INFO-H-100 - Informatique

Séance d'exercices 9
Introduction à Python
Quelques algorithmes

Université Libre de Bruxelles
Faculté des Sciences Appliquées

2011-2012

Boucles `for` imbriquées

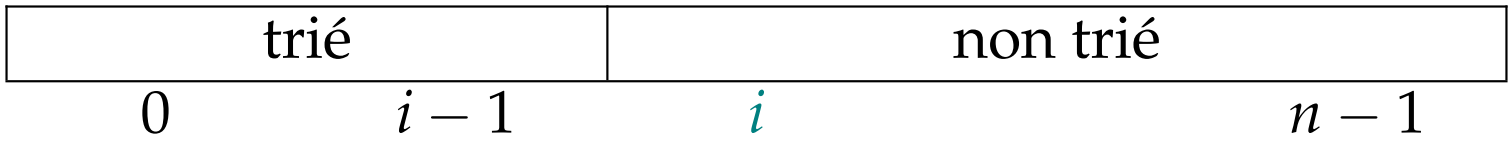
```
for ligne in range(5):
    for cellule in range(3):
        print (ligne, cellule),
    print
```

```
(0, 0) (0, 1) (0, 2)
(1, 0) (1, 1) (1, 2)
(2, 0) (2, 1) (2, 2)
(3, 0) (3, 1) (3, 2)
(4, 0) (4, 1) (4, 2)
```

```
for ligne in range(6):
    for cellule in range(6):
        if cellule<=ligne:
            print "X",
    print
```

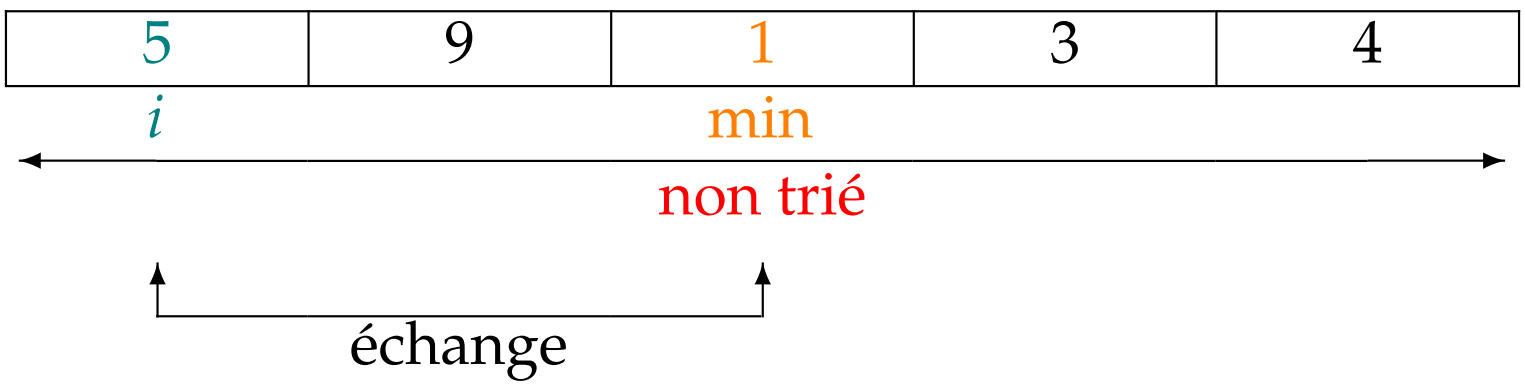
```
X
X X
X X X
X X X X
X X X X X
X X X X X X
```

Tri par sélection (1)



A chaque étape i (de 0 à $n-2$) :

