WEEK3: DYNAMIC PROGRAMMING

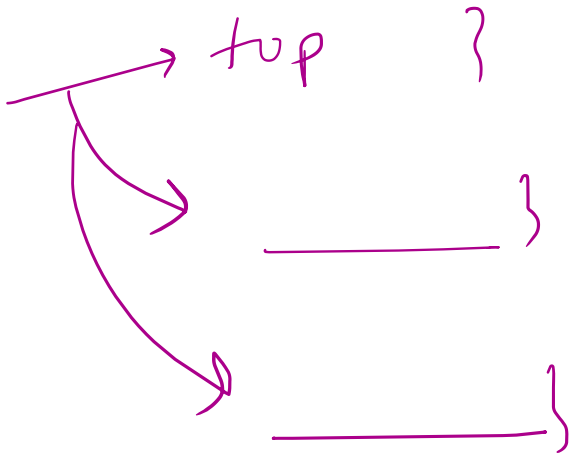
Agenda

Survey Comments

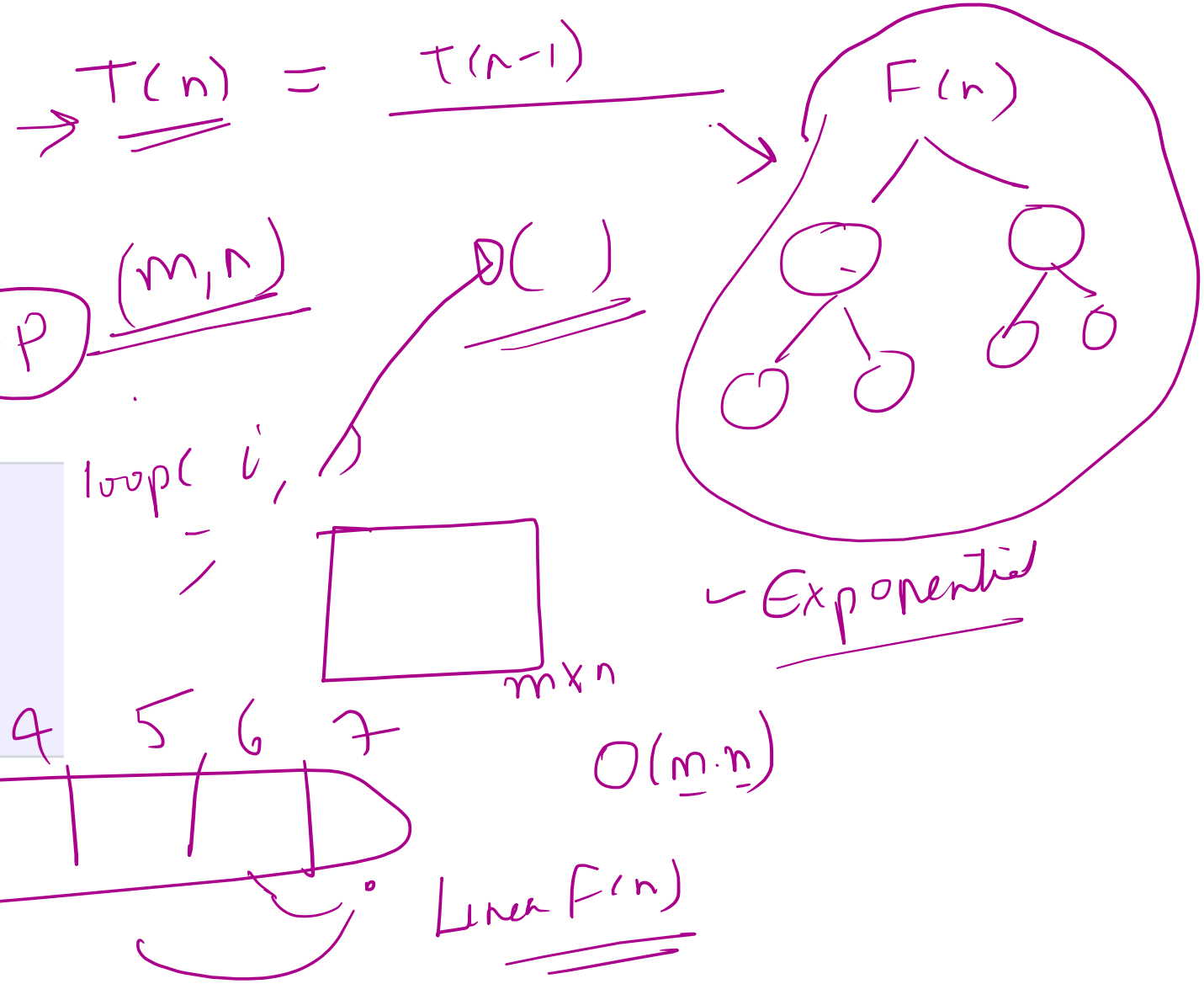
- Can you go over calculating **time complexity** of dynamic programming problems?
- can we go over more examples of analyzing time complexity of more difficult algorithms?
- Can we **go over the program examples** presented in this weeks explorations?
- If we could cover a Longest Common Sequence problem that would be really helpful!

