

→ algo du 1^{er} cours : complexité n = bcp trop pour qqch qui est utilisé si souvent.

Dichotomie / Diviser-pour-régner :

Entrée : tableau trié n , min le premier indice, max le dernier, e un élément

Sortie : i tq $A[i] = e$

Algo : si $\text{min} = \text{max}$
 si $\text{tab}[\text{min}] = e$ renvoyer min
 sinon renvoyer NonTrouvé

$\text{mid} \leftarrow (\text{min} + \text{max}) / 2$

si $\text{tab}[\text{mid}] < e$ renvoyer $\text{RechercheDicho}(\text{tab}, \text{mid} + 1, \text{max}, e)$

sinon renvoyer $\text{RechercheDicho}(\text{tab}, \text{min}, \text{mid}, e)$

Complexité : $\log_e(n)$

Version itérative (même complexité)

```

min ← 0
max ← n - 1
tant que min < max
  mid ← (min + max) / 2
  si tab[mid] = e renvoyer mid
  si tab[mid] < e
    min ← mid + 1
  sinon max ← mid
si tab[min] = e renvoyer min
sinon renvoyer NonTrouvé
  
```

Preuve par récurrence sur $m = \beta - \alpha + 1$ taille de l'intervalle

HR \forall tableau trié tab et \forall intervalle $[\alpha, \beta]$, on renvoie la bonne valeur.

Init si $m = 1$ alors $\alpha = \beta$
 Si élément trouvé, on renvoie α
 Sinon NonTrouvé

OK

Hérédité Soit $[\alpha, \beta]$ tq $\beta - \alpha + 1 = m + 1$
 et $\gamma = (\alpha + \beta) / 2$

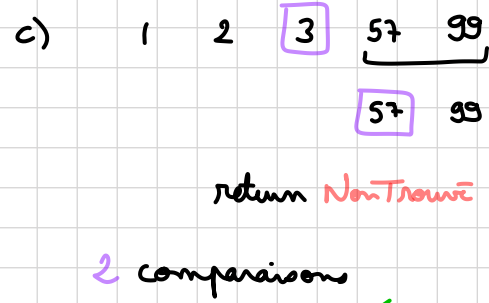
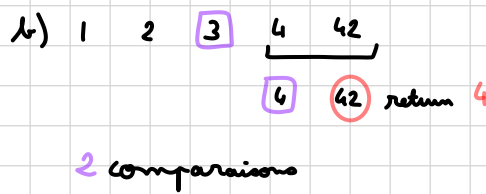
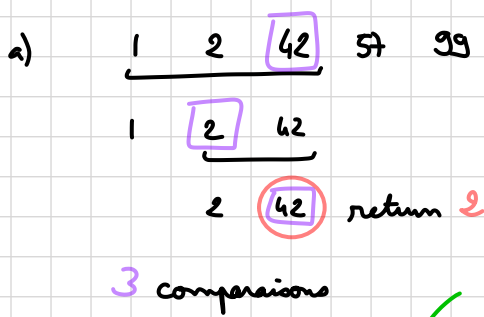
Alors l'exéc⁰ renvoie l'algo évalué :
 soit sur $(\text{tab}, \gamma, \beta, e)$
 soit sur $(\text{tab}, \alpha, \gamma, e)$

Comme $\beta - (\gamma + 1) + 1 \leq m$

EA TD 5

Exercice 1

1) RECURSIF



Exercice 2

1)

```
for (int i = 0; i < n; i++) {  
    if (t[i] == e) return i;  
}  
return -1;
```

Pire des cas = l'élément n'est pas dans le tableau

↳ n comparaisons /

2) En moyenne: $\frac{n}{2}$ comparaisons

↳ l'élément est au milieu du tableau. /

3) a) m recherches linéaires = $m \cdot n$ /

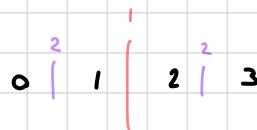
b) k par insert $\theta = n^2$

m recherches dichotomiques = $m \log_2(m)$

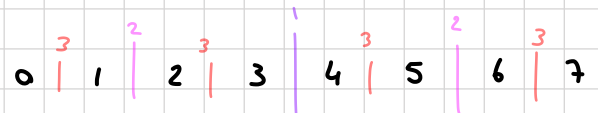
au total $n^2 + m \log_2(m)$ /

Exercice 3

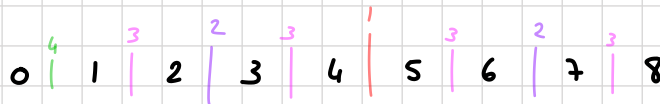
1) $n = 4$ 2 passes



$n = 8$ 3 passes



$n = 9$ 4 passes



2) $n = 2^k$ pièces

PieceFausseDicho (ensemble de n pièces, min le 1^{er} index, max le dernier)

si $\text{max} == \text{min}$ renvoyer min

$\text{mid} \leftarrow (\text{max} + \text{min}) / 2$

si $(\text{pesée}(\text{min}, \text{mid}) < \text{pesée}(\text{mid} + 1, \text{max}))$

retourner PieceFausseDicho(min, mid)

sinon retourner PieceFausseDicho(mid + 1, max)

}

$\log_2(n)$ pesées

3) même chose mais la pesée doit être coefficientée pour prendre en compte le fait que la partie gauche a 1 pièce de plus.

$\log_2(n) + 1$ pesées dans le pire des cas (tjs à gauche)

Exercice 4

1) $\begin{matrix} \text{pré} : & -1 & 0 & 1 \\ \text{post} : & 1 & 3 & 4 & 8 \end{matrix}$

2) $\begin{matrix} \cdot \text{tirés} \\ \cdot \text{plus gd/petit ou égale au pt fixe} \\ \quad = \text{au plus petit/gd indice des éléments.} \end{matrix}$

3) `public static boolean aPtFixe(int [] t) {`

`int min = 0;`

`int max = t.length;`

`while (min < max) {`

`int mid = (min + max) / 2;`

`if (t[mid] <= mid) max = mid;`

`else min = mid + 1;`

`}`

`return t[min] == min;`

`}`

Dans le pire des cas : $\log_2(n)$ comparaisons