



La Recherche Dichotomique (Binary Search)

Joseph AZAR

IUT Nord Franche-Comté
joseph.azar@univ-fcomte.fr

💡 Astuce : Appuyez sur **0** ou **Échap** pour la vue d'ensemble



Le problème du dictionnaire



Comment cherchez-vous un mot dans un dictionnaire ?



Le problème du dictionnaire



Comment cherchez-vous un mot dans un dictionnaire ?

Méthode 1 : Page par page (recherche linéaire)

- Commencer à la page 1
- Lire chaque mot jusqu'à trouver le bon
- 🕒 Très long ! (des milliers de pages)



Le problème du dictionnaire



Comment cherchez-vous un mot dans un dictionnaire ?

Méthode 1 : Page par page (recherche linéaire)

- Commencer à la page 1
- Lire chaque mot jusqu'à trouver le bon
- 🕒 Très long ! (des milliers de pages)

Méthode 2 : Diviser pour régner (recherche dichotomique)

Qu'est-ce que la recherche dichotomique ?

Définition : Algorithme de recherche qui divise l'espace de recherche en deux à chaque étape

Qu'est-ce que la recherche dichotomique ?

Définition : Algorithme de recherche qui divise l'espace de recherche en deux à chaque étape

 **Condition préalable OBLIGATOIRE :**

Le tableau **DOIT être trié** (en ordre croissant ou décroissant) !

Qu'est-ce que la recherche dichotomique ?

Définition : Algorithme de recherche qui divise l'espace de recherche en deux à chaque étape

 **Condition préalable OBLIGATOIRE :**

Le tableau **DOIT être trié** (en ordre croissant ou décroissant) !

 **L'idée principale :**

Qu'est-ce que la recherche dichotomique ?

Définition : Algorithme de recherche qui divise l'espace de recherche en deux à chaque étape

 **Condition préalable OBLIGATOIRE :**

Le tableau **DOIT être trié** (en ordre croissant ou décroissant) !

 **L'idée principale :**

- Regarder l'élément du **milieu**

Qu'est-ce que la recherche dichotomique ?

Définition : Algorithme de recherche qui divise l'espace de recherche en deux à chaque étape

 **Condition préalable OBLIGATOIRE :**

Le tableau **DOIT être trié** (en ordre croissant ou décroissant) !

 **L'idée principale :**

- Regarder l'élément du **milieu**
- Si c'est la valeur cherchée → trouvé !

Qu'est-ce que la recherche dichotomique ?

Définition : Algorithme de recherche qui divise l'espace de recherche en deux à chaque étape

 **Condition préalable OBLIGATOIRE :**

Le tableau **DOIT être trié** (en ordre croissant ou décroissant) !

 **L'idée principale :**

- Regarder l'élément du **milieu**
- Si c'est la valeur cherchée → trouvé !
- Si la valeur cherchée est plus petite → chercher à **gauche**

Qu'est-ce que la recherche dichotomique ?

Définition : Algorithme de recherche qui divise l'espace de recherche en deux à chaque étape

 **Condition préalable OBLIGATOIRE :**

Le tableau **DOIT être trié** (en ordre croissant ou décroissant) !

 **L'idée principale :**

- Regarder l'élément du **milieu**
- Si c'est la valeur cherchée → trouvé !
- Si la valeur cherchée est plus petite → chercher à **gauche**
- Si la valeur cherchée est plus grande → chercher à **droite**

Qu'est-ce que la recherche dichotomique ?

Définition : Algorithme de recherche qui divise l'espace de recherche en deux à chaque étape

 **Condition préalable OBLIGATOIRE :**

Le tableau **DOIT être trié** (en ordre croissant ou décroissant) !

