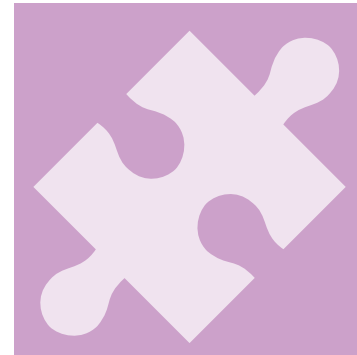


ALGO 2

CHAPITRE 5



Les enregistrements et les fichiers

Achtaich Khadija

Fichier

- Définition

Un FICHER (anglais : file) est un REGROUPEMENT LOGIQUE DE DONNEES MEMORISEES SUR UN SUPPORT PERMANENT (disque dur, par exemple) afin de permettre une réutilisation ultérieure des informations qu'il contient.

Enregistrement

Définition

- Un enregistrement est un bloc de données élémentaires qui décrit une entité.
- Un enregistrement (*anglais : record*) est composé de plusieurs champs.
- Un champ est l'une des données élémentaires d'un enregistrement

Exemple:

Dans un fichier « Clients », un enregistrement correspond aux données relatives à un client.

le numéro de client, le nom, le prénom, l'adresse..., sont des champs d'un enregistrement du fichier client.

Achtaich Khadija



Organisation des données dans un fichier

- L'organisation des données dans un fichier détermine comment seront placés chacun des enregistrements
- On cite trois types d'organisations:
 - **Organisation séquentielle**
 - **Organisation calculée**
 - **Organisation indexée**

Achtaich Khadija



Organisation séquentielle

- Dans des fichiers séquentiels, les enregistrements sont mémorisés consécutivement dans l'ordre de leur entrée et peuvent seulement être lus dans cet ordre.
- Si on a besoin d'un enregistrement précis dans un fichier séquentiel, il faut lire tous les enregistrements qui le précèdent, en commençant par le premier.

Achtaich Khadija



Organisation calculée

Dans des fichiers à placement calculé, les enregistrements sont placés à une position précise du fichier. Cette position est:

- Soit donnée comme numéro d'ordre de l'enregistrement dans le fichier,
- Soit calculée selon un algorithme de placement appliqué à une clef. (La clef correspond à l'un des champs de l'enregistrement permettant l'identification d'un enregistrement unique au sein du fichier)

Achtaich Khadija



Organisation indexée

Dans des fichiers à organisation indexée, au moins 2 fichiers sont nécessaires pour chaque fichier géré :

- le premier fichier contient les données
- le second fichier contient une table d'index qui à une valeur d'une clef d'un enregistrement conserve la position du fichier à laquelle il se trouve.

Achtaich Khadija



Modes d'accès aux enregistrements d'un fichier

- L'accès aux enregistrements d'un fichier est déterminé par l'organisation de ce fichier.
- On distingue deux modes d'accès:
 - **Accès séquentiel**
 - **Accès direct**

Achtaich Khadija



Accès séquentiel

- L'accès séquentiel consiste à parcourir, dans l'ordre dans lequel ils sont stockés, les enregistrements d'un fichier, sans retour arrière possible.
- Pour accéder à un enregistrement, il faut avoir lu tous les enregistrements qui le précèdent.

Achtaich Khadija



Accès direct

L'accès direct permet l'accès individuel à chacun des enregistrements d'un fichier, en y accédant directement :

- Soit grâce à un numéro d'ordre de placement ;
- Soit grâce à une clef.

Achtaich Khadija



Algorithmique – fichiers séquentiels

Le langage algorithmique (*et la plupart des langages de programmation*) met à la disposition du programmeur un ensemble d'instructions permettant la manipulation des fichiers.

Achtaich Khadija



Algorithmique – fichiers séquentiels

➤ Déclaration d'une variable de type FICHIER

VAR fichier : FICHIER

➤ Lire un fichier

LIRE_FICHIER(fichier, liste_des_variables)

➤ Tester la fin du fichier

FF(fichier)

retourne VRAI si la fin du fichier a été atteinte (il n'y a plus d'enregistrements à lire)

Achtaich Khadija



Algorithmique – fichiers séquentiels

➤ Ouvrir un fichier séquentiel

fichier ← OUVRIIR(nom_du_fichier ,mode_d'Ouverture)

Les modes possibles sont: LECTURE, ECRITURE,AJOUT
LECTURE/ECRITURE

Mode d'ouverture	Actions autorisée
Lecture	En mode lecture, seules seront autorisés l'accès aux données, sans modification possible du contenu
Ecriture	En mode écriture, le fichier est vidé de son contenu, et ne sera possible que l'écriture de nouveaux enregistrements
Ajout	En mode ajout, les données présentes dans le fichier seront préservées, et il sera possible d'ajouter de nouveaux enregistrements
Lecture/écriture	En mode lecture/écriture, les données présentes dans le fichier seront préservées, et il sera possible de modifier le contenu de certains enregistrements.