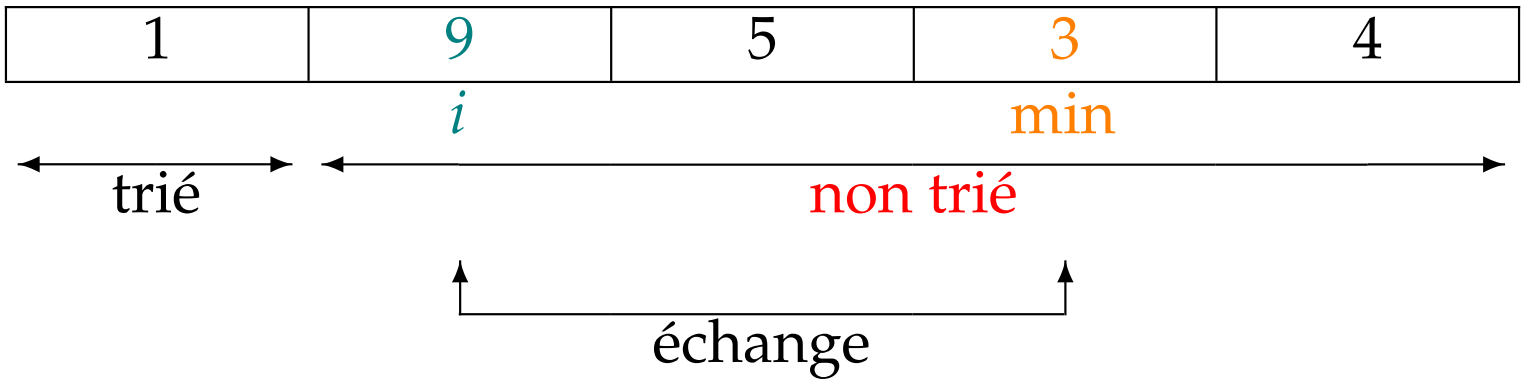
1. chercher le minimum parmi les éléments non triés
2. le placer à sa position définitive via un échange

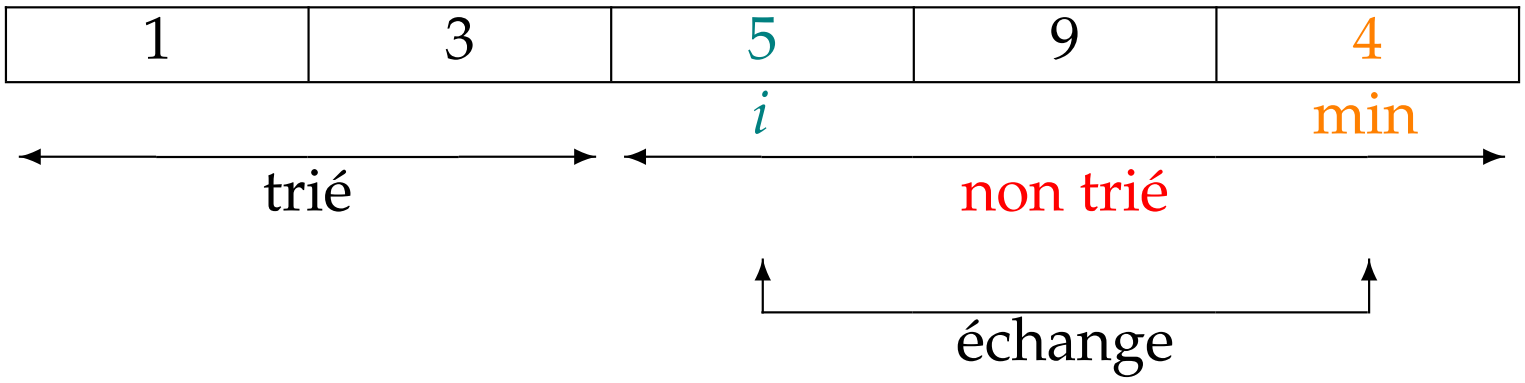


Tri par sélection (2)

A chaque étape i (de 0 à $n-2$) :

1. chercher le minimum parmi les éléments non triés
2. le placer à sa position définitive via un échange