 **L'idée principale :**

- Regarder l'élément du **milieu**
- Si c'est la valeur cherchée → trouvé !
- Si la valeur cherchée est plus petite → chercher à **gauche**
- Si la valeur cherchée est plus grande → chercher à **droite**
- Répéter jusqu'à trouver ou épuiser les possibilités



D'abord : La recherche linéaire

(Pour comprendre pourquoi on fait mieux)



La recherche linéaire (séquentielle)

Principe :



La recherche linéaire (séquentielle)

Principe :

1. Parcourir le tableau du début à la fin



La recherche linéaire (séquentielle)

Principe :

1. Parcourir le tableau du début à la fin
2. Comparer chaque élément avec la valeur cherchée



La recherche linéaire (séquentielle)

Principe :

1. Parcourir le tableau du début à la fin
2. Comparer chaque élément avec la valeur cherchée
3. S'arrêter quand on trouve (ou à la fin)



La recherche linéaire (séquentielle)

Principe :

1. Parcourir le tableau du début à la fin
2. Comparer chaque élément avec la valeur cherchée
3. S'arrêter quand on trouve (ou à la fin)

Problème : Si la valeur est à la fin, on parcourt TOUT le tableau !



La recherche linéaire (séquentielle)

Principe :

1. Parcourir le tableau du début à la fin
2. Comparer chaque élément avec la valeur cherchée
3. S'arrêter quand on trouve (ou à la fin)

Problème : Si la valeur est à la fin, on parcourt TOUT le tableau !

```
ALGORITHME RechercheLineaire(arr, N, valeur)
DEBUT
  POUR i DE 0 À N-1 FAIRE
    SI arr[i] = valeur ALORS
      RETOURNER i
    FIN SI
  FIN POUR
  RETOURNER -1
FIN
```

Complexité : $O(n)$



Recherche linéaire : Animation

Chercher la valeur : 42

5	12	17	23	38	42	56	68	75	89
0	1	2	3	4	5	6	7	8	9

Cliquez sur "Étape suivante" pour commencer



La recherche dichotomique

(L'algorithme optimal pour les tableaux triés)

Le principe pas à pas

Le principe pas à pas

Variables nécessaires :

- **gauche** : index de début de la zone de recherche
- **droite** : index de fin de la zone de recherche
- **milieu** : index du milieu = $(\text{gauche} + \text{droite}) / 2$

Le principe pas à pas

Variables nécessaires :

- **gauche** : index de début de la zone de recherche
- **droite** : index de fin de la zone de recherche
- **milieu** : index du milieu = $(\text{gauche} + \text{droite}) / 2$

Étapes de l'algorithme :

Le principe pas à pas

Variables nécessaires :

- **gauche** : index de début de la zone de recherche
- **droite** : index de fin de la zone de recherche
- **milieu** : index du milieu = $(\text{gauche} + \text{droite}) / 2$

Étapes de l'algorithme :

1. **1** Initialiser : gauche = 0, droite = N-1

Le principe pas à pas

Variables nécessaires :

- **gauche** : index de début de la zone de recherche
- **droite** : index de fin de la zone de recherche
- **milieu** : index du milieu = $(\text{gauche} + \text{droite}) / 2$

Étapes de l'algorithme :

1. Initialiser : gauche = 0, droite = N-1
2. Calculer milieu = $(\text{gauche} + \text{droite}) / 2$

Le principe pas à pas

Variables nécessaires :

- **gauche** : index de début de la zone de recherche
- **droite** : index de fin de la zone de recherche
- **milieu** : index du milieu = $(\text{gauche} + \text{droite}) / 2$

Étapes de l'algorithme :

1. Initialiser : gauche = 0, droite = N-1
2. Calculer milieu = $(\text{gauche} + \text{droite}) / 2$
3. Comparer arr[milieu] avec la valeur cherchée :
 - Si égal → **trouvé !**
 - Si valeur < arr[milieu] → droite = milieu - 1
 - Si valeur > arr[milieu] → gauche = milieu + 1

Le principe pas à pas

Variables nécessaires :

- **gauche** : index de début de la zone de recherche
- **droite** : index de fin de la zone de recherche
- **milieu** : index du milieu = $(\text{gauche} + \text{droite}) / 2$

Étapes de l'algorithme :

1. Initialiser : gauche = 0, droite = N-1
2. Calculer milieu = $(\text{gauche} + \text{droite}) / 2$
3. Comparer arr[milieu] avec la valeur cherchée :
 - Si égal → **trouvé !**
 - Si valeur < arr[milieu] → droite = milieu - 1
 - Si valeur > arr[milieu] → gauche = milieu + 1
4. Répéter tant que gauche ≤ droite



Recherche dichotomique : Animation

Chercher la valeur : 42

5	12	17	23	38	42	56	68	75	89
0	1	2	3	4	5	6	7	8	9

`gauche` = 0 | `droite` = 9 | `milieu` = ?