Achtaich Khadija



Algorithmique – fichiers séquentiels

Ecrire dans un fichier

ECRIRE(fichier, liste de variables)

Fermer un fichier

FERMER(fichier)

Achtaich Khadija



Algorithmique – fichiers séquentiels

Exemple:

VARIABLES

f1, f2 : **FICHER**

num, nom, prenom, email : CHAINE

DEBUT

f1 ← **OUVRIR**("fichier1.txt" , **LECTURE**)

f2 ← **OUVRIR**("fichier2.txt" , **ECRITURE**)

TANTQUE (NON **FF**(f1))

LIRE_FICHER(f1, num, nom, prenom, email)

ECRIRE_FICHER(f2, num, nom, prenom, email)

FINTANTQUE

FERMER(f2)

FERMER(f1)

FIN

Achtaich Khadija



Algorithmique – fichiers séquentiels

Exercice:

On souhaite mémoriser des noms des personnes dans un fichier nommé « personne.txt », créer les sous-programmes qui suivent :

- Une procédure de création du fichier qui contient les noms des personnes.
- Une procédure d'affichage des noms de personnes.
- Une fonction qui permet de chercher un nom passe en argument et qui renvoie vrai si ce dernier est existant et faux sinon.

Ecrire un algorithme principal faisant appel aux différents sous-programmes.

Achtaich Khadija



Algorithmique – fichiers séquentiels

Procedure Creation(fn : Fichier)

Var n : chaine , rep : caractere

Debut

fn ← Ouvrir(personne.txt, ECRITURE)

Rep ← 'O'

Tant que (rep = 'O') **Faire**

Ecrire("entrer un Nom : "),

Lire(n)

Ecrire_Fichier(fn,n)

Ecrire("Voulezvous ajouter un autre nom (O/N) : ")

Lire(rep)

Fin Tant que

Fermer(fn)

FinProcedure

Achtaich Khadija



Algorithmique – fichiers séquentiels

```
Procedure Affichage(fn : Fichier)
Var n : chaîne
Debut
  fn    Ouvrir(personne.txt,LECTURE)
Tant que NON(FF(fn)) Faire
  Lire_Fichier(fn,n)
  Ecrire(n)
Fin Tant que
Fermer(fn)
FinProcedure
```

Achtaich Khadija



Algorithmique – fichiers séquentiels

Fonction Recherche(x : chaine ; fn : Fichier) : Booleen

Var n : chaine, Trouve : Booleen

Debut

 fn ← Ouvrir(personne.txt, LECTURE)

Tant que (Trouve=faux) **ET** (NON(FF(fn))) **Faire**

 Lire(fn, n)

 Trouve ← (n = x)

Fin Tant que

Si (FF(fn)) **Alors**

 Retourne faux

Sinon

 Retourne vrai

Fin Si

Fermer(fn)

FinFonction

Achtaich Khadija



Algorithmique – fichiers séquentiels

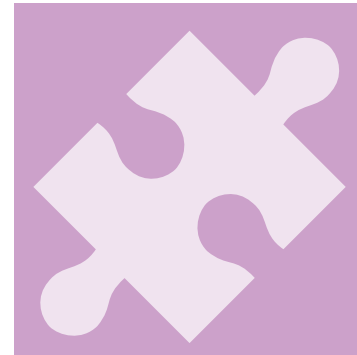
```
Algorithme personne  
Var F1:Fichier  
Debut  
  Creation(F1)  
  Affichage(F1)  
  Si Recherche("Riadh",F1) Alors  
    Ecrire("Riadh est existant dans le fichier")  
  Sinon  
    Ecrire("Riadh est non existant dans le fichier")  
  Fin Si  
Fin
```

Achtaich Khadija



ALGO 2

CHAPITRE 6



La complexité

Achtaich Khadija

Introduction à la complexité

- Il existe souvent plusieurs façons de programmer un algorithme.
- le choix de la solution s'impose lorsque le nombre d'opérations et la taille des données d'entrée sont importants.
- Deux paramètres sont déterminants : le temps d'exécution et l'occupation mémoire.

