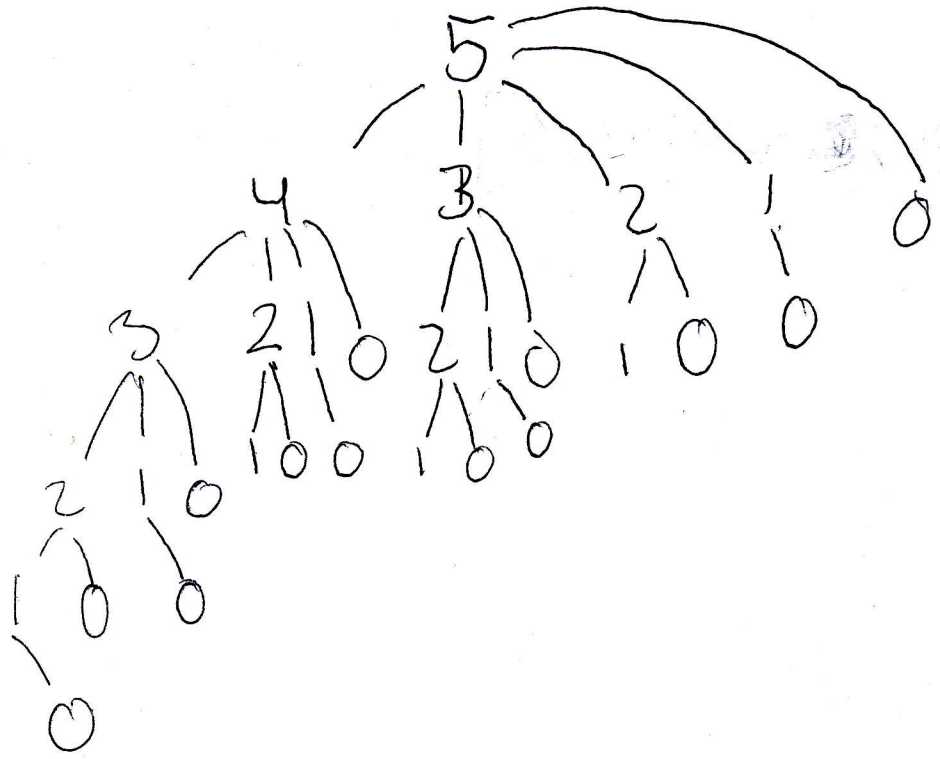
a) Rod Cutting

N=5.



→ Each level has the possible lengths that we may cut the given rod. At the root (5), we may decide not to cut the rod (0) or cut it into a "new" rod of size 1, 2, 3 or 4, and selling the remainder of that cut depending on price.

b) Rod Cutting

Let $N=4$.

$P=[1, 20, 33, 4]$.

where the i^{th} entry of $P(P_i)$ is the price of a rod of length i.

Then. $D=[1, 10, 11, 1]$

where the i^{th} entry of $D(D_i)$ is the density $\left(\frac{P_i}{i}\right)$ of a rod of length i.

Using the greedy method of taking the greatest density of the given rod length, k:

- Initially, $k=4$ since $N=4$. We have the option of cutting the rod by 1, 2 or 3 or leaving the rod as is and selling it. Since we want to grab the greatest density at every k^{th} length such that, $d \leq k$ where d is the index of greatest density. At $k=4$, $d=3$ since 11 is the most density in the interval, and therefore we will cut a rod of length 4 from our rod of length k. We will remain with $k' = k-4 = 1$ rod length left, and since we can only leave the rod as is to sell, we will have a profit of
  $P_4 + P_1 = 33+1 = 34$.
- However, taking the rod in two pieces of length 2 each, we would have a total profit of
  $P_2 + P_2 = 20+20 = 40$, which is a greater profit than the greedy approach.
- Therefore, the greedy solution does not always determine an optimal way to cut the rod.

a) Breaking Glass

At a given floor, i, we have two possible cases:

i) The glass breaks.

ii) The glass doesn't breaks.

Case i:

If the glass breaks on the i^{th} floor, we need to only check all floors less than i $(1 \ldots i-1)$ since all levels above i are guaranteed to break glass, and therefore we will have to check $i-1$ floors and if s represents the number of sheets we have, $s-1$ sheets remain for these $i-1$ floors.

Case ii:

If the glass doesn't break, we just have to check the floors above i $(i+1 \ldots f)$ where f is the total number of floors. If s represents the initial number of sheets, we would still have s to check all above floors.

Recurrence:

we want to minimize the worst case trials, so if we try every floor, and recursively determine each floors. Minimized worst-case, we can determine the ~~worst~~ "best" worst case with n floors and m glass sheets.