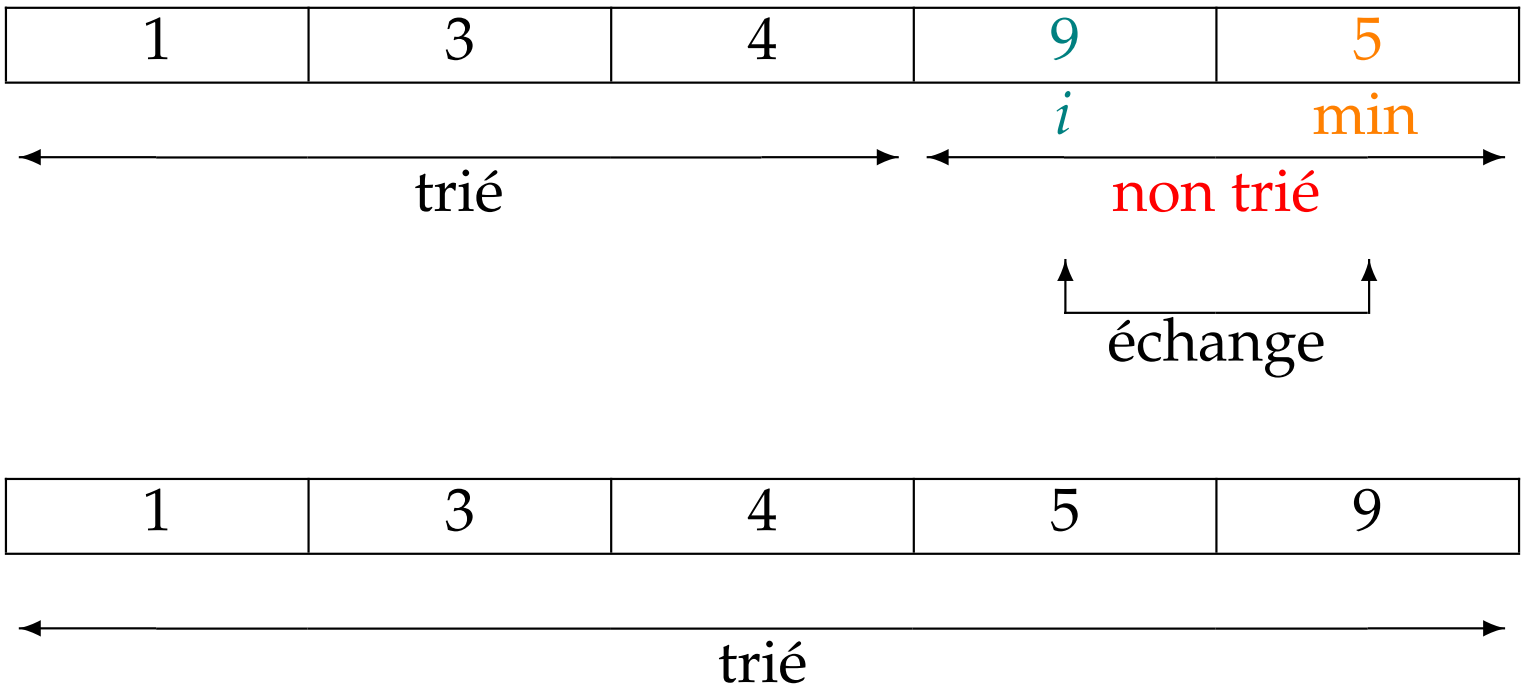




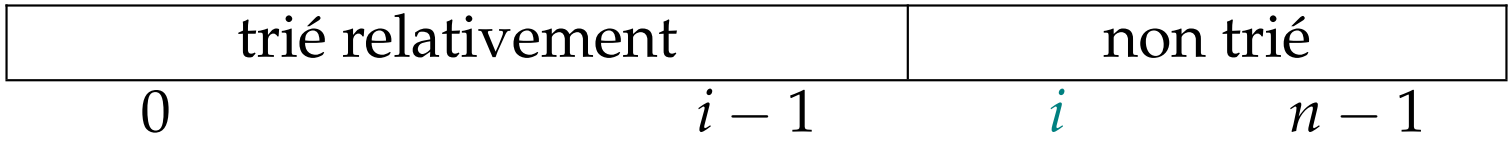
Tri par sélection (3)

