Programmation impérative

ENSIIE

Semestre 1 — 2020–21

Tri

Appartenance

A priori pas de lien entre indices et valeurs

Tester égalité avec chaque élément

Si tableau de taille N, N comparaisons au pire

Bien préciser retour :

- vrai/faux
- indice où est présent (mais si absent ?)

Recherche dichotomique

Si tableau trié :

- on teste au milieu
- si milieu plus grand, on cherche dans le sous-tableau gauche
- sinon dans le sous-tableau droite

Si tableau de taille N, $\log_2(N)$ comparaisons

Tableau trié :

$$\forall i, j \in [0, N-1]. \quad i \leq j \quad \Rightarrow \quad \texttt{t}[i] \leq \texttt{t}[j]$$

Comment obtenir un tableau trié ?

- Tri

Tris

Plusieurs algorithmes (à bulles, fusion, rapide, ...)

Propriétés différentes

- complexité
- stabilité
- utilisation mémoire (en place ou pas)

Tri par sélection

Idée : mettre le minimum au début

1. Trouver le minimum
2. Échanger avec le début
3. Recommencer avec la suite du tableau

- Complexité ?
- Importance de la configuration initiale ?

Recherche dichotomique

Précondition (`requires`) : tableau trié
Si valeur trop grande à indice courant : chercher avant, sinon après

Bentley, *Programming Pearls* :

- Public : programmeurs pro Bell labs et IBM
- Deux heures
- Langage au choix (voire même pseudo code)

$\Rightarrow$ 90% incorrects (plusieurs années, $>$ 100 personnes)

Knuth, *Sorting and Searching* :
première publication 1946, première version correcte 1962. . .

Algorithme

Problème : tableau `t`, valeur e, $e \in$ `t` ?

- Succès : indice de e
- Échec : -1

Idée : deux bornes `low` et `up`

Invariant : Si $e \in$ `t` alors $indice_e \in [$`low`, `up`$[$

Variant : `low` et `up` se rapprochent strictement

Arrêt : sous-tableau vide, c-à-d `low` $\geq$ `up`

Pile

Pile

Structure de donnée abstraite :

- ensemble d'éléments
- premier arrivé, dernier sorti (LIFO)
- accès à l'élément au sommet en temps constant

Fonctions :

- créer une pile vide
- tester si une pile est vide
- ajouter un élément (empiler)
  - par effet
- retirer le sommet et le récupérer (dépiler)
  - par effet, avec retour valeur

Pour maintenir propriétés : que ces fonctions

Implémentation à l'aide de tableau statique

Taille fixe de tableau

- la pile peut être pleine

Besoins :

- tableau
- étage courant

$\Rightarrow$ `struct`