- how to find a recurrence relation?

$$F(n) = \begin{cases} 1 & \text{If } n = 0 \\ 1 & \text{If } n = 1 \\ F(n-1) + F(n-2) & \text{otherwise} \end{cases}$$



$$F(n) = \begin{cases} 1 & \text{If n = 0} \\ 1 & \text{If n = 1} \\ F(n-1) + F(n-2) & \text{otherwise} \end{cases}$$

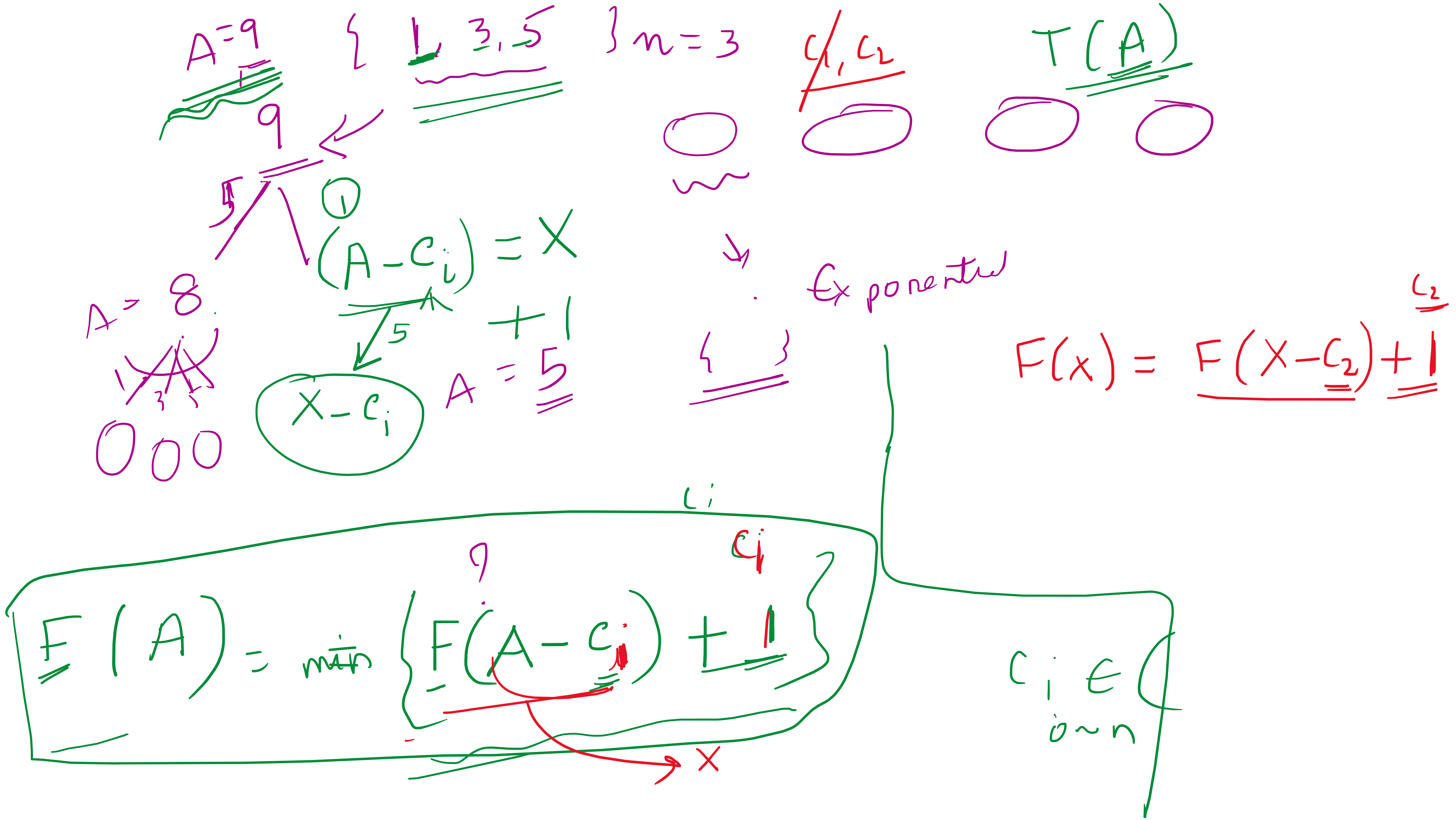



```
FibMemo = {0:1, 1:1}
def Fib_top_down(n):
    if n in FibMemo:
        return FibMemo[n]
    FibMemo[n] = Fib_top_down(n-1) + Fib_top_down(n-2)
    return FibMemo[n]

print(Fib_top_down(5))
```