A chaque étape i (de 0 à $n-2$) :

1. chercher le minimum parmi les éléments non triés
2. le placer à sa position définitive via un échange

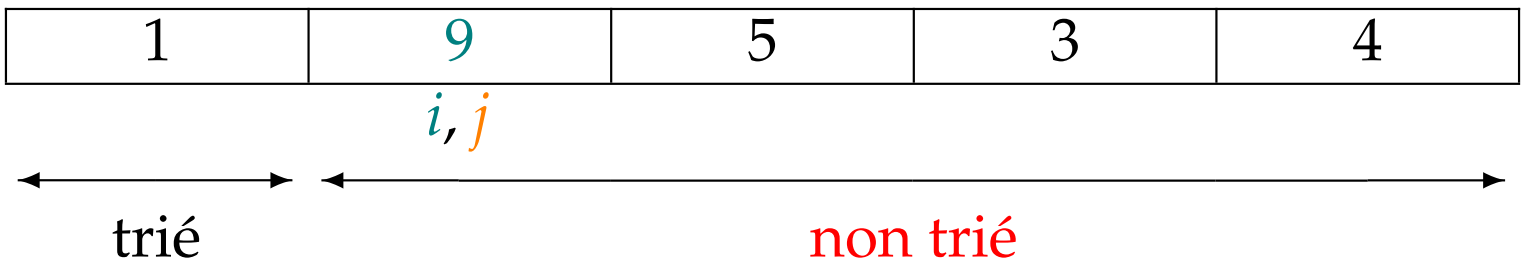


Tri par insertion (1)



A chaque étape i (de 1 à $n-1$) :

1. trouver la place relative j de l'élément i parmi les éléments triés
2. déplacer l'élément i à la place j

