# 1.Boucle imbriquées

1. 25 fois

2. 10 fois

3. n\*n

4. (n\*(n+1))/2

# 2.Tri de tableaux par sélection

1.

0 - tab = {3,25,50,8,1,3,49}

1- tab = {1,25,50,8,3,3,49}

2 - tab = {1,3,50,8,25,50,49}

3 - tab = {1,3,3,8,25,50,49}

4 - tab = {1,3,3,8,25,50,49}

5 - tab = {1,3,3,8,25,50,49}

6 - tab = {1,3,3,8,25,49,50}

2.

Le nombre maximum d’étape pour trier un tableau de taille n : n(n-1) / 2

On passe (n+1)+(n+2)+…+1+0 fois dans la boucle interne.

4. Nombre maximum d’étape : n(n-1) / 2

5. A1 :A chaque passage dans la boucle externe, tous les membre à gauche sont trié. On construit un tableau avec un élément de plus. La partie de droite est non trié et contient des éléments plus grand que la partie de gauche.

A2 :A chaque passage dans la boucle interne

# 3.Recherche dichotomique dans un tableau

1. tab = {3,6,12,21,30,33,35,40}

rechercheVite(tab, 17)

|  |  |  |  |
| --- | --- | --- | --- |
| X | Gauche | Droite | Milieu |
|  | 0(3) | 7(40) | 3(21) |
|  | 0(3) | 2(15) | 1(6) |
|  | 2(15) | 2(15) | 2(15) |
|  | 3(21) | 2(15) | 2(15) |

3.A chaque étape, on divise la taille du tableau par 2. Le nombre max d’étape est le nombre de fois où l’on peut diviser n par 2. Complexité en log(n).

4.On n’élimine pas tous les éléments déjà traités à la fin d’un itération.

# 4.Recherche du maximum dans un tableau

1.

public class TD1{

/\*max

Précondition: tab de n entier quelconque

Postcondition:return l'indice du plus grand entier

\*/

public static int max(int tab[]){

int max = 0;

for(int i = 0; i < tab.length; i++){

if(tab[i] > max){max = tab[i];}

}

return max;

}

2.

public class TD1{

public static int maxTrie(int tab[]){

int max = 0 ;

for(int i = 0; i < tab.length; i++){

if(tab[i] > max){max = tab[i];}

}

return max;

}

3.