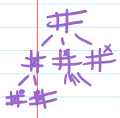
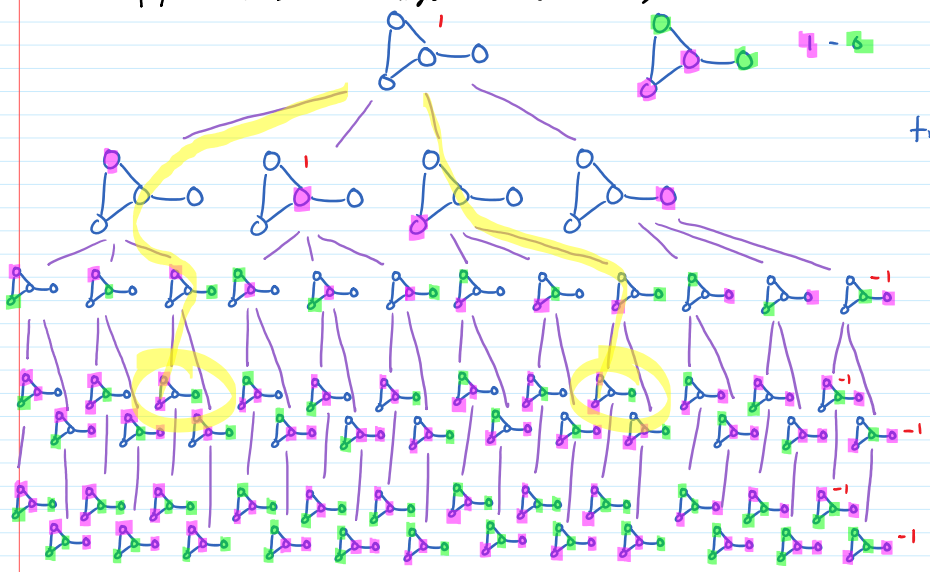
1 2 3 4 8 6 8 < 13

Divisors: Start with 1...n, players take turns taking a number with remaining divisors; opponent gets all the remaining divisors. Game is over when no moves remain; winner is player with higher sum (draw if =)



Graph: take turns coloring a vertex in a graph with your color player who covers the most edges wins (draw if =)



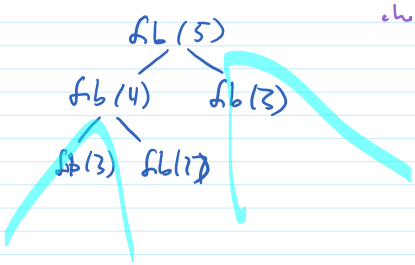
transposition table
- keeps track of values for nodes already seen

Minimax(p)

if p is end of game
  return value according to rules
else
  let S = positions reachable in 1 move from p
  if p is P1's turn
    return max Minimax(p')
           p'∈S
  else
    return min Minimax(p')
           p'∈S

```
def fib(n):
    if n<2: add (n,n) to memo
        return n
    else:
        return fib(n-1) + fib(n-2)
```