DP algorithms – Running Time

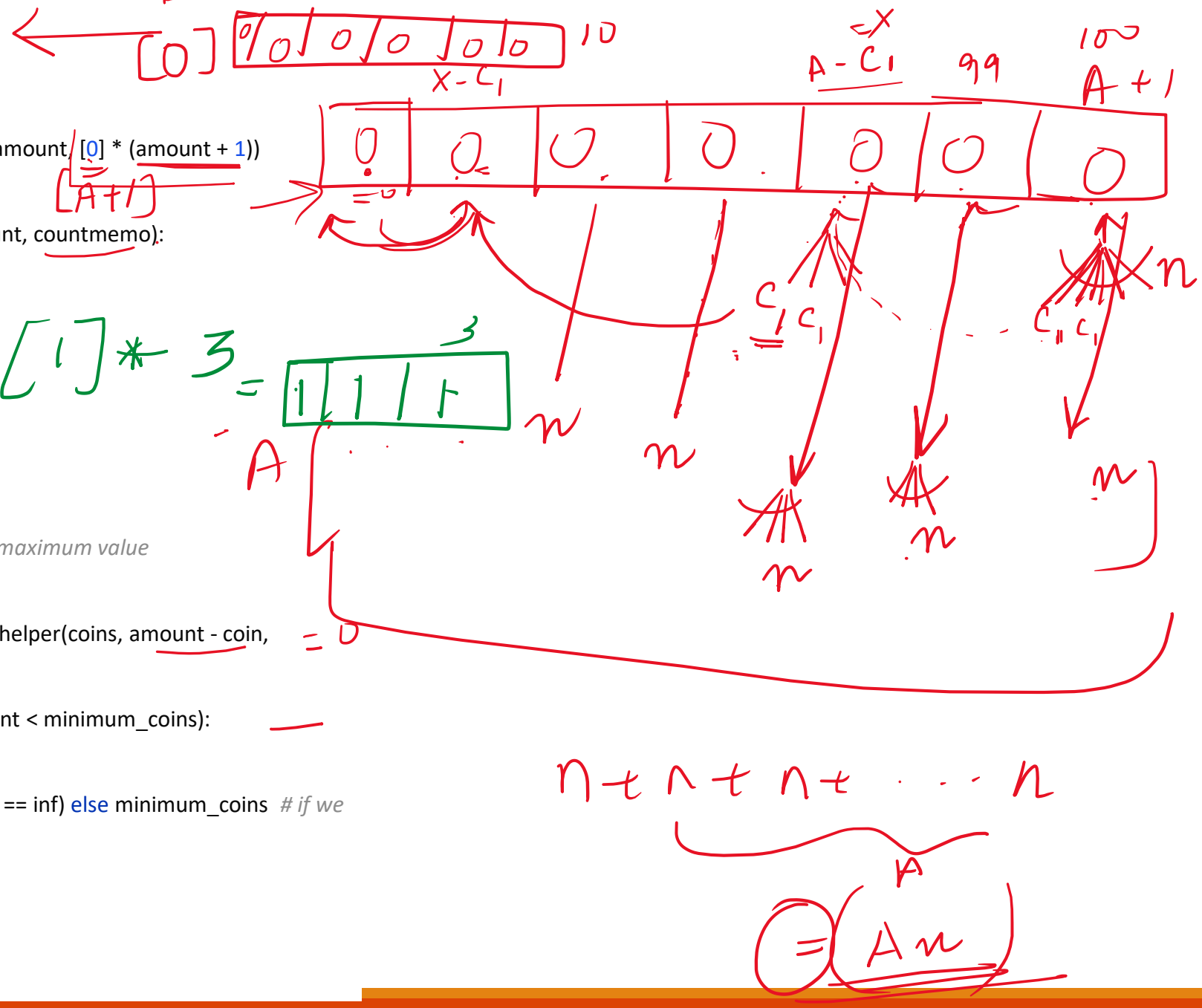
- Running time for dynamic programming algorithms.