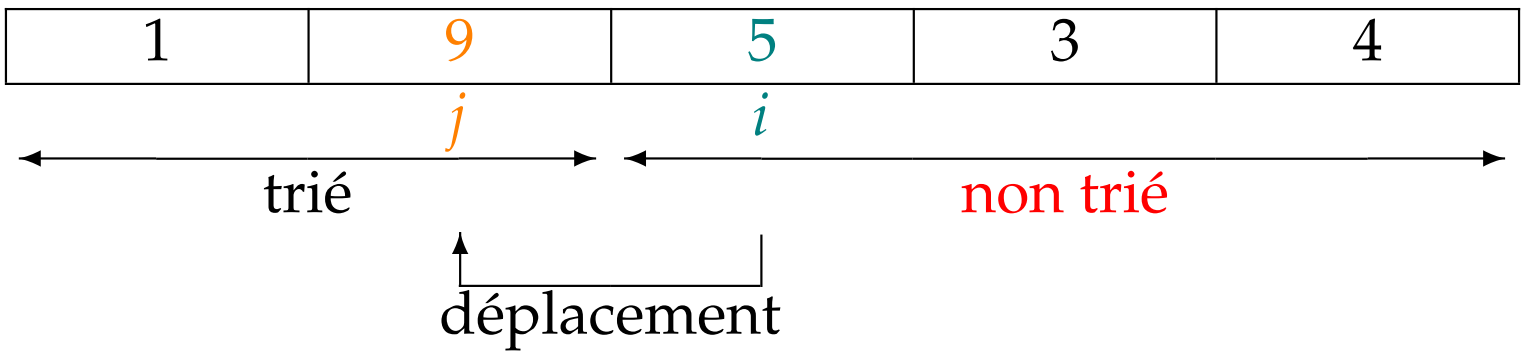


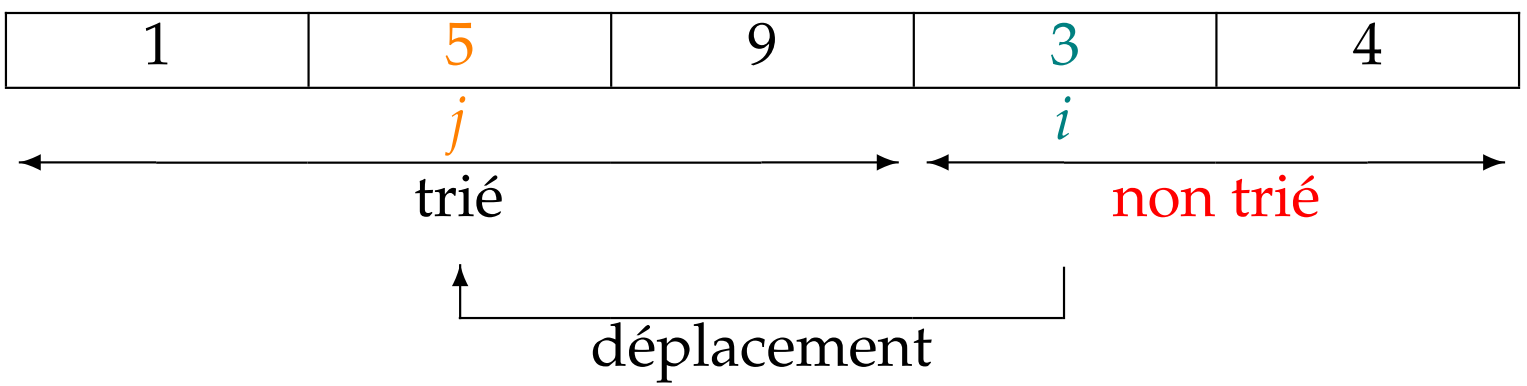
L'élément i est déjà à sa place

Tri par insertion (2)

A chaque étape i (de 1 à $n-1$) :

1. trouver la place relative j de l'élément i parmi les éléments triés
2. déplacer l'élément i à la place j