Cliquez sur "Étape suivante" pour commencer



Algorithme : Recherche dichotomique

```
ALGORITHME RechercheDichotomique(arr, N, valeur)
```

```
DEBUT
```

```
    gauche ← 0
```

```
    droite ← N - 1
```

```
    TANT QUE gauche ≤ droite FAIRE
```

```
        milieu ← (gauche + droite) / 2
```

```
        SI arr[milieu] = valeur ALORS
```

```
            RETOURNER milieu
```

```
        SINON SI arr[milieu] > valeur ALORS
```

```
            droite ← milieu - 1
```

```
        SINON
```

```
            gauche ← milieu + 1
```

```
        FIN SI
```

```
    FIN TANT QUE
```

```
    RETOURNER -1
```

```
FIN
```



Algorithme : Recherche dichotomique


```
ALGORITHME RechercheDichotomique(arr, N, valeur)
DEBUT
    gauche ← 0
    droite ← N - 1

    TANT QUE gauche ≤ droite FAIRE
        milieu ← (gauche + droite) / 2


        SI arr[milieu] = valeur ALORS
            RETOURNER milieu
        SINON SI arr[milieu] > valeur ALORS
            droite ← milieu - 1
        SINON
            gauche ← milieu + 1
        FIN SI
    FIN TANT QUE

    RETOURNER -1
FIN
```


Complexité : $O(\log n)$ 🎉

 **Traçage : Chercher 42 dans [5, 12, 17, 23, 38, 42, 56, 68, 75, 89]**

Itération	gauche	droite	milieu	arr[milieu]	Comparaison	Action


 **Traçage : Chercher 42 dans [5, 12, 17, 23, 38, 42, 56, 68, 75, 89]**

Itération	gauche	droite	milieu	arr[milieu]	Comparaison	Action
1	0	9	4	38	$42 > 38$	gauche = 5


 **Traçage : Chercher 42 dans [5, 12, 17, 23, 38, 42, 56, 68, 75, 89]**

Itération	gauche	droite	milieu	arr[milieu]	Comparaison	Action
1	0	9	4	38	$42 > 38$	gauche = 5
2	5	9	7	68	$42 < 68$	droite = 6

Traçage : Chercher 42 dans [5, 12, 17, 23, 38, 42, 56, 68, 75, 89]

Itération	gauche	droite	milieu	arr[milieu]	Comparaison	Action
1	0	9	4	38	$42 > 38$	gauche = 5
2	5	9	7	68	$42 < 68$	droite = 6
3	5	6	5	42	$42 = 42$	Trouvé ! 

Traçage : Chercher 42 dans [5, 12, 17, 23, 38, 42, 56, 68, 75, 89]

Itération	gauche	droite	milieu	arr[milieu]	Comparaison	Action
1	0	9	4	38	$42 > 38$	gauche = 5
2	5	9	7	68	$42 < 68$	droite = 6
3	5	6	5	42	$42 = 42$	Trouvé ! 

Seulement 3 comparaisons au lieu de 6 avec la recherche linéaire !

⚡ Pourquoi c'est si rapide ?

Pourquoi c'est si rapide ?

À chaque étape : on élimine la **moitié** des éléments restants !

Pourquoi c'est si rapide ?

À chaque étape : on élimine la **moitié** des éléments restants !

Exemple avec 1000 éléments :

Étape	Éléments restants
Début	1000
1	500
2	250
3	125
4	62
5	31
6	15
7	7
8	3
9	1

Pourquoi c'est si rapide ?

À chaque étape : on élimine la **moitié** des éléments restants !

Exemple avec 1000 éléments :

Étape	Éléments restants
Début	1000
1	500
2	250
3	125
4	62
5	31
6	15
7	7
8	3
9	1

Maximum 10 comparaisons pour 1000 éléments !



Comprendre $O(\log n)$

$\log_2(n)$ = "Combien de fois peut-on diviser n par 2 ?"



Comprendre $O(\log n)$

$\log_2(n)$ = "Combien de fois peut-on diviser n par 2 ?"

