ER2023

① $a = 1$ (1 rec. call)

$b = 2$ (tamanho input $\frac{1}{2}$)

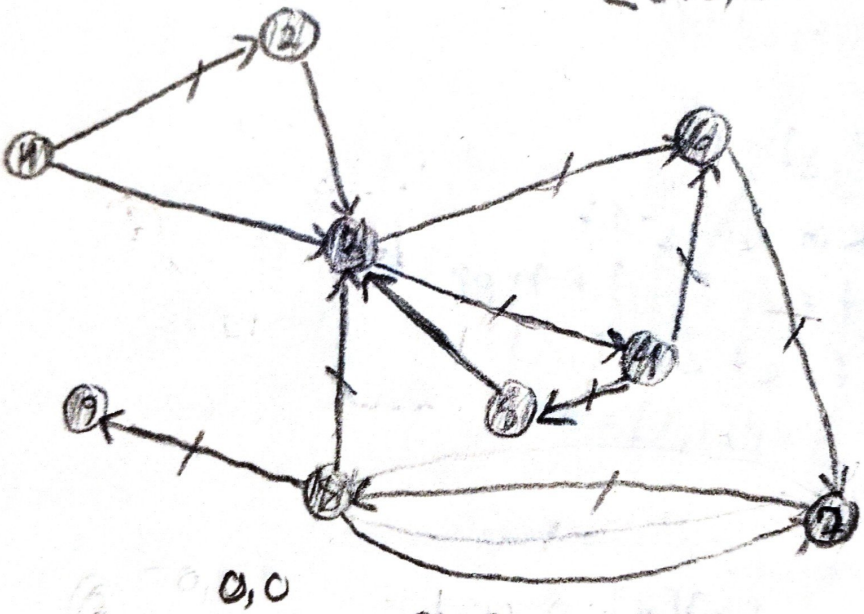
$c = 1$ (trabalho n/ recursivo é loop de $1, \ldots, n/2$)

