
Les tableaux

Algorithmique et Programmation en C

Chapitre 5 Karim Bouzoubaa

Chapitre 5 : Les tableaux

- Objectifs

- Comprendre la notion de tableaux et la situer par rapport aux autres types de données
- Être capable de déclarer, initialiser et utiliser un tableau
- Connaître la différence entre un tableau monodimensionnel et multidimensionnel
- Connaître les algorithmes de base de recherche et de tri d'un tableau

- Sommaire

- Introduction
- La déclaration d'un tableau
- Les tableaux simples
- Les tableaux multi-dimensions
- Les méthodes de recherche
- Les méthodes de tri

Introduction

- Supposons une classe de 5 étudiants. Chaque étudiant a une note finale. On voudrait écrire un algo qui affiche toutes les notes supérieures à la moyenne de la classe.

- Algorithme "notes sup à la moyenne"
- Entrées : note1, note2, note3, note4, note5 type entier
- Auxiliaires : i, moy type entier
- Traitement:

Début

Lire note1, note2, note3, note4, note5

moy = (note1 + note2 + note3 + note4 + note5) / 5

si (note1 ≥ moy) Afficher(note1)

si (note2 ≥ moy) Afficher(note2)

si (note3 ≥ moy) Afficher(note3)

si (note4 ≥ moy) Afficher(note4)

si (note5 ≥ moy) Afficher(note5)

Fin

Introduction

- L'algorithme n'est valide que pour une classe de 5 étudiants
- Si on utilise une classe de 100 étudiants, déclarer 100 variables, 100 sélections, etc. !!!



- On associe plutôt un seul nom à l'ensemble des variables des notes d'étudiants (NOTE) et l'emploi d'un indice (i) qui permet de repérer un élément quelconque de l'ensemble

NOTE[0]	NOTE[1]	NOTE[2]	NOTE[3]	NOTE[4]
16	15	12	15	18

Tableau

- Un tableau est un ensemble ordonné qui contient un ensemble fixe d'éléments de même type

NOTE[0]	NOTE[1]	NOTE[2]	NOTE[3]	NOTE[4]
16	15	12	15	18

- NOTE : nom du tableau
- Un tableau de 5 éléments (indices de 0 à 4)
- `NOTE[2] = 12`

Tableau

- Algorithme "notes sup à la moyenne"
- Const: MAX = 5
- Entrées : NOTE tableau de MAX entier
- Auxiliaires : i, som type entier, moy reel
- Traitement:

Début

som = 0

Pour i = 0 à MAX-1 {

 lire(NOTE[i])

 som = som + NOTE[i]

}

moy = som / MAX

Pour i = 0 à MAX-1 {

si (NOTE[i] ≥ moy) Afficher(NOTE[i])

}

Fin

Tableau

- Un algo pour calculer l'écart type de n valeurs $\sqrt{\frac{1}{n} \sum_i (x_i - \bar{x})^2}$

```
• Algorithme "calcule de l'ecart
type"
• Const: MAX = 5
• Entrées : X tableau de MAX entier
• Auxiliares : i, som type entier,
moy, ec reel
• Traitement:
  Début
  som = 0
  Pour i = 0 à MAX-1 {
    lire(X[i])
    som = som + X[i]
  }
  moy = som / MAX
```

```
som = 0
Pour i = 0 à MAX-1 {
  som = som + (X[i] - moy)^2
}
ec = racine(som / MAX)
Afficher ("l'ecart type est:", ec)
Fin
```

Tableau à 2 dimensions

- Algo qui lit et affiche une matrice et la somme de ses éléments

- Algorithmme "matrice"
- Const: MAX1 = 5, MAX2 = 8
- Entrées : MAT tableau de MAX1*MAX2 entier
- Auxilieres : i,j, som type entier
- Traitement:
Début
som = 0
Pour i = 0 à MAX1-1 {
 Pour j = 0 à MAX2-1 {
 lire(MAT[i][j])
 som = som + MAT[i][j]}
 }
}

```
Pour i = 0 à MAX1-1 {  
    Pour j = 0 à MAX2-1 {  
        Afficher (MAT[i][j])  
    }  
}  
Afficher ("la somme est:", som)  
Fin
```

Méthodes de recherche

- Rechercher si un élément existe ou non dans le tableau
 - Recherche linéaire: facile, $O(n)$
 - Recherche dichotomique: $O(\log_n)$, prix: trier le tableau

```
• Algorithmme "recherche lineaire"  
• Const: MAX = 5  
• Entrées : X tableau de MAX entier  
• Auxilieres : i, val type entier  
  Trouve type booleen  
• Traitement:  
  Début  
  Pour i = 0 à MAX-1 {  
    lire(X[i])  
  }  
  lire val
```

```
i = 0  
trouve = faux  
TQ (trouve == faux ET i < MAX) {  
  si (X[i]== val) trouve = vrai  
  sinon i = i + 1  
}  
si (trouve == vrai)  
  Afficher("elt trouve a pos:", i)  
sinon  
  Afficher ("elt non trouve")  
Fin
```

Méthodes de tri

- Tri: opération qui consiste à ordonner les éléments en ordre séquentiel
 - Tri par extraction
 - Tri par échange
 - Tri à bulles

```
• Algorithmme "tri par extraction"
• Const: MAX = 5, GRANDE = 9999
• Entrées : X, Y tableau de MAX
entier
• Auxilieres : i, j, indMin type
entier
• Traitement:
  Début
  Pour i = 0 à MAX-1 {
    lire(X[i])
  }
```

```
  Pour i = 0 à MAX-1 {
    indMin = 0
    Pour j = 1 à MAX-1 {
      si (X[j]<X[indMin])
        indMin = j
    }
    Y[i] = X[indMin]
    X[indMin] = GRANDE
  }
  Pour i = 0 à MAX-1 {
    afficher(Y[i])
  }
  Fin
```

Exercices

1. Algo qui calcule la somme des éléments d'un tableau
2. Algo qui détermine si une matrice carrée est la matrice unité
3. Algo de recherche dichotomique
4. Algo de tri par échange