Structures de données abstraites

Sandrine Vial
sandrine.vial@uvsq.fr

Avril 2016

Une File

Définition

Analogie avec une file d'attente :
FIFO (First In First Out ou **Premier Arrivé Premier Servi**)

- On rajoute un élément à la fin de la file
- On supprime l'élément qui est en tête de file.

Une File

Opérations Principales

1. Insertion d'un élément
2. Suppression d'un élément (le plus ancien de la file)
3. Création d'une file vide
4. Tester si une file est vide
5. Quel est l'élément le plus ancien de la file ?
6. ...

Mise en œuvre

1. A l'aide d'un tableau (*nombre maximum d'éléments dans la file fixé*)

Type de données

```
typedef struct file {
  int T[NMAX];
  int debut;
  int fin;
} File;
```

Mise en œuvre : un tableau

- Créer une file vide :
  - *Entrée* : rien
  - *Sortie* : Une file avec aucun élément stocké.

```
File CreerFile()
{
  File F;

  F.debut  =  0;
  F.fin = F.debut;
  return F;
}
```

Mise en œuvre : un tableau

- La file est-elle vide :
  - *Entrée :* F (une file)
  - *Sortie :* vrai si la file est vide, faux sinon.

```
int FileVide(File F)
{
  if (F.debut == F.fin)
    return 1;
  else
    return 0;
}
```

Mise en œuvre : un tableau

- La file est-elle pleine :
  - *Entrée :* F (une file)
  - *Sortie :* vrai si la file est pleine, faux sinon.

```
int FilePleine(File F)
{
  if (F.debut == (F.fin + 1)%NMAX)
    return 1;
  else
    return 0;
}
```

Mise en œuvre : un tableau

- Insertion d'un élément :
  - *Entrée :* f (une file) et elt (un entier)
  - *Sortie :* la file f dans laquelle elt a été inséré

```
File InsererFile (File F, int e)
{
  if (!FilePleine(F))
    {
      F.T[F.fin] = e;
      F.fin = (F.fin + 1)%NMAX;
    }
  return F;
}
```

Mise en œuvre : un tableau

- Suppression d'un élément :
  - *Entrée :* f (une file)
  - *Sortie :* renvoie l'élément qui était au sommet de la file f et supprime l'élément de la file

```
int SupprimerFile(File *F)
{
  int e = -1;

  if (!FileVide(*F))
    {
      e = (*F).T[(*F).debut];
      (*F).debut = ((*F).debut + 1) % NMAX;
    }
  return e;
}
```

Utilisation d'une file

Exemple

```
File f ; int x ;

    f = CreerFile() ; ;
    f = InsererFile(f,12) ;
    f = InsererFile(f,34) ;
    f = InsererFile(f,23) ;
    x = SupprimerFile(&f) ;

    x = SupprimerFile(&f) ;
```

Utilisation d'une file

Exemple

```
File f ; int x ;
        f = CreerFile() ; ;
        f = InsererFile(f,12) ;
        f = InsererFile(f,34) ;
        f = InsererFile(f,23) ;
        x = SupprimerFile(&f) ;

        x = SupprimerFile(&f) ;
```

Utilisation d'une file

Exemple

```
File f ; int x ;

    f = CreerFile() ; ;
    f = InsererFile(f,12) ;
    f = InsererFile(f,34) ;
    f = InsererFile(f,23) ;
    x = SupprimerFile(&f) ;


    x = SupprimerFile(&f) ;
```

12
Début

Fin

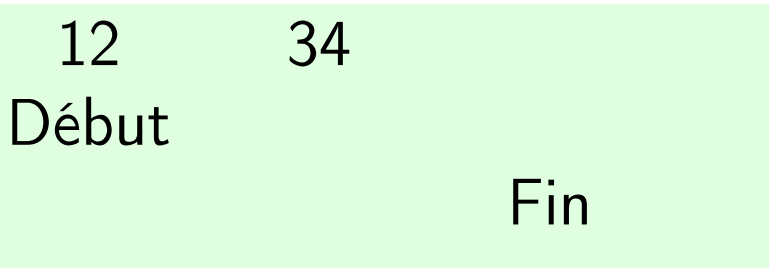
Utilisation d'une file

Exemple

```
File f ; int x ;

    f = CreerFile() ; ;
    f = InsererFile(f,12) ;
    f = InsererFile(f,34) ;
    f = InsererFile(f,23) ;
    x = SupprimerFile(&f) ;


    x = SupprimerFile(&f) ;
```