$T(n) = T(\frac{n}{2}) + n$

$\log_b a = \log_2 1 = 0 < 1 = c$

$\Theta(n)$

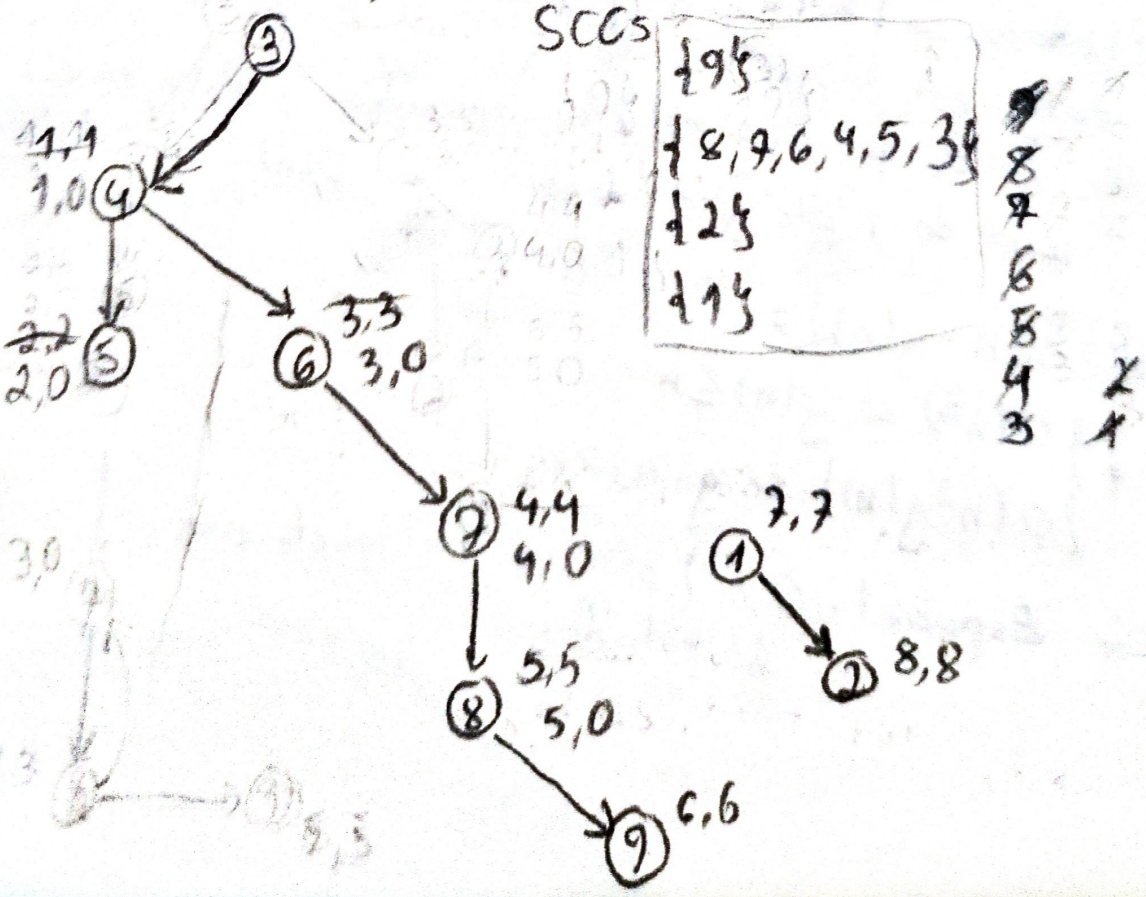
② <dfs, low>



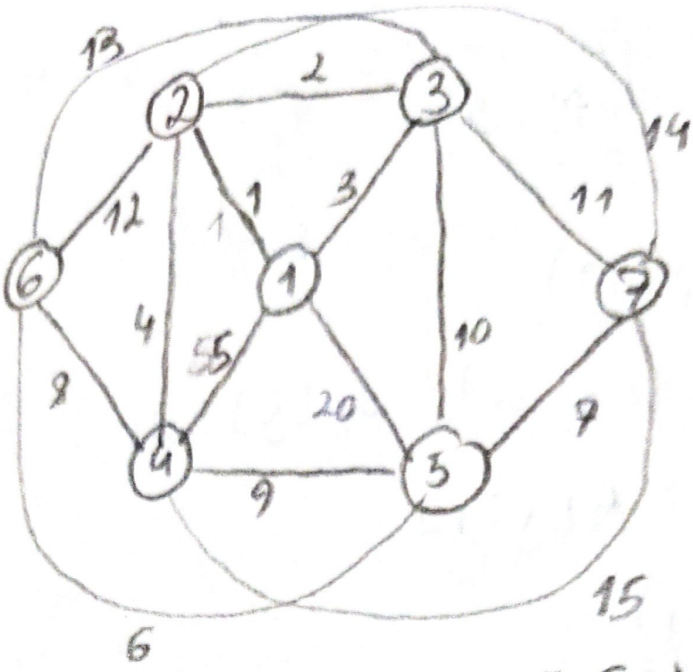
0,0

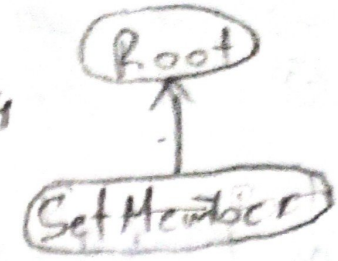
SCCs

{9}
{8, 7, 6, 4, 5, 3}
{2}
{1}



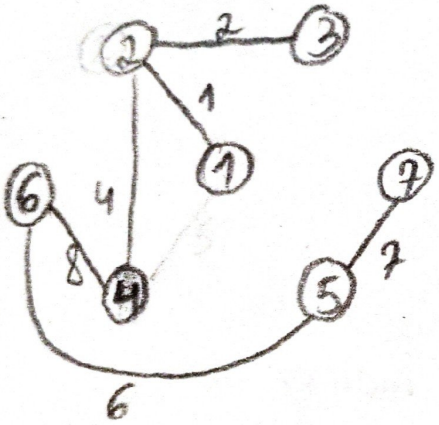
③

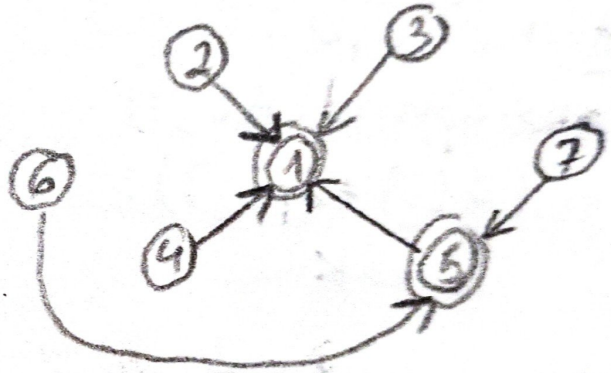




DSU

MST





④ $\bar{L} = \frac{1}{n}\sum_{i}^{1} L_i$

Base Case: $n = 1$

$mean(L, 1) = L_1$

$\bar{L} = \frac{1}{1} L_1 = L_1$

Inductive Hypothesis:

$\frac{L[K]}{K} + \left(mean(L, K-1) \times \frac{K-1}{K}\right) =$

$mean(L, K) = \frac{1}{K}\sum_{i}^{K} L_i$

Inductive Step:

Para $K+1$ elementos:

$$\frac{L[K+1]}{K+1} + \left(mean(L, K) \times \frac{K}{K+1}\right) = \frac{L[K+1]}{K+1} + \left(\frac{K}{K(K+1)}\sum_{i}^{K} L_i\right) =$$

$$= \frac{L[k+1]}{K+1} + \frac{1}{K+1}\sum_{i}^{K} L_i = \frac{1}{(K+1)}\sum_{i}^{K+1} L_i \;//$$

⑤

	-2	1	-3	4	-1	2	1	-5	4
dp =	②	①	-2	④	3	5	⑥	1	5

↑ Largest

→ Find $\max_{0 \le i < n}(dp(i))$

→ Go through i to 0 until $dp[j] = A[j]$

→ Max subarray = $A[j:i]$

```
i := max_{0<=i<n} (dp(i))
ans := []
for j in i, ..., 0:
    ans.append(A[j])
    if (A[j] = dp[j]):
        break

return ans.reversed
```

⑥ dp[i][sum1][sum2] = False para todos os índices

↳ "Os primeiros i elementos podem formar 2 subconjuntos cujas somas são sum1 e sum2?"

```
function check(N, n, S):
    dp(0,0,0) = True
