Chapitre 2 : types de données abstraits

Plan

- 1/ Définitions
- 2/ Tableaux
- 3/ Listes
  - 3.1/ Piles
  - 3.2/ Files
- 4/ Arbres
  - 4.1/ Tas
- 5/ Graphes

1/ Définitions

- Type de donnée abstrait (TDA) = modélisation logique d'une structure de données, décrivant les opérations possibles (ex : ajouter, supprimer, modifier)
  - Un type de donnée abstrait peut être implémenté de plusieurs manières
  - On associe souvent aux opérations une complexité donnée (mais cette complexité ne peut être atteinte que par de bonnes implémentations)
- Les types de données abstraits permettent d'abstraire l'algorithme

2/ Tableaux

- Définition : un tableau *(array)* est un TDA permettant de stocker un nombre limité d'éléments, et d'accéder en temps constant à chaque élément à partir de son indice
  - Remarque : les éléments sont contigus en mémoire
- Opérations
  - Accès à un élément à partir de son indice : *O(1)*
  - Ajout d'un élément en fin : *O(1)*
  - Insertion d'un élément : *O(n)*
  - Suppression d'un élément : *O(n)*
  - Recherche d'un élément : *O(n)*
- Remarque : on peut concevoir des tableaux circulaires

3/ Listes

- Définition : une liste *(list)* est un TDA permettant de stocker des éléments, dans lequel chaque maillon permet d'accéder à l'élément ou au maillon suivant (avec une indication du dernier maillon)
- Opérations
  - Accès à un élément à partir de son indice : *O(n)*
  - Ajout d'un élément en tête : *O(1)*
  - Ajout d'un élément en fin : *O(n)*
  - Recherche d'un élément : *O(n)*
  - Insertion d'un élément (trouvé) : *O(1)*
  - Suppression d'un élément (trouvé) : *O(1)*
- Remarque : on peut concevoir des listes doublement chaînées, circulaires, etc.

3.1/ Piles

- Définition : une pile *(stack)* est un TDA permettant de stocker des éléments, et pour lequel l'ordre de suppression des éléments est l'inverse de l'ordre d'ajout *(LIFO = Last In First Out)*
- Opérations :
  - Empiler : *O(1)*
  - Dépiler : *O(1)*
- Remarques :
  - Si on empile 1 puis 2 puis 3, on dépilera 3 puis 2 puis 1
  - Une pile peut être implémentée avec un tableau ou une liste

3.2/ Files

- Définition : une file *(queue)* est un TDA permettant de stocker des éléments, et pour lequel l'ordre de suppression des éléments est égal à l'ordre d'ajout *(FIFO = First In First Out)*
- Opérations
  - Enfiler : *O(1)*
  - Défiler : *O(1)*
- Remarques :
  - Si on enfile 1 puis 2 puis 3, on défilera 1 puis 2 puis 3
  - Une file peut être implémentée avec un tableau circulaire, une liste simplement chaînée (avec ajout en fin), ou une liste doublement chaînée (avec ajout en tête)

4/ Arbres

- Définition : un arbre *(tree)* est un TDA permettant de stocker des éléments, dans lequel chaque nœud père permet d'accéder à un élément et à plusieurs nœuds fils
- Opérations :
  - Accéder au *x*-ème fils d'un nœud : *O(1)*
  - Parcourir tous les nœuds de l'arbre : *O(n)*
- Remarque : on peut concevoir plusieurs types d'arbres (binaires, ternaires, *n*-aires, etc.) et enregistrer plusieurs types d'informations (dans les nœuds et/ou sur les arcs)

4.1/ Tas

- Définition : un tas *(heap)* est un TDA permettant de stocker des éléments, dans lequel on peut récupérer rapidement le plus petit élément
- Opérations
  - Ajouter un élément : $O(log(n))$
  - Lire le minimum : $O(1)$
  - Enlever le minimum : $O(log(n))$
- Remarque : un tas s'implémente généralement avec un arbre binaire

Algorithmes sur les tas

- Ajouter dans un tas : on ajoute le nouvel élément comme une feuille, puis on le « tasse » à partir de ce nouvel élément
  - Tassement : Pour chaque noeud à partir du dernier :
    - On compare le noeud et son père
    - On les inverse si besoin, et on recommence avec le père
- Supprimer dans un tas : on enlève le premier élément, on le remplace par la dernière feuille, puis on le « tasse » à partir de ce premier élément
  - Tassement : Pour chaque noeud à partir du premier :
    - On compare le noeud et ses fils
    - On les inverse si besoin, et on recommence avec le fils
- Construction du tas : on peut construire un tas en $O(n)$
  - Ajout de tous les éléments dans un tableau non trié, puis tassement de $n/2$ à 1