UE- PYTHON ET STRUCTURES DE DONNÉES AVANCÉES

Durée : 4H (CM :1H30, TD :1H, TP :1H30)

Définition et caractéristiques

- ❖ Un tas (ou monceau) est un arbre binaire partiellement complet (arbre parfait) tel que
  - ✓tout nœud interne possède une clé supérieure ou égale aux clés de ses fils pour un tas max (binary maxheap)
  - ✓tout nœud interne possède une clé inférieure ou égale aux clés de ses fils pour un tas min (binary minheap)

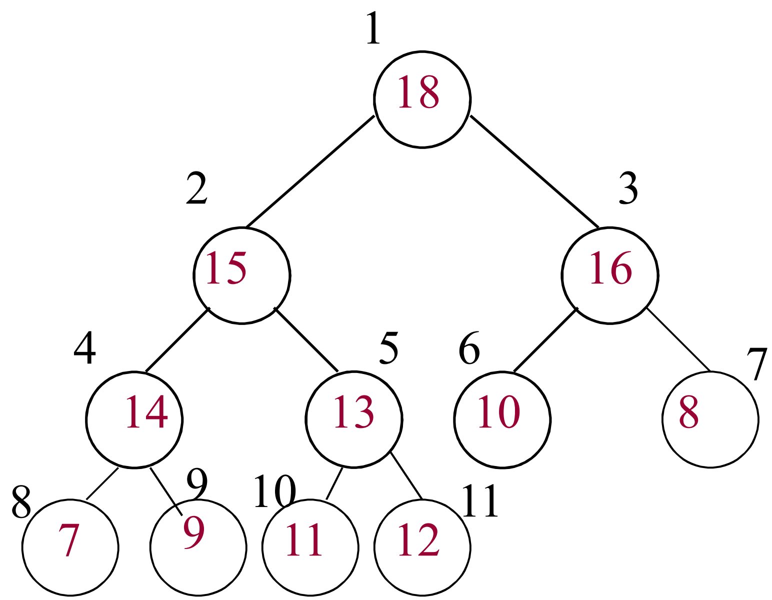
NB: Généralement, le tas max est appelé simplement tas

- ➢On a la propriété suivante :
  - ✓ Pour tous les nœuds de l'arbre autre que la racine, étiquette(*père*) ≥ étiquette(*fils*).

- ❖ Un tas est une structure de données qui
  - ◦ Permet un nouveau type de tri (Tri par tas)
  - ◦ Permet l'implémentation de files de priorité

Représentation par tableaux

- Un tas peut être représenté par un tableau T avec un accès en O(1) à chaque nœud:
  - ✓ On numérote les sommets par un parcours en largeur, de gauche à droite (Mémorisation des nœuds séquentiellement de la racine aux feuilles et de gauche vers la droite).
  - ✓ Fils gauche de T[i] est en T[2i]
  - ✓ Fils droit de T[i] est en T[2i + 1]
  - ✓ Père de T[i] est en T[i/2]



Si on traduit cette propriété sur la représentation T[1..n], on obtient :

$T[i] \geq T[2i]$ et $T[i] \geq T[2i + 1]$, $i \geq 1$, $2i + 1 \leq n$

Ou

$T[i] \geq T[2i]$, $i \geq 1$, $2i \leq n$

➢ $T[1] \geq T[i]$, $i \leq n$

tab T:

18	15	16	14	13	10	8	7	9	11	12
1	2	3	4	5	6	7	8	9	10	11

Quelques procédures

– La procédure PERCOLERs’exécute en *O*(lg *n*), permet de la conservation de la propriété de tas max.

– La procédure CONSTRUIRE-TAS-MAX, qui s’exécute en un temps linéaire, produit un tas max à partir d’un tableau d’entrée non-ordonné.

– La procédure ENTASSER , recevant pour paramètres un tas T et une valeur x et ajoutant x dans T en préservant la structure de tas

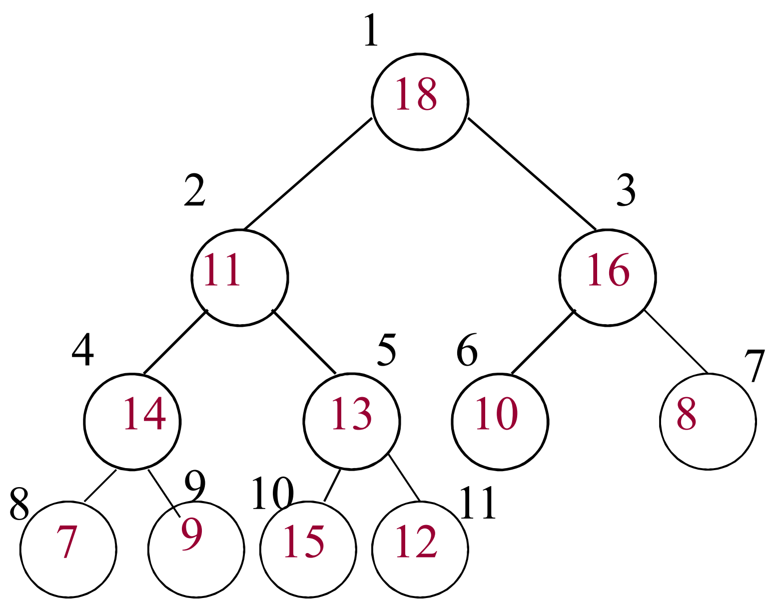
– Les procédures INSÉRER-TAS-MAX, EXTRAIRE-MAX-TAS, AUGMENTERCLÉ-TAS et MAXIMUM-TAS, qui s’exécutent en *O*(lg *n*), permettent d’utiliser la structure de données de tas pour gérer une file de priorité