  Fibonacci number

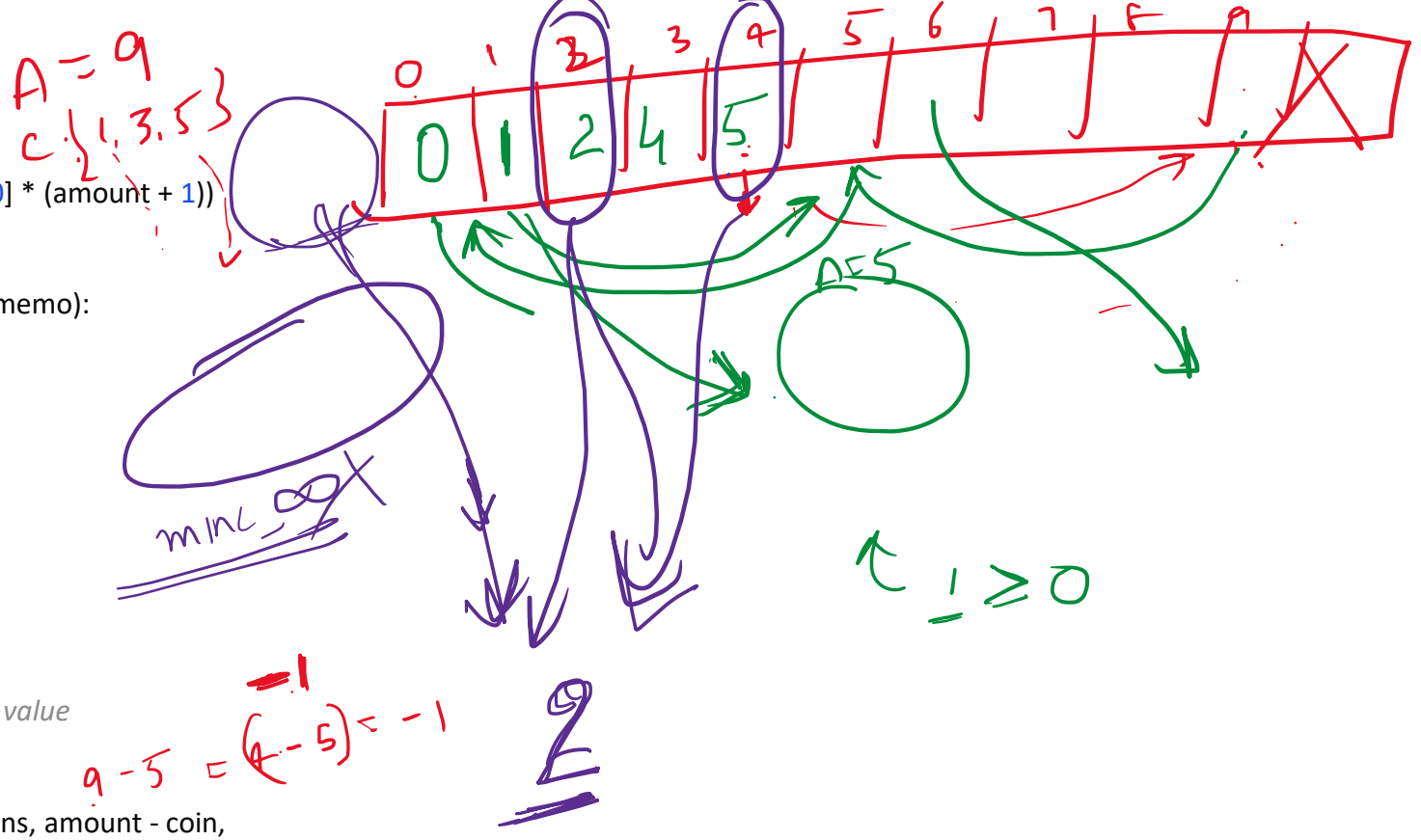


A = 9
{ 1, 3, 5 } n = 3
c1, c2
T(A)
(A − ci) = X
A = 8
X − ci
A = 5
+ 1
Exponential
F(x) = F(X − c2) + 1
F(A) = min { F(A − ci) + 1 }
ci ∈ (0 ~ n

```
import sys
def makechange_topdown( coins, amount):
    if amount == 0:
        return 0

    return makechange_topdown_helper(coins, amount, [0] * (amount + 1))


def makechange_topdown_helper( coins, amount, countmemo):
    if (amount < 0):
        return -1

    if (amount == 0):
        return 0

    if (countmemo[amount] != 0):
        return countmemo[amount]

    inf = sys.maxsize
    minimum_coins = sys.maxsize  # set to some maximum value

    for coin in coins:
        temp_coincount = makechange_topdown_helper(coins, amount - coin,
countmemo)

        if (temp_coincount >= 0 and temp_coincount < minimum_coins):
            minimum_coins = 1 + temp_coincount

    countmemo[amount] = -1 if (minimum_coins == inf) else minimum_coins  # if we
found a new minimum use it
    print(countmemo)
    return countmemo[amount]


print(makechange_topdown([1,2,3], 4))
```



```
import sys
def makechange_topdown( coins, amount):
    if amount == 0:
        return 0

    return makechange_topdown_helper(coins, amount, [0] * (amount + 1))


def makechange_topdown_helper( coins, amount, countmemo):
    if (amount < 0):
        return -1

    if (amount == 0):
        return 0

    if (countmemo[amount] != 0):
        return countmemo[amount]

    inf = sys.maxsize
    minimum_coins = sys.maxsize  # set to some maximum value

    for coin in coins:
        temp_coincount = makechange_topdown_helper(coins, amount - coin,
countmemo)

        if (temp_coincount >= 0 and temp_coincount < minimum_coins):
            minimum_coins = 1 + temp_coincount

    countmemo[amount] = -1 if (minimum_coins == inf) else minimum_coins  # if we