    for i in 1, ..., n:
        cur = N[i-1]
        for s1 in 0, ..., S/3:
            for s2 in 0, ..., S/3:
                dp[i,s1,s2] = dp[i-1, s1, s2]
                if(s1 >= cur and dp(i-1, s1-cur, s2) = True):
                    dp(i, s1, s2) = True
                if(s2 >= cur and dp(i-1, s1, s2-cur) = True):
                    dp(i, s1, s2) = True
return dp[n, S/3, S/3]
```

Time Complexity: $O(n S^2)$

$n \times \frac{S}{3} \times \frac{S}{3}$ estados preenchidos na tabela dp.

Space Complexity: $O(n S^2)$

tabela dp de tamanho $n \times \frac{S}{3} \times \frac{S}{3}$

A solução está correta

Abordagem Correta pois:

Caso base: dp(0,0,0) = True está correto pois num set de 0 elementos, podemos formar 2 subsets cuja soma dos elementos é 0

Relações dentro da tabela: Em cada passo, são consideradas 3 opções para o elemento N[i]:

→ Colocar no terceiro subset:

dp[i, s1, s2] permanece igual a dp[i-1, s1, s2] pois os estados da tabela dependem dos 2 primeiros conjuntos

→ Colocar no primeiro subset:

se dp[i-1, s1 - N[i], s2] é verdade, então colocando N[i] no 1º conjunto faz com que dp[i, s1, s2] seja verdade

→ Colocar no segundo subset (mesma explicação)

⇓

Então, o elemento $dp[n, \frac{S}{3}, \frac{S}{3}]$ guarda a resposta para se os primeiros n elementos conseguem criar 2 subsets cujas somas dos elementos são $\frac{S}{3}$ (o terceiro terá, automaticamente, sum = $\frac{S}{3}$)