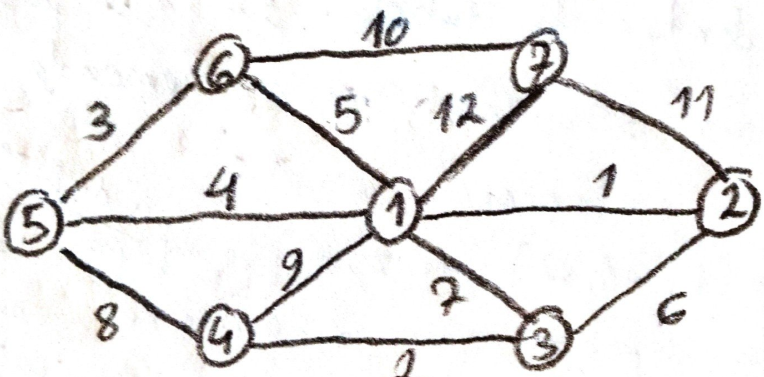
EN 2024

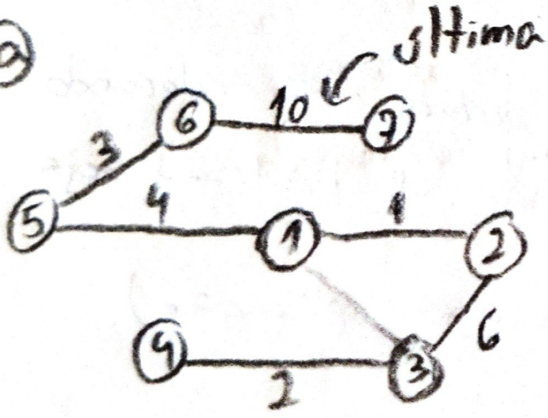
(1)

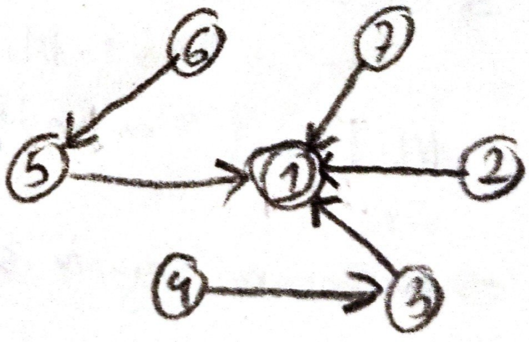
```
function C(n, cur, rem):
    if len(cur) ≥ n/2
        print(cur)

    for i in 0, ..., |rem|
        add := rem[i]
        new_comb := cur ∪ {add}
        new_rem := rem[i+1 :]
        C(n, new_comb, new_rem)
```

(2)

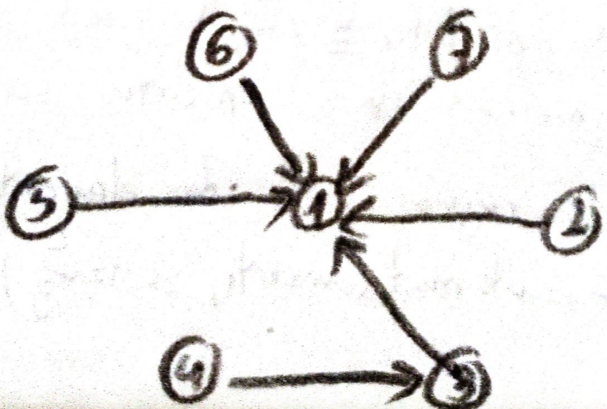


(a)