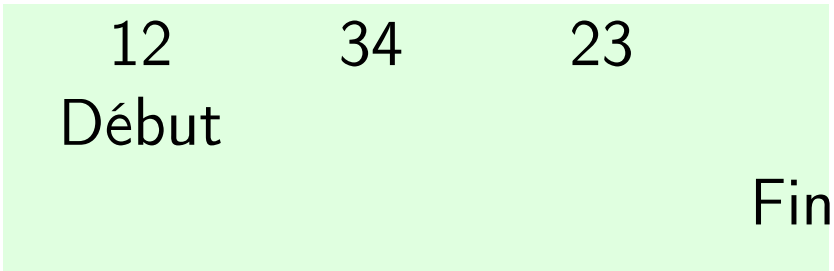


Utilisation d'une file

Exemple

```
File f ; int x ;
        f = CreerFile() ; ;
        f = InsererFile(f,12) ;
        f = InsererFile(f,34) ;
        f = InsererFile(f,23) ;
        x = SupprimerFile(&f) ;

        x = SupprimerFile(&f) ;
```



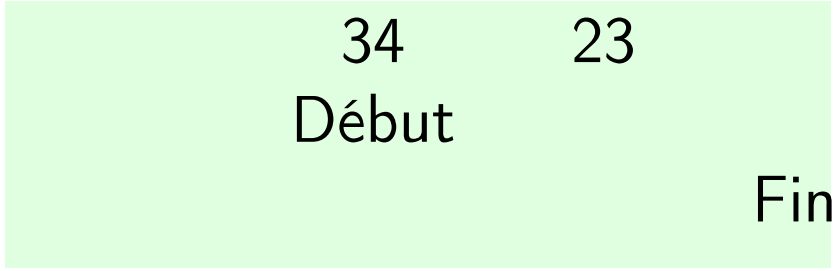
Utilisation d'une file

Exemple

```
File f ; int x ;

    f = CreerFile() ; ;
    f = InsererFile(f,12) ;
    f = InsererFile(f,34) ;
    f = InsererFile(f,23) ;
    x = SupprimerFile(&f) ;

    x = SupprimerFile(&f) ;
```

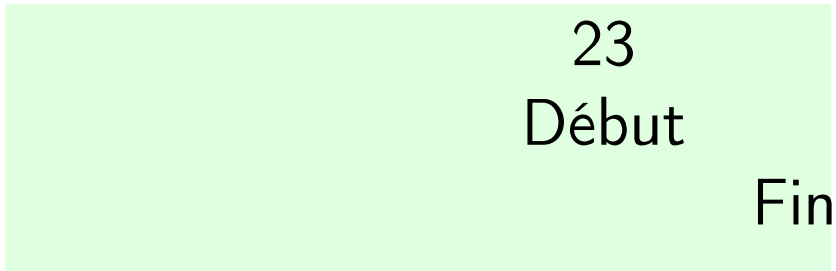


Utilisation d'une file

Exemple

```
File f ; int x ;
        f = CreerFile() ; ;
        f = InsererFile(f,12) ;
        f = InsererFile(f,34) ;
        f = InsererFile(f,23) ;
        x = SupprimerFile(&f) ;

        x = SupprimerFile(&f) ;
```



Un tas binaire

Definition

Un tableau T qui peut être vu comme un arbre A.

1. La **racine** du tas est en $T[1]$
2. Sachant l'indice i d'un élément (un **nœud**) du tas :
   - **Pere(i)** : élément d'indice $\lfloor \frac{i}{2} \rfloor$.
   - **Gauche(i)** : élément d'indice $2i$.
   - **Droit(i)** : élément d'indice $2i + 1$.

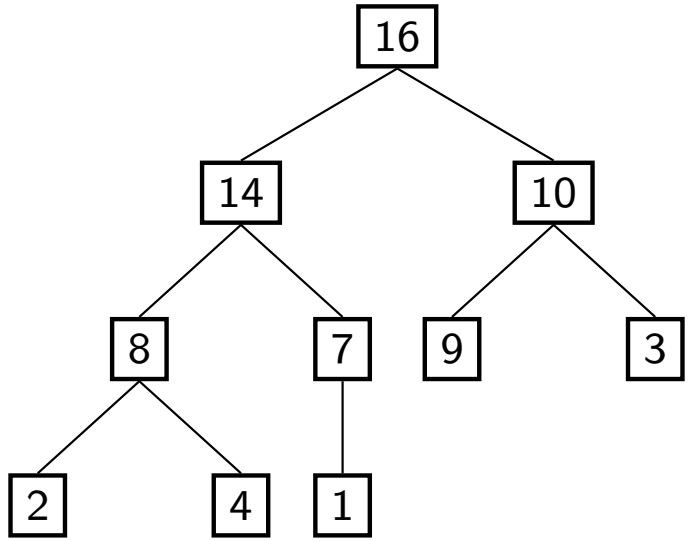
Propriété de tas

Pour chaque nœud i autre que la racine

$$T[Pere(i)] \geq T[i]$$

Un tas binaire

1	2	3	4	5	6	7	8	9	10
16	14	10	8	7	9	3	2	4	1



Un Tas Binaire : les opérations élémentaires

1. **Entasser** : sert à garantir le maintien de la propriété de tas binaire.
2. **Construire_Tas** : transforme un tableau non ordonné en tas binaire.
3. ...