Exemples :

- $\log_2(8) = 3$ ($8 \rightarrow 4 \rightarrow 2 \rightarrow 1$)
- $\log_2(16) = 4$
- $\log_2(1024) = 10$
- $\log_2(1\,000\,000) \approx 20$



Comprendre $O(\log n)$

$\log_2(n)$ = "Combien de fois peut-on diviser n par 2 ?"

Exemples :

- $\log_2(8) = 3$ ($8 \rightarrow 4 \rightarrow 2 \rightarrow 1$)
- $\log_2(16) = 4$
- $\log_2(1024) = 10$
- $\log_2(1\,000\,000) \approx 20$

Taille (n)	Linéaire $O(n)$	Dicho $O(\log n)$
10	10	4
100	100	7
1 000	1 000	10
1 000 000	1 000 000	20

Comprendre $O(\log n)$

$\log_2(n)$ = "Combien de fois peut-on diviser n par 2 ?"

Exemples :

- $\log_2(8) = 3$ ($8 \rightarrow 4 \rightarrow 2 \rightarrow 1$)
- $\log_2(16) = 4$
- $\log_2(1024) = 10$
- $\log_2(1\,000\,000) \approx 20$

Taille (n)	Linéaire $O(n)$	Dicho $O(\log n)$
10	10	4
100	100	7
1 000	1 000	10
1 000 000	1 000 000	20

Plus le tableau est grand, plus l'avantage est énorme !



Comparaison des deux méthodes

Critère	Recherche linéaire	Recherche dichotomique



Comparaison des deux méthodes

Critère	Recherche linéaire	Recherche dichotomique
Complexité	$O(n)$	$O(\log n)$



Comparaison des deux méthodes

Critère	Recherche linéaire	Recherche dichotomique
Complexité	$O(n)$	$O(\log n)$
Tableau trié requis	Non	Oui !



Comparaison des deux méthodes

Critère	Recherche linéaire	Recherche dichotomique
Complexité	$O(n)$	$O(\log n)$
Tableau trié requis	Non	Oui !
Facilité d'implémentation	Très simple	Moyenne



Comparaison des deux méthodes

Critère	Recherche linéaire	Recherche dichotomique
Complexité	$O(n)$	$O(\log n)$
Tableau trié requis	Non	Oui !
Facilité d'implémentation	Très simple	Moyenne
Petit tableau ($n < 10$)	OK	OK



Comparaison des deux méthodes

Critère	Recherche linéaire	Recherche dichotomique
Complexité	$O(n)$	$O(\log n)$
Tableau trié requis	Non	Oui !
Facilité d'implémentation	Très simple	Moyenne
Petit tableau ($n < 10$)	OK	OK
Grand tableau ($n > 1000$)	Lent	Rapide !



Démonstration interactive

Entrez une valeur à chercher :

Tableau trié :

3	7	11	15	19	23	27	31	35	39	43	47
51	55	59									

0	1	2	3	4	5	6	7	8	9	10	11
12	13	14									

Entrez une valeur et cliquez sur "Chercher"

Erreurs courantes à éviter

Erreurs courantes à éviter

1 Oublier que le tableau doit être trié

La recherche dichotomique ne fonctionne PAS sur un tableau non trié !

Erreurs courantes à éviter

1 Oublier que le tableau doit être trié

La recherche dichotomique ne fonctionne PAS sur un tableau non trié !

2 Mauvais calcul du milieu

`milieu = (gauche + droite) / 2` — Attention à la division entière !

⚠ Erreurs courantes à éviter

1 Oublier que le tableau doit être trié

La recherche dichotomique ne fonctionne PAS sur un tableau non trié !

2 Mauvais calcul du milieu

$\text{milieu} = (\text{gauche} + \text{droite}) / 2$ — Attention à la division entière !

3 Condition d'arrêt incorrecte

TANT QUE $\text{gauche} \leq \text{droite}$ (et non $\text{gauche} < \text{droite}$)

Sinon on peut rater l'élément quand $\text{gauche} = \text{droite}$!

⚠ Erreurs courantes à éviter

1 Oublier que le tableau doit être trié

La recherche dichotomique ne fonctionne PAS sur un tableau non trié !

2 Mauvais calcul du milieu

$\text{milieu} = (\text{gauche} + \text{droite}) / 2$ — Attention à la division entière !

