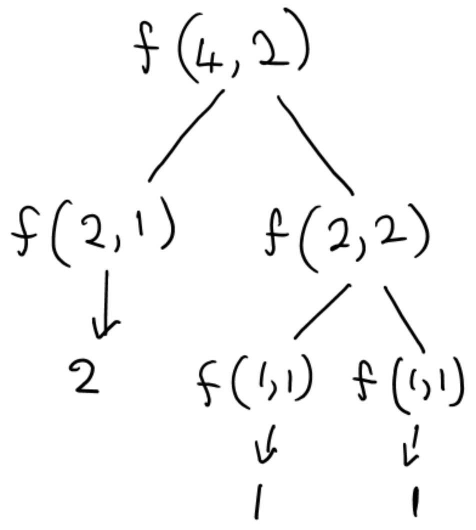
1. Glass floor
    a. Optimal solution: if we were given m glass sheets and n floors, we can use an algorithm similar to binary search: recursively diving the floors. If the glass breaks after being dropped from a floor, we would then know that we have to use the lower floors, else if it did not break, we -test it with higher floors.
    b.



d. with 4 floors and 2 glass, we end up with 3 distinct subproblems.

e. With n floors and m sheets, we would have at most n – 1 subproblems

f. To memoize this recursion:

1. We can create a two dimensional array (i,j) with lengths (floors +1, sheets+1).

2. have a nested loop

    a. if floor is the first, set memAry[i][j] to 1

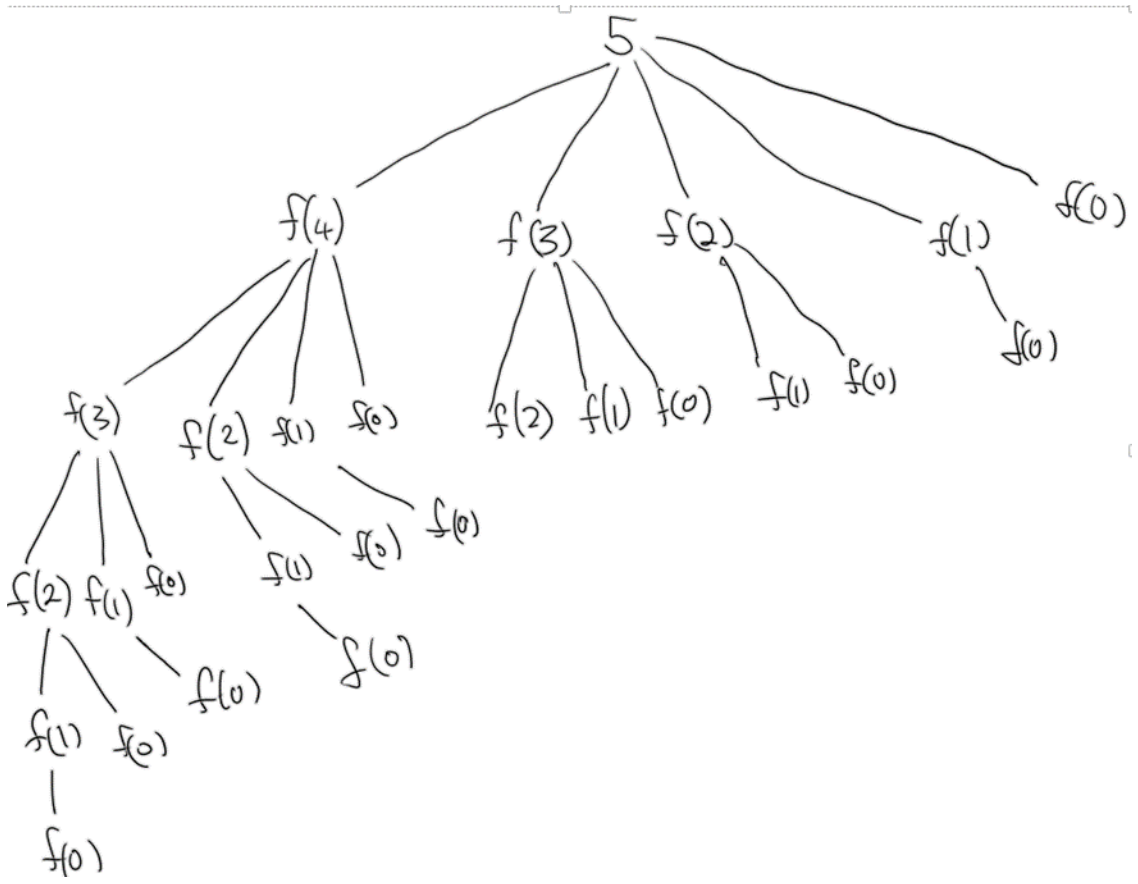
    b. if sheet is 1, set memAry[i][j] to i

    c. otherwise, we calculate the entry based on previous entries:

        i. dividing the floors into 2

        ii. retrieving the max value between memAry[i][j-1] and memAry[i][j]

2. ROD CUTTING


5
f(4)
f(3)
f(2)
f(1)
f(0)
f(3)
f(2)
f(1)
f(0)
f(2)
f(1)
f(0)
f(1)
f(0)
f(0)
f(2)
f(1)
f(0)
f(1)
f(0)
f(0)
f(1)
f(0)
f(0)
f(0)
f(1)
f(0)
f(0)
f(0)
f(0)