Mise en œuvre

Algorithme 1 Entasser

Entasser(T : tableau, i : entier)

▷ ***Entrée : T (un tableau) et i un indice du tableau***

▷ ***Sortie : l'élément d'indice i est la racine d'un tas binaire***

▷ ***Pré-Conditions : Les éléments d'indices Gauche(i) et Droit(i) sont les racines de deux tas binaires.***

▷ ***Variables locales :***

l,r,max : entier ;

Début
l ← Gauche(i) ;
r ← Droit(i) ;
si (l ≤ taille (T) et T[l] > T[i])
max ← l ;
sinon
max ← i ;
fin si
si (r ≤ taille (T) et T[r] > T[max])
max ← r ;
fin si
si (max ≠ i)
T[i] ↔ T[max] ;
Entasser(T,max) ;
fin si
Fin

Mise en œuvre

Algorithme 2 Construction d'un tas binaire à partir d'un tableau

```
Construire_Tas(T : tableau, i : entier)
▷ Entrée : T (un tableau)
▷ Sortie : T est un tas binaire
▷ Variables locales :
    i : entier ;
Début
pour i de ⌊ longueur(T)/2 ⌋ à 1  faire
    Entasser(T,i) ;
fin pour
Fin
```

Une File de Priorité

Definition

Un ensemble dans lequel chaque élément possède une valeur et une priorité.

Opérations fondamentales

- **Insérer** : insère un nouvel élément dans l'ensemble.
- **Maximum** : renvoie l'élément de plus grande priorité.
- **Extraire_Max** : supprime de l'ensemble et renvoie l'élément de plus grande priorité.

Un tas permet de modéliser une file de priorité.

Tas = File de Priorité

Le plus grand élément du tas : $T[1]$.

Tas = File de Priorité

Algorithme 3 Extraire l'élément maximum

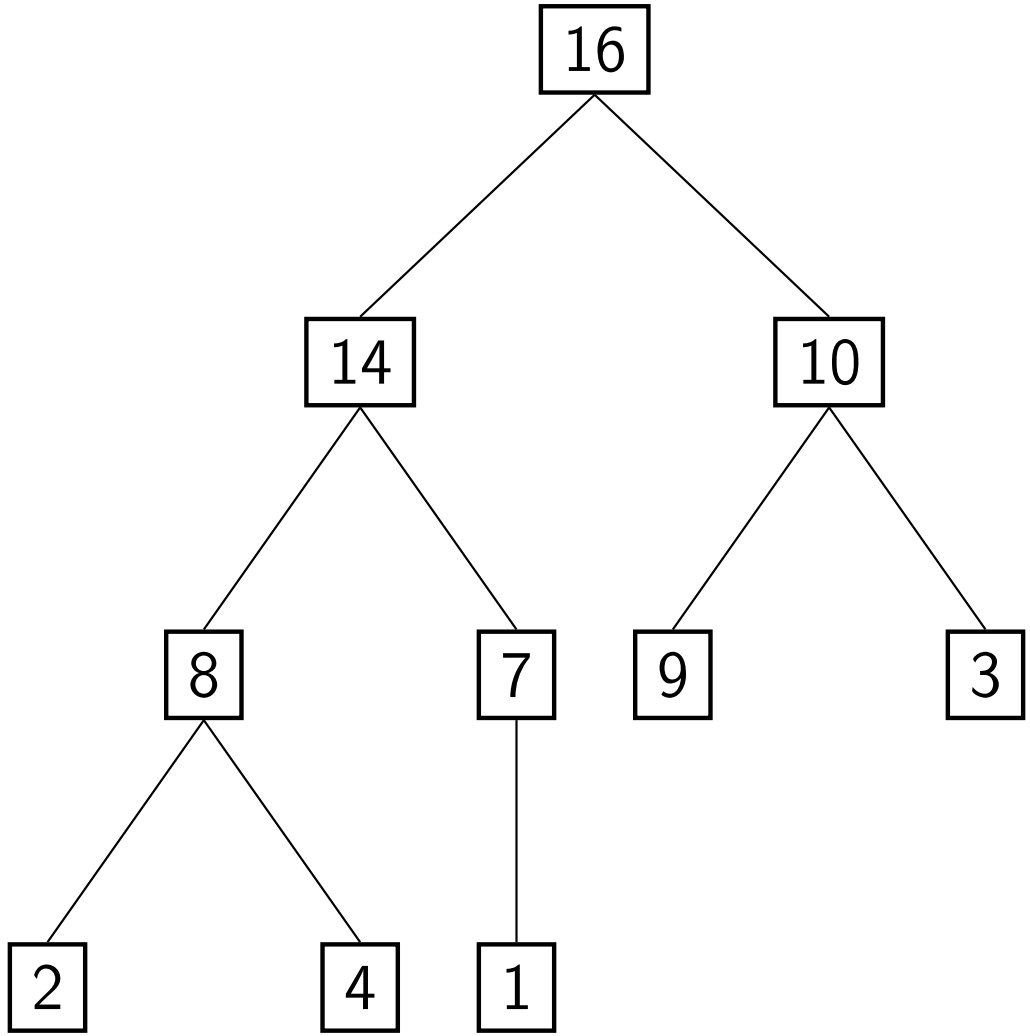
```
Extraire_Max(T : tableau) : entier
▷ Entrée : T (un tas binaire)
▷ Sortie : T est un tas binaire sans l'élément maximum et retourne l'élément maximum
▷ Variables locales :
    max : entier ;
Début
    si (taille(T) < 1)
        Afficher un message d'erreur ;
    fin si
    max ← T[1] ;
    T[1] ← T[taille(T)] ;
    taille(T) = taille(T) - 1 ;
    Entasser(T,1) ;
    retourner max ;
Fin
```