Achtaich Khadija



Introduction à la complexité

La notion de complexité décrit le temps et la mémoire nécessaires pour exécuter un algorithme et permet donc de caractériser son efficacité.

Achtaich Khadija



Types de complexités

Il existe deux types de complexités: **complexité temporelle** et **complexité spatiale**.

Définition 1: la complexité temporelle d'un algorithme est le temps mis par ce dernier pour transformer les données du problème considéré en un ensemble de résultats.

Définition 2 : la complexité spatiale d'un algorithme est l'espace mémoire utilisé par ce dernier pour transformer les données du problème considéré en un ensemble de résultats.

Achtaich Khadija



Types de complexités

Remarque: Le coût de la mémoire étant aujourd'hui relativement faible, on cherche en général à améliorer la complexité en temps plutôt que la complexité en mémoire.

Dans ce qui suit, nous allons nous concentrer beaucoup plus sur le temps d'exécution

Achtaich Khadija



Exemple: Somme des n premiers entiers

➤ Calcul à l'aide d'une boucle :

Entrées: entier naturel n

Sorties: entier naturel somme

Somme \leftarrow 0

i \leftarrow 1

tant que (i \leq n) faire

Somme \leftarrow somme+i

i \leftarrow i+1

retourner somme

Coût(A1) : 2n additions

Achtaich Khadija



Exemple: Somme des n premiers entiers

➤ Calcul à l'aide de la formule mathématique:

Entrées: entier naturel n

Sorties: entier naturel somme

Somme $\leftarrow n+1$

Somme $\leftarrow \text{somme} * n$

Somme $\leftarrow \text{somme} / 2$

retourner somme

Coût(A2) : 1 addition + 1 multiplication + 1 division

Achtaich Khadija



Exemple: Somme des n premiers entiers

Comparaison du coût des deux algorithmes en terme d'opérations arithmétiques :

- $\text{Coût}(A1) : 2n$ opérations arithmétiques.
- $\text{Coût}(A2) : 3$ opérations arithmétiques.

Pour $n > 1$, $\text{Coût}(A2) < \text{Coût}(A1)$.

L'algorithme A2 est plus efficace que l'algorithme A1.

Achtaich Khadija



Evaluation de la complexité d'un Algorithme

- ❖ Le temps d'exécution dépend de la longueur de l'entrée.
- ❖ Ce temps est une fonction $T(n)$ où n est la longueur des données d'entrée.

Achtaich Khadija



Evaluation de la complexité d'un Algorithme

- ❖ Pour l'évaluer, il faut choisir une mesure élémentaire :
 - nombre de comparaisons,
 - nombre d'affectations,
 - nombre d'opérations arithmétiques,
 - ...
- ❖ La complexité algorithmique s'exprime en fonction de la taille n des entrées.
- ❖ **Il n'y a pas un ensemble de règles standardisé pour évaluer la complexité d'un algorithme.**

Achtaich Khadija



Quelques Règles d'évaluation de la complexité d'un Algorithme

➤ une séquence d'instructions $x_1; x_2; \dots; x_n$

$$\text{Coût}(x_1; x_2; \dots; x_n) = \sum \text{Coût}(x_k)$$

Exemple

somme \leftarrow n+1

Somme \leftarrow somme*n

Somme \leftarrow somme/2

$$\text{Coût}(A2) = \text{coût}(X1) + \text{coût}(X2) + \text{coût}(X3) = 3$$

Achtaich Khadija



Quelques Règles d'évaluation de la complexité d'un Algorithme

➤ La boucle simple : Tant que $(i < n)$ faire x_i

$$\text{Coût(boucle)} = \sum_i (\text{Coût(comparaison)} + \text{Coût}(x_i))$$

Exemple

tant que $(i \leq n)$ faire

 somme \leftarrow somme + i

 i \leftarrow i + 1

fin tant que

$$\text{Coût}(A) = \text{Coût}(i \leq n) + \text{Coût}(\text{somme} + i) + \text{Coût}(i + 1) = 3n$$

Achtaich Khadija



Quelques Règles d'évaluation de la complexité d'un Algorithme

➤ Les instructions conditionnelles :

Si condition faire X_{vrai} sinon faire x_{faux}

$\text{Coût}(\text{conditionnelle}) \leq \text{Coût}(\text{test}) + \text{Sup}(\text{Coût}(x_{\text{vrai}}); \text{Coût}(x_{\text{faux}}))$

