TD 1

Exercice 1 :

1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | m | n | i | t[i] |
| Avant la boucle | 3 | 8 | / | / |
| Dans la boucle | 3 | 8 | 0 | 3 |
|  | 3 | 8 | 1 | 5 |
|  | 3 | 8 | 2 | 7 |
|  | 1 | 8 | 3 | 1 |
|  | 1 | 8 | 4 | 10 |
|  | 1 | 8 | 5 | 2 |
|  | 1 | 8 | 6 | 7 |

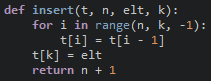
2. Le résultat de la fonction sera le minimum du tableau t, mais jusqu’à n – 1.

3. On fait n comparaisons.

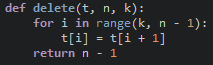
4. Nombre d’affectations : minimum 1, maximum n.

! Modifier les exemples de ces morts je suis attardé !

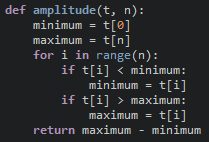
Exercice 2 :



Exercice 3 :

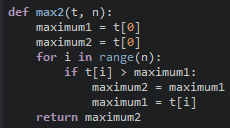


Exercice 4 :



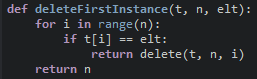
Exercice 5 :

1.



2. Il y a n comparaisons.

Exercice 6 :



Exercice 7 :

1.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| n | 10 | 9 | 8 |  |  |  |  |  |  |
| stop | False | False | True |  |  |  |  |  |  |
| numberOfElement | 9 | 8 | 8 |  |  |  |  |  |  |

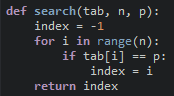
2. Meilleur cas : x €/ t

Pire cas : pour tout i, t[i] == x donc O(n²)

3. Meilleur cas : x €/ t ? O(n)

Exercice 8 :

Exercice 10 :



Exercice 11 :



Exercice 12 :

TD 2

Exercice 1 :

1. Le nombre d’affectations est de i + 1
2. Meilleur cas : O(1)

Pire cas : O(n)

Exercice 2 :

1. Meilleur cas : 0 affectations donc O(0)

Pire cas : 2n² donc O(n²)

Exercice 3 :

1. Nombre d’affectations :

Meilleur cas : n + 1

Pire cas : 3n² + 1

1. Meilleur cas : Θ(n)

Pire cas : Θ(n²)

1. Dans le cas où i + 1 est plus grand que n – 1, le tableau ne sera pas trié.

Exercice 4 :

1. (d + g) // 2
2. x = 20 :

(0 + 9) // 2 = 4 -> 20

x = 4 :

(0 + 9) // 2 = 4 -> 20

(0 + 4) // 2 = 2 -> 13

(0 + 2) // 2 = 1 -> 4 (flemme la suite)

TD 4

**Exercice 1**

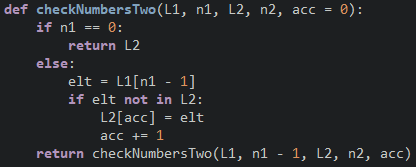
1. Pile d'exécution de checkNumbers([2, 3, 6, 2], 4, 0, 1) GEKOMPRI -> False

2. SAME GEKOMPRI -> True

3. Recherche des doubons dans la liste

4. Meilleur cas : O(1) ; Pire cas : O(n²)

**Exercice 2**

1. Pile d’exécution checkNumbersTwo:  
   1) elt = L1[8] = 9 -> pas dans L2 -> L2[0] = 9  
   2) elt = L1[7] = 3 -> pas dans L2 -> L2[1] = 3  
   3) elt = L1[6] = 1 -> pas dans L2 -> L2[2] = 1  
   4) elt = L1[5] = 1 -> dans L2  
   5) elt = L1[4] = 2 -> pas dans L2 -> L2[4] = 2  
   6) elt = L1[3] = 4 -> pas dans L2 -> L2[5] = 4  
   7) elt = L1[2] = 3 -> dans L2  
   8) elt = L1[1] = 5 -> pas dans L2 -> L2[7] = 5  
   9) elt = L1[0] = 3 -> dans L2  
   10) L2 = [9, 3, 1, None, 2, 4, None, 5, None]
2. Met les éléments de L1 en partant de la fin, au début de L2, sans affecter les doublons.  
   Cependant à l’emplacement théorique des doublons, il y aura None dans L2.
3. 
4. (insérer fonction pas encore faite)

TD 5

1. -> 77, 44, 99, 66, 33, 55, 88, 22  
   -> 22, 44, 99, 66, 33, 55, 88, 77  
   -> 22, 33, 99, 66, 44, 55, 88, 77  
   -> 22, 33, 44, 66, 99, 55, 88, 77  
   -> 22, 33, 44, 55, 99, 66, 88, 77  
   -> 22, 33, 44, 55, 66, 99, 88, 77  
   -> 22, 33, 44, 55, 66, 77, 88, 99  
   -> 22, 33, 44, 55, 66, 77, 88, 99
2. ?
3. ?
4. Θ(n²)
5. O(n²)
6. n – 1
7. n – 1
8. ?
9. -> 77, 44, 99, 66, 33, 55, 88, 22  
   -> 44, 77, 99, 66, 33, 55, 88, 22  
   -> 44, 66, 77, 99, 33, 55, 88, 22  
   -> 33, 44, 66, 77, 99, 55, 88, 22  
   -> 33, 44, 55, 66, 77, 99, 88, 22  
   -> 33, 44, 55, 66, 77, 88, 99, 22  
   -> 22, 33, 44, 55, 66, 77, 88, 99