Modification de la clé d'un noeud

❑Entrée : soient un tableau T et un indice i

❑ Condition :

✓ les arbres binaires Gauche(i) et Droite(i) sont des tas max
✓A[i] peut être plus petit que ses enfants (et donc un tri est nécessaire)
✓La procédure EntasserMax va faire évoluer l'arbre afin d'obtenir un tas max en i



Procédure EntasserMAX

Fonction PARENT(*i*)
retourner *i*/2

Fonction GAUCHE(*i*)
retourner 2*i*

Fonction DROITE(*i*)
retourner 2*i* + 1

```
Procédure EntasserMAX(T,i)
    Debut
        l ← Gauche(i)
        r ← Droite(i)
        si l ≤ taille(T) et T[l] > T[i]
            alors max ← l
            sinon max ← i
        FinSi
        si r ≤ taille(T) et T[r] > T[max]
            alors max ← r
        FinSi
        si max ≠ i
            alors Permute (T[i], T[max])
                  EntasserMAX(T,max)
        FinSi
    FIN
```

Procédure ConstruireTasMAX version1

On va effectuer un parcours ascendant par niveau et transformer si nécessaire les sous arbres rencontrés pour qu’ils vérifient la propriété du TAS.

```
Procédure ConstruireTasMAX (T[1..n] )
        i : entier
        Début
                i ← n div 2
                Tant que i ≥ 1 faire
                        EntasserMAX(T,i])
                        i ← i – 1
                Fin Tant que
        Fin ;
```

Procédure ConstruireTasMAX version2

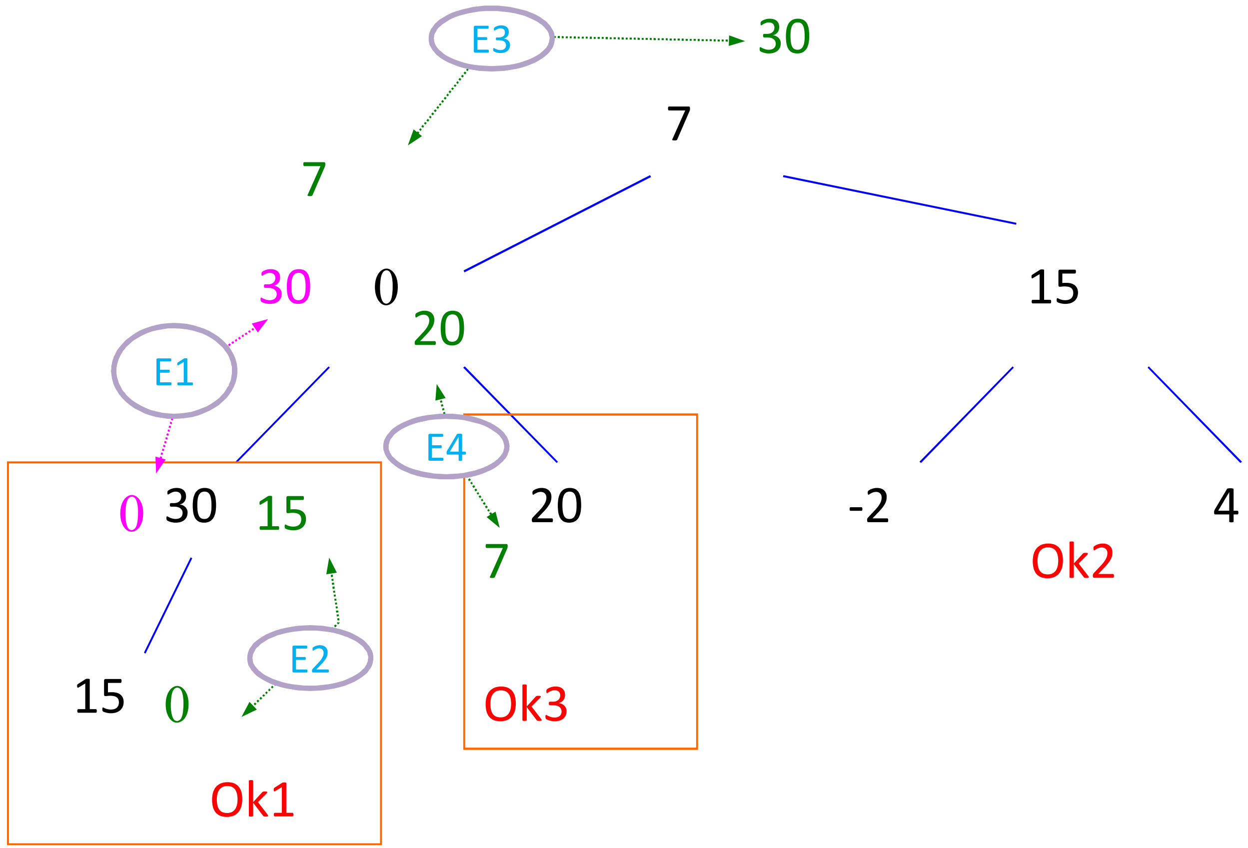
```
Procedure TransformeEnTasMAX (T[i..n] )
        FilsMax : entier
        Début
                FilsMax ← 2i                                # indice fils gauche, s’il existe
                Si FilsMax ≤ n alors                        # il y a au moins un sous arbre
                        Si FilsMax < n alors                # il y a 2 sous-arbres
                                Si T [FilsMax + 1] > T[FilsMax] alors
                                        FilsMax ← FilsMax + 1
                                FinSi
                                                #FilsMax désigne le plus grand des 2 fils
                        Fin Si
                        Si T[FilsMax] > T[i] alors
                                Permute (T[FilsMax], T[i])
                                TransformeEnTasMax (T[FilsMax..n])
                        Fin Si
                Fin Si
        Fin
```

