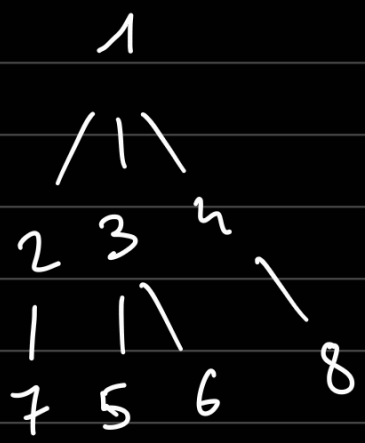
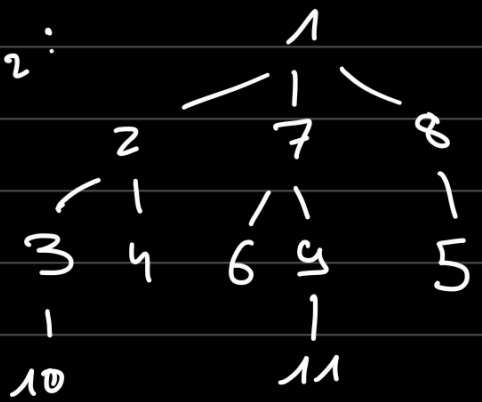
Exercice 1 : BFS = distance du point d'origine

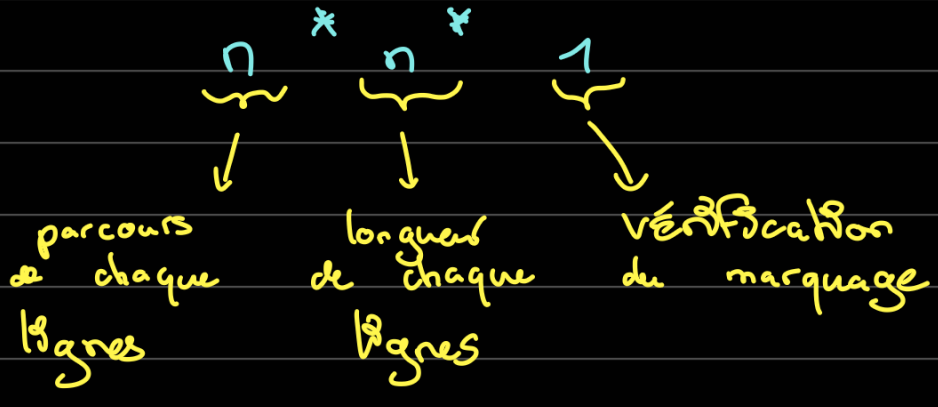
G_1 :



G_2 :



Exercice 2 :



Exercice 3 :

```
def distances(G, s)
    for u in V(G):
        d(u) = ∞
    dist(s) = 0
    Q = [s]
    while Q ≠ 0:
        u = defiler(Q)
        ∀uv ∈ E:
            if d(v) = ∞
                d(v) = d(u) + 1
                enfiler(Q, v)
```

Correction :

1) `def distances(G(V,E), s):` — $s \in V$

```
n = |V|, m = |E|
d = [+∞] * n                    → O(n)
d[s] = 0
Q = [s]
while Q ≠ ∅ :                   ┐
    cur = Q.pop()               │
    for n in E[cur]             │
        if d[n] != +∞           │
            continue            │
        d[n] = d[cur] + 1       │  O(m)
        Q.append(n)             │
return d                        ┘
```

2) `def path(G(V,E), r, d):` — r : target

```
p = [r]                                          ┐
while d[p[-1]] != 0                              │
    cur = p[-1]                                  │
    for n in E[cur]:   # parmi les voisins du dernier noeud   │  O(m)
        if d[n] == d[cur] - 1                    │
            p.append(n)                          │
            break   # on n'ajoute qu'un seul voisin   │
return p                                         ┘
```

donc $O(n \cdot m)$ si on répète cet algorithme pour les n sommets

Exercice 4 sur Feuille :