Tas = File de Priorité

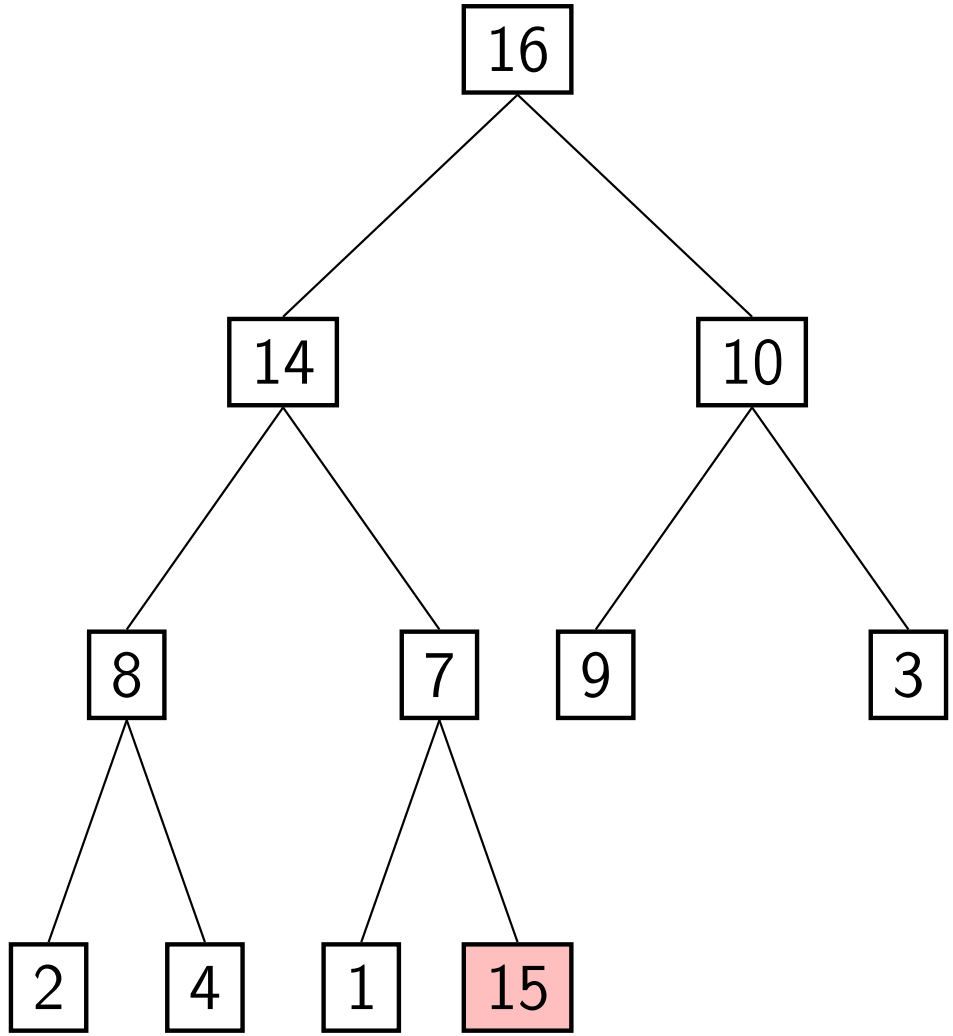
Algorithme 4 Insérer un élément dans un tas

```
Insérer(T : tableau, p :entier)
▷ Entrée : T (un tas binaire)
▷ Sortie : T est un tas binaire contenant l'élément de priorité p
▷ Variables locales :
    i : entier ;
Début
    Taille(T) ← Taille(T) + 1 ;
    i ← Taille(T) ;
     tant que (i > 1 et T[pere(i)] < p)  faire
        T[i] ← T[Pere(i)] ;
        i ← Pere(i) ;
    T[i] ← p ;
Fin
```