Exemple :

si $(i/2=0)$ **alors** $n \leftarrow i/2$

sinon

$i \leftarrow i+1$

$n \leftarrow i/2$

$\text{Coût}(A) \leq \text{Coût}(i/2 = 0) + \text{Sup}(\text{Coût}(n \leftarrow i/2); \text{Coût}(i \leftarrow i + 1; n \leftarrow i/2))$

$\text{Coût}(A) \leq 3$

Achtaich Khadija



Types de complexités

Soit:

- d : une entrée de taille n ,
- D_n : l'ensemble des entrées d possibles,
- Coût $A(d)$: le coût de l'algorithme A pour l'entrée d .

On peut alors calculer trois complexités différentes :

- la complexité dans le pire des cas,
- la complexité dans le meilleur des cas,
- la complexité moyenne.

Achtaich Khadija



Types de complexités

Complexité dans le pire des cas:

$$\text{Max}_A = \text{Sup}(\text{Coût } A(d) / d \in D_n)$$

➤ C'est cette complexité qu'on utilise le plus souvent pour comparer deux algorithmes.

Achtaich Khadija



Types de complexités

Complexité dans le meilleur des cas:

$$\text{Min}_A = \text{Inf} (\text{Coût } A(d) / d \in D_n)$$

Achtaich Khadija



Types de complexités

Complexité moyenne:

$$\text{Moy}_A = \sum_{d \in D_n} p(d) \text{Coût } A(d)$$

- La plus satisfaisante mais très difficile à calculer par méconnaissance de la distribution de probabilité suivi par les entrées.

Achtaich Khadija



Types de complexités

Exemple:

```
fonction rechercheElement(tab:tableau entier, x:entier): booléen
entier i;
début
    i <- 0
    tantque (i < n)
        si (tab(i) = x) alors
            retourne VRAI
        finsi
    fintantque
    retourne FAUX
finfonction
```

Achtaich Khadija



Types de complexités

n = la taille du tableau, a = affectation d'entier, c = comparaison d'entier.

Complexité au pire (x n'est pas dans le tableau) : $a + n * (2 * c)$

Complexité au mieux (x dans la 1^{ère} case du tableau): $a + 2 * c$

Complexité en moyenne : considérons qu'on a 50% de chance que x soit dans le tableau, et 50% qu'il n'y soit pas, et, s'il y est sa position moyenne est au milieu.

Le temps d'exécution est $[(a + n * (2 * c)) + (a + (n/2) * (2 * c))] / 2$

Achtaich Khadija



Types de complexités

Remarque:

Il est clair que pour certains algorithmes, il n'y a pas lieu de distinguer entre ces trois mesures de complexité.

Achtaich Khadija



Complexité asymptotique: Introduction

- Les algorithmes s'exécutent souvent sur des entrées de grande taille n .
- le temps précis d'exécution d'un programme n'est pas intéressant, mais **l'ordre de grandeur de ce temps en fonction de la taille des données** ;
- une approximation asymptotique de la complexité est suffisante.

Achtaich Khadija



Complexité asymptotique: Définition

Définition:

La complexité asymptotique d'un algorithme décrit le comportement de celui-ci quand la taille n des données du problème traité devient de plus en plus grande, plutôt qu'une mesure exacte du temps d'exécution.

Achtaich Khadija



Complexité asymptotique: Notation O

Définition:

Soient f et $g : \mathbb{N} \rightarrow \mathbb{R}^+ : f = O(g)$ ssi $\exists c \in \mathbb{R}^+ ; \exists n_0 \in \mathbb{N}$ tels que:

$$\forall n > n_0; f(n) \leq c \times g(n)$$

Utilité:

Le temps d'exécution est borne

Signification:

Pour toutes les grandes entrées (i.e., $n \geq n_0$), on est assuré que l'algorithme ne prend pas plus de $c \times g(n)$ étapes. (borne supérieur)

Achtaich Khadija



Complexité asymptotique: Notation O

Les algorithmes s'exécutent souvent sur des entrées de grande taille n .

Conséquences :

- le coût exacte d'un algorithme n'est pas intéressant,
- une approximation asymptotique de la complexité est suffisante.

