '-', 'b'))  2 points

print(solve('x', (('a', '-', 'x'), '=', 'c')))
#  ('x', '=', ('a', '-', 'c'))  2 points

print(solve('x', (('x', '-', 'b'), '=', 'c')))
#  ('x', '=', ('c', '+', 'b'))  2 points

print(solve('x', (('a', '*', 'x'), '=', 'c')))
#  ('x', '=', ('c', '/', 'a'))  2 points

print(solve('x', (('x', '*', 'b'), '=', 'c')))
#  ('x', '=', ('c', '/', 'b'))  2 points
```

```python

print(solve('x', (('a', '/', 'x'), '=', 'c')))
#  ('x', '=', ('a', '/', 'c'))  2 points

print(solve('x', (('x', '/', 'b'), '=', 'c')))
#  ('x', '=', ('c', '*', 'b'))  2 points

print(solve('y', ('y', '=', (('m', '*', 'x'), '+', 'b'))))
# ('y', '=', (('m', '*', 'x'), '+', 'b'))  2 points

print(solve('x', ('y', '=', (('m', '*', 'x'), '+', 'b'))))
# ('x', '=', (('y', '-', 'b'), '/', 'm'))  2 points

print(solve('a', (('b', '+', 'c'), '=', ('d', '*', (('a', '/', 'e'), '-', 'f')))))
# ('a', '=', (((('b', '+', 'c'), '/', 'd'), '+', 'f'), '*', 'e'))  5 points

```