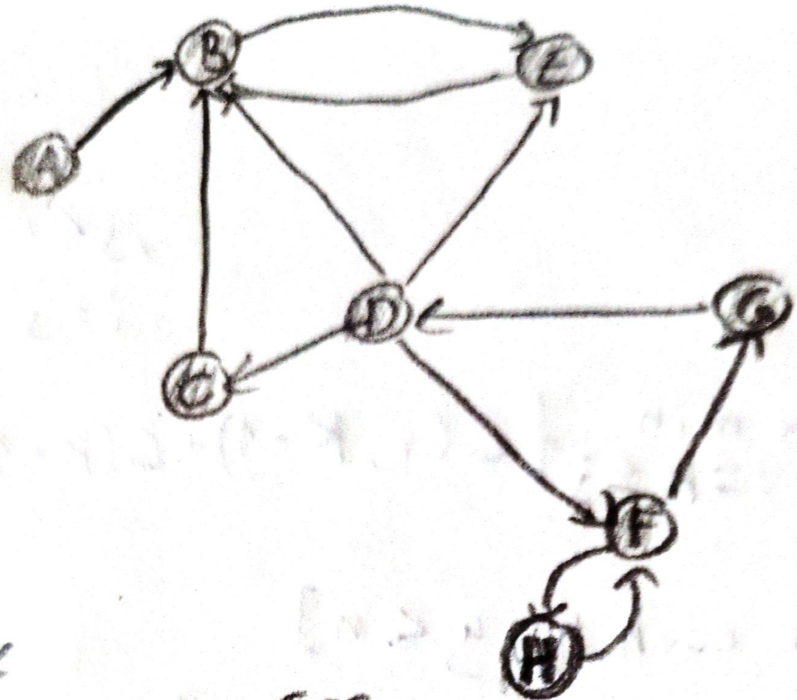




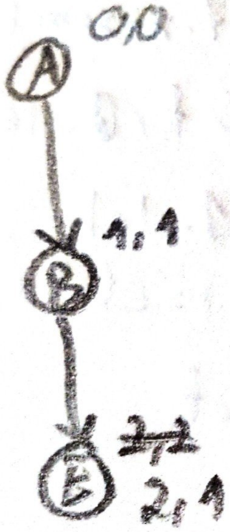
(b)



3



DFS Tree



SCCs:

{E, B}

{A}

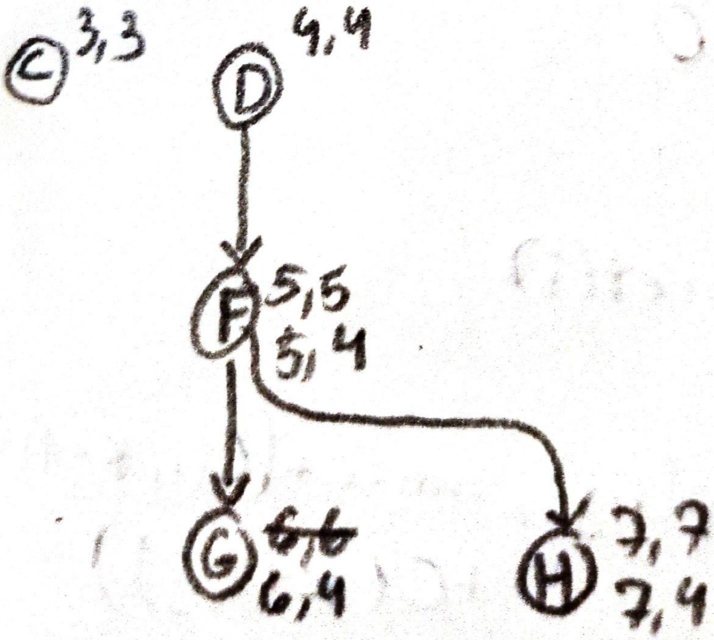
{C}

{H, G, F, D}

E B A

C

H G F D



4) $i = 1, \dots, n$
$j = 1, \dots, n$
$c_i \geq 0$

$$C(i,j) = \begin{cases} 0 & , j < i \\ c_i & , i = j \\ \sum_{\ell=i}^{j} c_\ell + \min_{i \leq k \leq j} \{ C(i, k-1) + C(k+1, j) \} & , \text{else} \end{cases}$$

dp[x][y] = [-1 : for each x, y < n]

```
function C(i, j, pref):
    if dp(i, j) ≠ -1:
        return dp(i, j)
    if i = j:
        return dp(i, j) := c_i
    if j < i:
        return dp(i, j) := 0
    else:
        dp(i, j) := ∞
        sumc := pref(j) - pref(i)
        for k in i, ..., j:
            dp[i, j] = min(dp[i, j], ~~sum c~~ + (C(i, k-1) +
                       + C(k+1, j))
        return dp[i, j] = dp[i, j] + sumc

function prefix-sum(c)
    pref(i) = [0 for 0 .. |c|]
    for i in 1 .. |c|
        pref(i) = c[i] + pref(i-1)
    return pref

C(1, n, prefix-sum(c))
```