found a new minimum use it
    print(countmemo)
    return countmemo[amount]


print(makechange_topdown([1,2,3], 4))
```



min coins A

```
def makechange_bottomup(coins, amount):
    min_count_table = [amount + 1] * (amount + 1)  # setting array elements to some large value that is
not possible answer

    min_count_table[0] = 0  # setting the base case

    for i in range(1, amount + 1):  # iterate through all possible amount values from base case
        for j in range(0, len(coins)):  # find the number of coins needed for each coin denomination
            coin_val = coins[j]
            if (coin_val <= amount and (
                    i - coin_val) >= 0):  # if denomination value is less than amount then we can use the coin
                # replace min_count_table[i] with minumum value of coins possible
                min_count_table[i] = min(min_count_table[i], min_count_table[i - coin_val] + 1)

    # we have a valid count of coins if min_count_table[amount] is valid
    if min_count_table[amount] > amount:
        result = -1
    else:
        result = min_count_table[amount]

    return result


print(makechange_bottomup([1, 3, 5], 8))
```

O(A×n)

LCS

Sm =

Sn

abcde

abf

[0] * [n] =

afb

Ex I/p o/p ba

bca

ba

L.C.sub

- How can we implement an algorithm using dynamic programming?

- DP Approaches : **Bottom-up approach and Top-Down approach**

- algorithm implementation
  - Debug existing code / your code on pycharm
  - Visualize code on https://pythontutor.com/visualize.html#mode=edit

General

- More dynamic programming examples

- https://www.geeksforgeeks.org/top-20-dynamic-programming-interview-questions/
- https://www.interviewbit.com/courses/programming/topics/dynamic-programming/

★ Form Groups / Find friends to go discuss code