3 Condition d'arrêt incorrecte

TANT QUE $\text{gauche} \leq \text{droite}$ (et non $\text{gauche} < \text{droite}$)

Sinon on peut rater l'élément quand $\text{gauche} = \text{droite}$!

4 Oublier de mettre à jour gauche/droite

$\text{gauche} = \text{milieu} + 1$ et $\text{droite} = \text{milieu} - 1$ — Sinon boucle infinie !



Applications dans le monde réel



Applications dans le monde réel

-  **Dictionnaires** et encyclopédies






Applications dans le monde réel

-  **Dictionnaires** et encyclopédies
-  **Moteurs de recherche**







Applications dans le monde réel

-  **Dictionnaires** et encyclopédies
-  **Moteurs de recherche**
-  **Bases de données** (index)








Applications dans le monde réel

-  **Dictionnaires** et encyclopédies
-  **Moteurs de recherche**
-  **Bases de données** (index)
-  **Jeux vidéo** (détection de collision)








Applications dans le monde réel

-  **Dictionnaires** et encyclopédies
-  **Moteurs de recherche**
-  **Bases de données** (index)
-  **Jeux vidéo** (détection de collision)
-  **Annuaire téléphonique**









Applications dans le monde réel

-  **Dictionnaires** et encyclopédies
-  **Moteurs de recherche**
-  **Bases de données** (index)
-  **Jeux vidéo** (détection de collision)
-  **Annuaire téléphonique**
-  **Bibliothèques musicales**



Applications dans le monde réel

-  **Dictionnaires** et encyclopédies
-  **Moteurs de recherche**
-  **Bases de données** (index)
-  **Jeux vidéo** (détection de collision)
-  **Annuaire téléphonique**
-  **Bibliothèques musicales**

Fun fact :

Git utilise la recherche dichotomique (`git bisect`) pour trouver quel commit a introduit un bug !

Points clés à retenir

Points clés à retenir

1 Diviser pour régner

À chaque étape, on élimine la moitié des possibilités

Points clés à retenir

1 Diviser pour régner

À chaque étape, on élimine la moitié des possibilités

2 Complexité $O(\log n)$

Extrêmement efficace pour les grands tableaux

Points clés à retenir

1 **Diviser pour régner**

À chaque étape, on élimine la moitié des possibilités

2 **Complexité $O(\log n)$**

Extrêmement efficace pour les grands tableaux

3 **Prérequis : tableau TRIÉ**

Sans tri, l'algorithme ne fonctionne pas !



Points clés à retenir

1 **Diviser pour régner**

À chaque étape, on élimine la moitié des possibilités

2 **Complexité $O(\log n)$**

Extrêmement efficace pour les grands tableaux

3 **Prérequis : tableau TRIÉ**

Sans tri, l'algorithme ne fonctionne pas !

4 **Trois variables clés**

gauche, droite, milieu - à mettre à jour correctement



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1					
2					
3				23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0				
2					
3				23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9			
2					
3				23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4		
2					
3				23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4	16	
2					
3				23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4	16	gauche = 5
2					
3				23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4	16	gauche = 5
2	5				
3				23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4	16	gauche = 5
2	5	9			
3				23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4	16	gauche = 5
2	5	9	7		
3				23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4	16	gauche = 5
2	5	9	7	56	
3				23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4	16	gauche = 5
2	5	9	7	56	droite = 6
3				23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4	16	gauche = 5
2	5	9	7	56	droite = 6
3	5			23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4	16	gauche = 5
2	5	9	7	56	droite = 6
3	5	6		23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4	16	gauche = 5
2	5	9	7	56	droite = 6
3	5	6	5	23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4	16	gauche = 5
2	5	9	7	56	droite = 6
3	5	6	5	23	



Exercice : Tracer l'algorithme

Tableau : [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]

Valeur cherchée : 23

Remplissez le tableau :

Itération	gauche	droite	milieu	arr[milieu]	Action
1	0	9	4	16	gauche = 5
2	5	9	7	56	droite = 6
3	5	6	5	23	Trouvé !



Fin du cours

Questions ?

Joseph AZAR

joseph.azar@univ-fcomte.fr

IUT Nord Franche-Comté



Ressources : [Wikipedia](#) - [Binary Search](#)