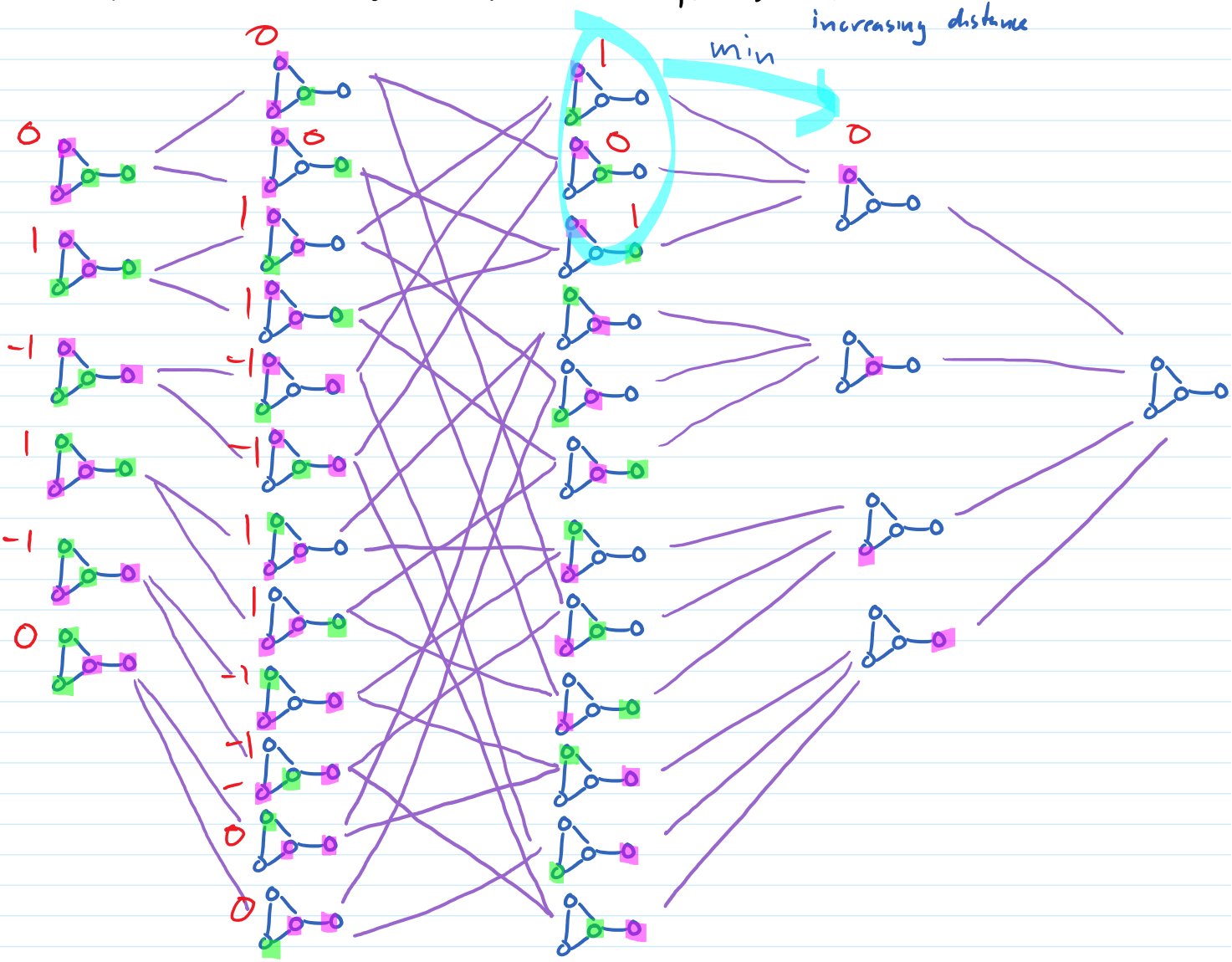
check if n-1, x is in memo
if so, use value in memo
check if (n-2, y) in memo



Order positions by maximum distance to end.

Determine winner of distance 0 positions (end) by applying rules

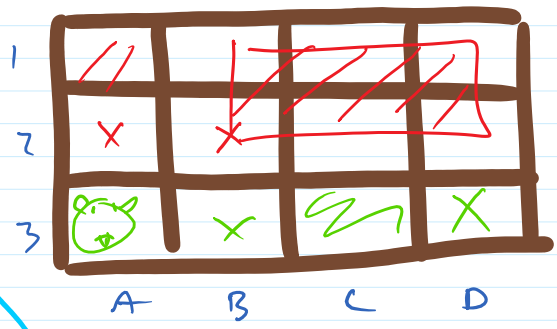
Use recursive formula to determine value of other positions in order of increasing distance



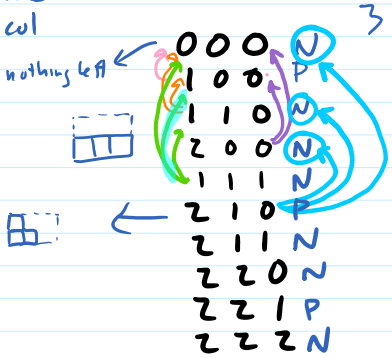
Chomp

Play on mxn grid. Take turns selecting remaining cell, remove all above and to right.

Last move loses



2 row
3 col



P: prev player has winning strat
N: next player has winning strat

```
Minimax(p)
    if p is end of game
        return value according to rules
    else
        let S = positions reachable in 1 move from p
        if S contains a P position
            return N
        else
            return P
```

Could use dynamic programming:

top/mid/bot 000
100
010
001
110
101
011
200
020
002
111
000 ← 210
201
⋮

but Nim is finite, impartial, and normal (Kayles is too, can find equiv to Nim)

Nim-sum

For Nim, there is a winning move if and only if the bitwise exclusive or of the number of stones left in each row is non-zero, and the winning moves are the ones that make the bitwise exclusive or 0.

(So a position is an N-position if and only if Nim-sum ≠ 0)

0 0 0 0	4	0100
0 0 0 0 0 0 0	7	0111
0 0 0 0 0 0 0 0 0 0	10	1010
		1001

1010
1001
0011 = 3

compute Nim-sum x
find most sig bit of x
find term (row) with msb set, let n = # stones
reduce that row to $n \oplus x$ stones (now Nim-sum is 0)