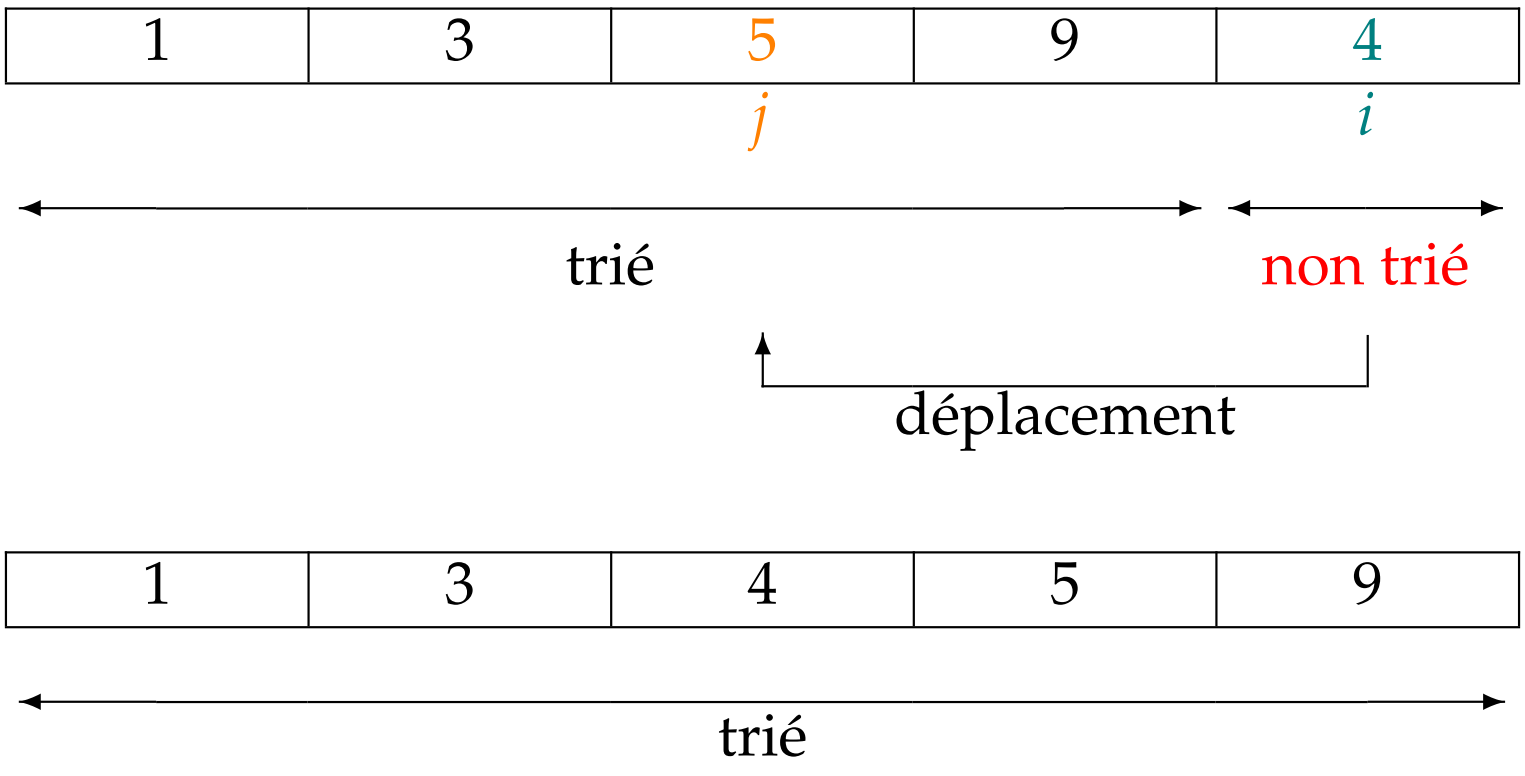




Tri par insertion (3)

A chaque étape i (de 1 à $n-1$) :

1. trouver la place relative j de l'élément i parmi les éléments triés
2. déplacer l'élément i à la place j



Recherche dichotomique (1)

La recherche dichotomique est un algorithme efficace pour trouver un élément dans une séquence triée.

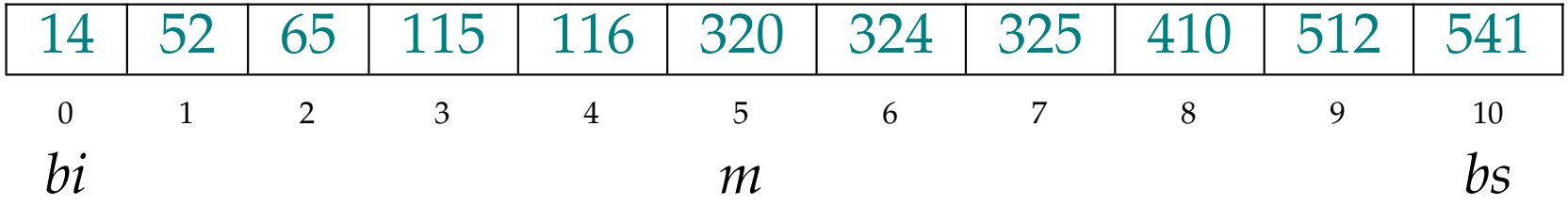
Pour trouver une valeur x dans un intervalle trié $[bi, bs]$:

- On compare x avec la valeur m au milieu de l'intervalle.
- Si la recherche n'est pas fructueuse, on réitère la recherche avec la première partie $[bi, m-1]$ de l'intervalle ou la seconde $[m+1, bs]$ suivant l'endroit où x est potentiellement présent.

