Chapitre 9 : Exemple de complexité en *O(log n)*

- **Recherche dichotomique (ou binaire)**
  - **Objectif**: recherche de la position d'un élément de clé *x* donnée, dans un vecteur à *n* éléments indicés *0..n-1* et **triés en ordre non décroissant**

$\forall$ i,j. $0 \leqslant i < j < n \Rightarrow V[i] \leqslant V[j]$

  - **Méthode: Appeler *Recherche(0, n-1)* avec**

Recherche (bi, BS):=

m = (bi+BS)/2 **//indice du milieu**

soit *V[m]* est l'élément cherché $\Rightarrow$ *trouvé*

soit *V[m]* est trop grand $\Rightarrow$ *Recherche(bi,m-1)*

soit *V[m]* est trop petit $\Rightarrow$ *Recherche(m+1,BS)*

Recherche dichotomique (ou binaire)

- **Méthode raffinée:**

  Appeler *Recherche(0, n-1)* avec

  Recherche (bi, BS):=
  if (*[bi..BS]* ≠ ∅) *// si intervalle non vide*
  ***m = (bi+BS)/2* //indice du milieu**
  soit *V[m]* == *x* ⇒ *trouvé*
  soit *V[m]* > *x* ⇒ *Recherche(bi,m-1)*
  soit *V[m]* < *x* ⇒ *Recherche(m+1,BS)*
  else *// intervalle vide*
  x not in V

Programme C++

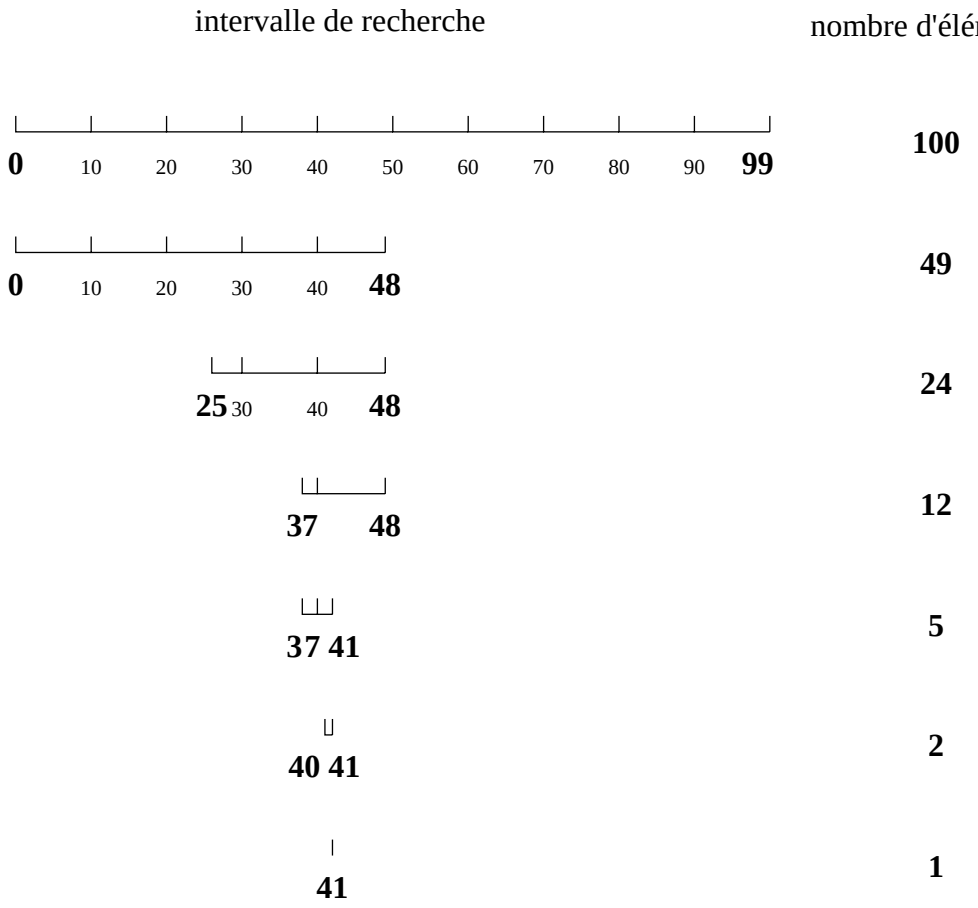
```
const int NOT_FOUND = -1;
typedef int elem;  // on pourrait avoir un autre type elem
typedef elem Vect[MAX];

int RechercheDichotomique(Vect V, int n, elem x)
{
   int bi = 0;
   int Bs = n-1;
   int m;
   do
   // Recherche (bi,BS)
   {
      m = (bi+Bs)/2;
      if (V[m] < x)
        bi = m+1;
      else
        Bs = m-1;
   }
   while(x != V[m] && bi <= Bs);
   if (x != V[m])
     m = NOT_FOUND;
   return m;
}
```

Complexité max de la recherche dichotomique

O(log(n))

- Exemple de recherche : ***V[41] < x < V[42]***



Chaque tour de la boucle do est en ***O(1)***.

La complexité de l'algorithme vaut *m* = **le nombre max de tours de boucle.**

Au pire, *[bi, BS]* divisé par 2 à chaque étape.
Donc: $2^{m-1} \leqslant n \leqslant 2^m$

Donc: $m \leqslant log_2(n) + 1$