Procédure ConstruireTasMAX version2

On va effectuer un parcours ascendant par niveau et transformer si nécessaire les sous arbres rencontrés pour qu’ils vérifient la propriété du TAS.

```
Procédure ConstruireTasMAX (T[1..n] )
        i : entier
        Début
                i ← n div 2
                Tant que i ≥ 1 faire
                        TransformeEnTasMAX (T[i..n])
                        i ← i – 1
                Fin Tant que
        Fin ;
```

Exemple



Algorithme

```
Procedure TrisTAS T([1..n] )
          Nb : entier
          Début
                    ConstruireTaxMAX (T[1..n])
                    Permute (t[1], T[n])
                    Nb ← n – 1
          Tant que Nb > 1 faire
                              TransformeEnTasMAX (T[1..Nb])
                              Permute (T[1], T [Nb])
                              Nb ← Nb – 1
                    Fin Tant que
          Fin ;
```

Définition et caractéristiques

- ❑Structure de données permettant de gérer un ensemble S d'éléments auxquels on associe une "clé"
- ✓Cette clé permet la définition des priorités

- ❑Application directe de la structure de tas
  - ❖4 opérations (File-Max)
    - ✓ MaximumTAS(T)
    - ✓ExtraireMaxTAS(T)
    - ✓AugmenterCleTAS(T,x,k)
    - ✓Insérer(T,x)
  - ❖Si on inverse l'ordre des priorités (File-Min), on obtient les opérations symétriques (Minimum, Extraire-Min, Diminuer-Clé)
- ❑Utilisation :
- ❖Cas d'une liste de tâches sur un ordinateur

- ✓La file de priorité gère les tâches en attente selon leur priorité
- ✓Quand une tâche est terminée, on exécute la tâche de priorité la plus élevée suivante
- ✓Il est possible d'insérer dans la file une tâche, éventuellement prioritaire

Procédures et fonctions

❖ Retourner l'élément ayant la clé maximale

```
Fonction MaximumTAS T([1..n] )
        Début
                Retourner (T[1])
        Fin
```

❖ Retourner l'élément ayant la clé maximale en le supprimant de la file

```
Fonction ExtraireMaxTAS (T[1..n] )
        max : entier
        Début
                max ← T[1]
                T[1] ← T[taille[T]]
                taille[T] ← taille[T]-1
                EntasserMax(T,1)
                Retourner(max)
        Fin
```

Procédures et fonctions

❖ Augmenter la valeur d'une clé

```
Procedure AugmenterCleTAS (T,i,clé )
        Début
          Si clé < T[i]
              alors ERREUR
           sinon
              T[i]  ← clé
               TantQue i> 1 et T[Parent(i)]<T[i]  faire
                       permute(T[i], T[Parent(i)])
                       i← T[Parent(i)]
                 Fin Tanque
             FinSI
        Fin
```

❖ Insérer un élément dans un Tas

```
Procedure InsererTasMAX(T, clé )
        Début
          taille[T] ← taille[T]+1
          AugmenterCleTAS (T, taille[T] ,clé )
        Fin
```