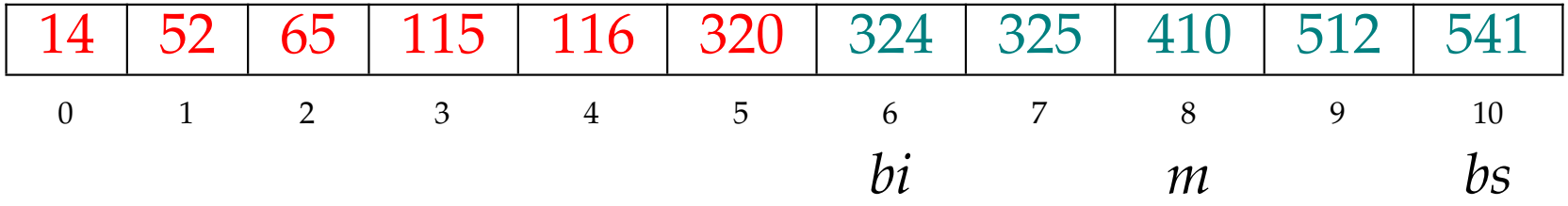
14	52	65	115	116	320	324	325	410	512	541
0	1	2	3	4	5	6	7	8	9	10
bi					m					bs

Recherche dichotomique (2)

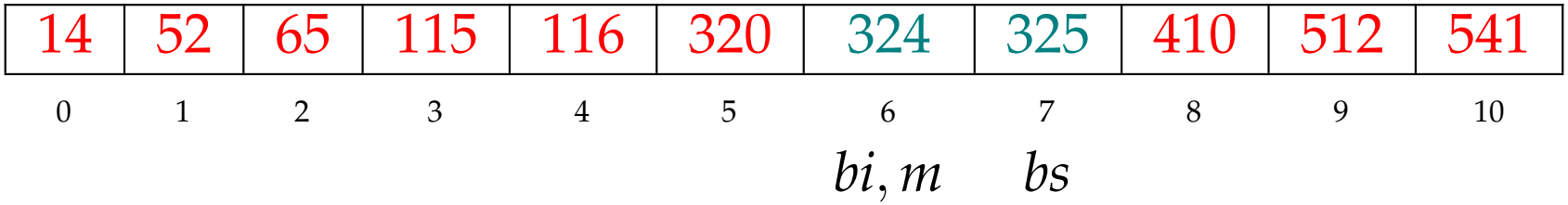
Recherche de 324 :



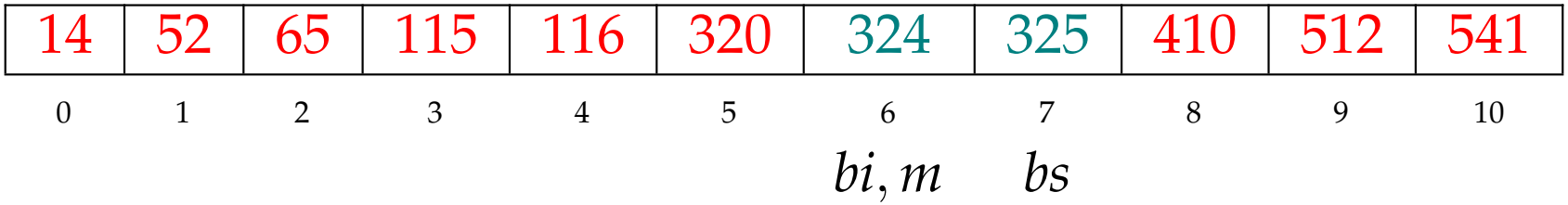
324 est supérieur à 320



324 est inférieur à 410



Recherche dichotomique (3)



On a trouvé l'élément 324.

Si *bi* était devenu supérieur ou égal *bs*, c'est que l'élément cherché n'était pas dans la séquence.

L'algorithme est efficace car au lieu de tester chaque élément (n opérations dans le pire des cas), on réduit à chaque fois l'espace de recherche par deux. On fera donc $\log_2(n)$ opérations au pire cas.

Discussion

Soit n, la taille des données :

Quel est approximativement, dans le pire des cas, le nombre d'opérations sur les données pour les tris par (1) sélection et par (2) insertion ?

- 1
- $\log(n)$
- $n \log(n)$
- n^2
- n^3
- ...

Exercices