Achtaich Khadija



Complexité asymptotique: Notation O

Echelle de fonction de complexité :

	Nom de la complexité
$O(1)$	constante
$O(\log(n))$	logarithmique
$O(n)$	linéaire
$O(n^2)$	quadratique
$O(n^k)$	polynomiale
$O(2^n)$	exponentielle

Achtaich Khadija



Notation O: Règles de simplification

Si

$$f(n) = O(g(n))$$

et

$$g(n) = O(h(n)),$$

alors

$$f(n) = O(h(n)).$$

Achtaich Khadija



Notation O: Règles de simplification

Si

$$f(n) = O(kg(n))$$

où $k > 0$ est une constante,

alors

$$f(n) = O(g(n)).$$

Achtaich Khadija



Notation O: Règles de simplification

Si

$$f_1(n) = O(g_1(n))$$

et

$$f_2(n) = O(g_2(n))$$

alors

$$f_1(n)f_2(n) = O(g_1(n) g_2(n))$$

Achtaich Khadija



Complexité asymptotique: Exemples

Exemple 1:

$T(n) = c$. On écrit $T(n) = O(1)$.

Exemple 2:

$$T(n) = c_1 n^2 + c_2 n.$$

$$c_1 n^2 + c_2 n \leq c_1 n^2 + c_2 n^2 = (c_1 + c_2) n^2$$

pour tout $n \geq 1$.

$$T(n) \leq cn^2 \text{ où } c = c_1 + c_2 \text{ et } n_0 = 1.$$

Donc, $T(n)$ est en $O(n^2)$

Achtaich Khadija



Complexité asymptotique: Exemples

Exemple 3:

Initialiser un tableau d'entiers

pour $i \leftarrow 0$ à $n-1$

$Tab(i) \leftarrow 0;$

Finpour

Il y a n itérations

Chaque itération nécessite un temps $\leq c$,
où c est une constante (comparaison + une affectation).

Le temps est donc $T(n) \leq cn$

Donc **$T(n) = O(n)$**

Achtaich Khadija



Complexité asymptotique: Exemples

Exemple 4:

$a \leftarrow b;$

Temps constant: $O(1)$.

Exemple 5:

Somme $\leftarrow 0$

Pour $j \leftarrow 1$ à n

 pour $i \leftarrow 1$ à n

 somme \leftarrow somme+1

 Finpour

Pour $k \leftarrow 0$ à $n-1$

$A[k] = k$

 Finpour

Temps: $O(1) + O(n^2) + O(n) = O(n^2)$

Achtaich Khadija



Complexité asymptotique: Exemples

Exemple 6:

```
somme ← 0
Pour i ← 1 à n
  j ← 1
  tant que (j ≤ i)
    somme ← somme + 1
    j ← j + 1
  Fin tant que
Fin pour
```

Temps: $O(1) + O(n^2) = O(n^2)$

Achtaich Khadija

Complexité asymptotique: Exemples

Exemple 7:

```
somme ← 0
k ← 1
tant que (k ≤ n)
  Pour j ← 1 à n
    somme ← somme + 1
  Finpour
  k ← k * 2
Fin tantque
```

Temps: $O(n \log_2 n)$

Achtaich Khadija



Complexité asymptotique: Exemples

Exemple 8: cas récursif

Fonction factorielle(n : entier) : entier

Début

Si $n = 0$ ou $n = 1$

Retourne 1

Sinon

Retourne $n * \text{fact}(n-1)$

FinSi

Finfonction

Achtaich Khadija



Complexité asymptotique: Exemples

Si $n = 1$, $\text{com}(n) = c$ donc **la complexité est d'ordre $O(1)$**

On pose une équation de récurrence : appelons $\text{com}(n)$ la complexité

$$\begin{cases} \text{com}(n) = c + \text{com}(n-1) + o & \text{si } n \neq 1 \\ \text{com}(1) = c \end{cases}$$

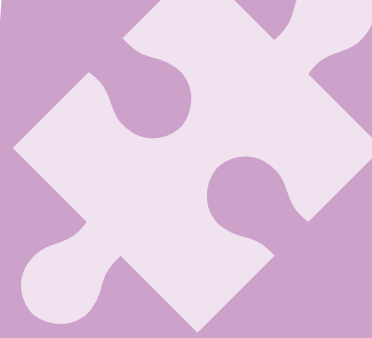
On résoud cette équation de récurrence :

$$\text{com}(n) = n * c + (n-1) * o = O(n)$$

Temps: $O(n)$

Achtaich Khadija





GOOD



LUCK

Achtaich Khadija