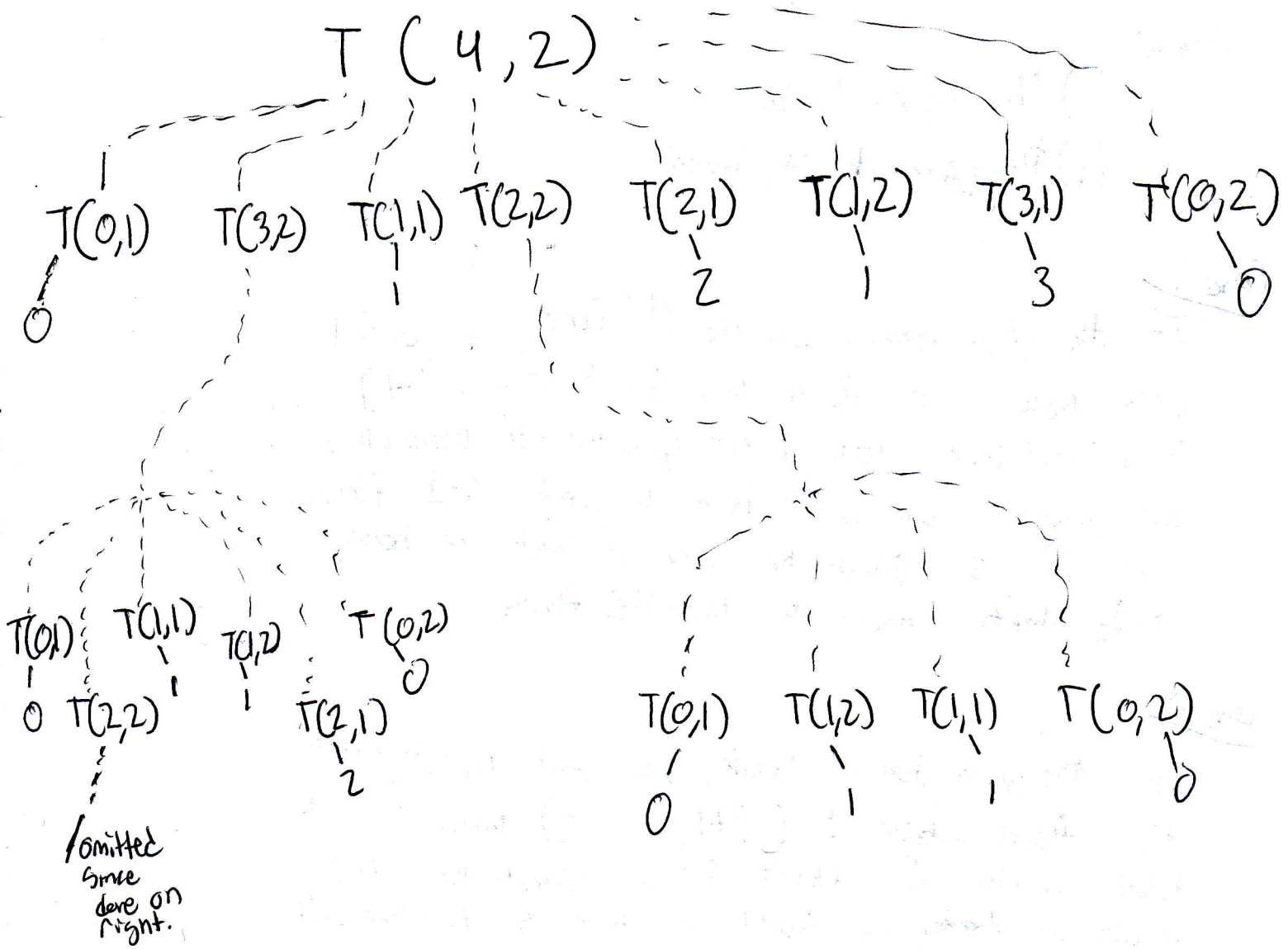
∴ MinimumTrials (n, m):

∴ for all floors, i from $1 \ldots n$.

$$\text{MinimumTrials}_n = 1 + \min\left(\max\left(\text{MinimumTrials}_{n-1}(n-1, m-1), \text{MinimumTrials}_{n-i}(n-i, m)\right)\right)$$

$$T(n,m) = \begin{cases} 0 & \text{if } n=0 \text{ or } m=0 \\ 1 & \text{if } n=1,\ m>0 \\ n & \text{if } m=1. \\ 1+\min\left\{ \max\left\{ T(n-1, m-1),\ T(n-i, m) \text{ for all } 1 \le i \le n. \right. \right. \end{cases}$$

b) Breaking Glass



d) Breaking Glass

# of distinct subproblems = 8
for n=4
m=2.

e) Breaking Glass

# of distinct subproblems for any $m, n \geq 0$ = $2 * n$

At every floor, it either breaks or does not.

f) To memoize, first I would initialize a $n \times n$ matrix. If i represents the row, j represents the column then:

- For every column where $i=0$, I would let the entry be 0, since $i=0$ corresponds to no glass available.
- For every column where $i=1$, I would let the entry be j, since if we only have one piece of glass, each floor would need to be checked one by one. In the worst case.
- For every row where $j=0$, I would let the entry be 0 since if we have no floors, no trials need to be done.
- For every row where $j=1$, I would let the entry be 1 since one floor only means one trial needs to be done.

Using these preset values, we can recurse from lower floors first and set our memo values as we increase floors, and therefore, all subproblems of lower floors would be covered until we reach entry $[n-1][n-1]$, which is our minimized worst case we are looking for.