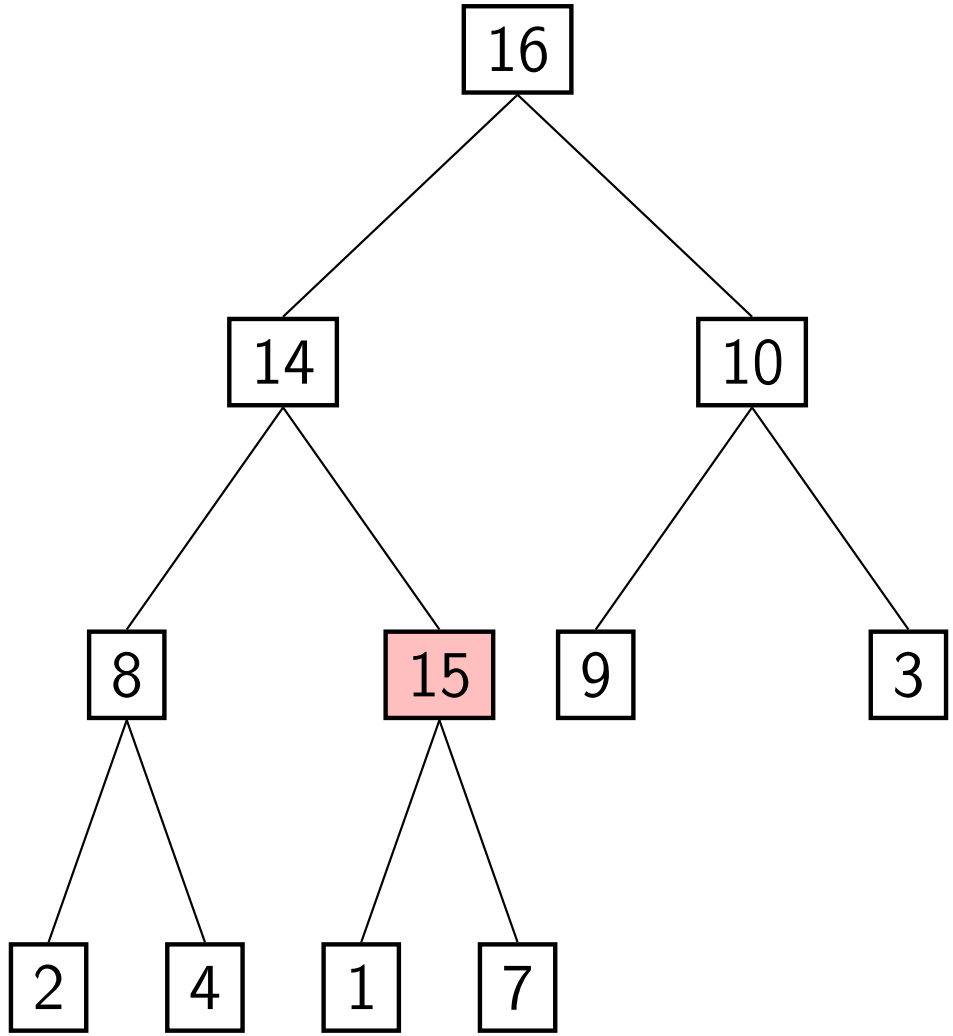
Exemple d'insertion : insertion du nœud 15



Exemple d'insertion : insertion du nœud 15



Exemple d'insertion : insertion du nœud 15



Exemple d'insertion : insertion du nœud 15