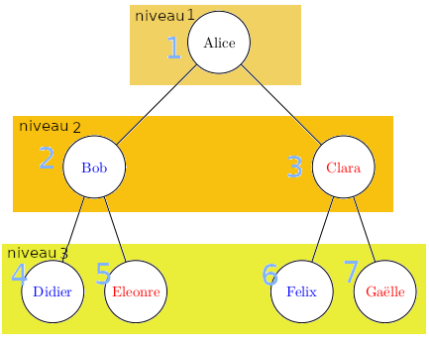
Algo 19 – **Parcours en largeur d'abord d'un Arbre Binaire**

La (presque) seule différence avec le parcours en profondeur vient de la structure linéaire utiliser :

Profondeur → **Pile** Largeur → **File**



Principe général

Si l'arbre binaire n'est pas VIDE

1. **Enfiler l'arbre** dans une File
2. **Tant que la File n'est pas vide**
    1. **Défiler le prochain arbre a.**
    2. Explorer la racine de l'arbre a
    3. **Enfiler le sous-arbre gauche (sag) de a** si sag n'est pas VIDE.
    4. **Enfiler le sous-arbre droite (sad) de a** si sad n'est pas VIDE.

Fonction du parcours en largeur d'abord (en version itérative) [et version File du pauvre à droite]

```
1   def parcours_largeur(arbre:'AB') -> None:
2       """Exploration en largeur de l'arbre (version itérative)"""
3
4       if not est_ABV(arbre):
5           f = nv_FV()
6           enfiler(f, arbre)                       # f.append(arbre)
7           while not est_file_vide(f):             # while f!= []:     ou len(f)!= 0
8               a = defiler(f)                      #     a = f.pop(0)  !! linéaire !!
9               print(contenu(racine(a)))
10              g = gauche(a)
11              if not est_ABV(g):
12                  enfiler(f, g)                   #     f.append(g)
13              d = droite(a)
14              if not est_ABV(d):
15                  enfiler(f, d)                   #     f.append(d)
```

Fonction du parcours en programmation récursive

```
1   def parcours_largeur_2(arbre:'AB') -> None:
2       """Exploration en largeur de l'arbre (version récursive)"""
3
4       if not est_ABV(arbre):
5           f = nv_FV()
6           enfiler(arbre, f)
7           exploration_suivante(f)
8
9   def exploration_suivante(f:'File') -> None:
10      """Affiche la prochaine racine et
11       stocke dans la File f les deux prochains sous-arbres détectés"""
12
13      if not est_FV(f):   # Cas récursif
14          a = defiler(f)
15          print(contenu(racine(a)))
16          g = gauche(a)
17          if not est_ABV(g):
18              enfiler(f, g)
19          d = droite(a)
20          if not est_ABV(d):
21              enfiler(f, d)
22          return exploration_suivante(f)
```

La fonction récursive est **exploration_suivante()**. La fonction **parcours_largeur_2() ne sert en réalité qu'à la lancer correctement.**