Semaine du 7 au 11 décembre

En raison des conseils de classe de Terminale du 10 et 11 décembre (fin à 20h30), ce document ne vous arrive que samedi 12 décembre. Il n'y aura pas de controle la semaine prochaine.

Séance 2 (jeudi 10 décembre ou vendredi 11 décembre) :

Cours sur la complexité : voir la **nouvelle version** du cours sur les algorithmes

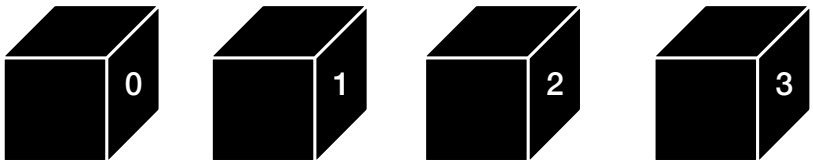
Les notions vues dans ce cours sont à savoir par coeur et feront souvent l'objet de questions dans les contrôles à venir.

Exercice 3 et 6 de la feuille d'exercices

Correction page suivante…

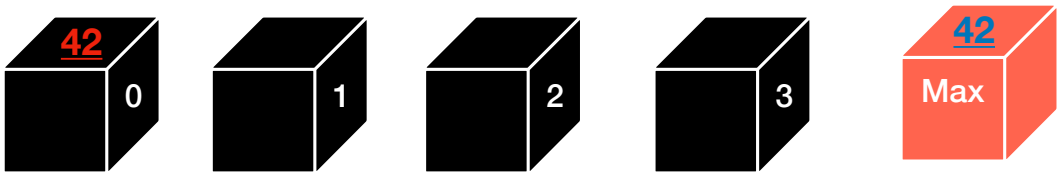
Exercice 3 :

a) Pour **trouver le maximum d'un ensemble de nombres**, encore une fois, réfléchissez en terme de cases-mémoire dans l'ordinateur :

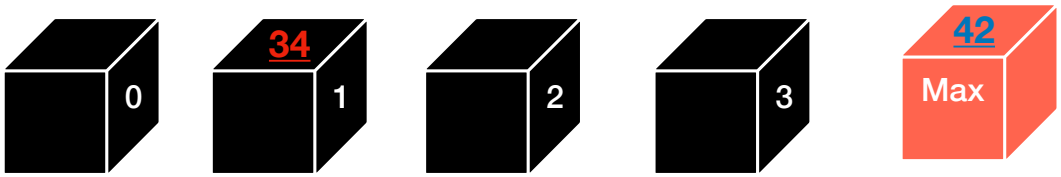


Tant que l'on n'ouvre pas les boites de la mémoire de mon ordinateur, nous ne savons pas ce qu'il y a dedans… Tout ce que nous savons est finalement le numéro de la boite (en théorie, l'adresse hexadécimale mémoire)

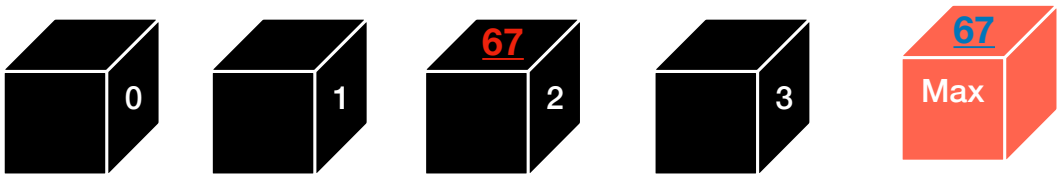
On va donc devoir ouvrir chaque boite l'une après l'autre :



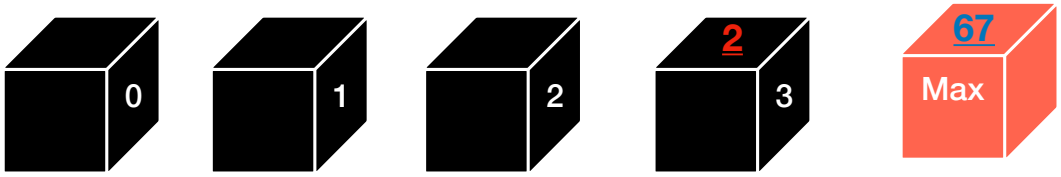
numéro 0 : Ok, pour l'instant, le maximum est 42 (le premier que je viens d'ouvrir). Je l'enregistre dans une nouvelle case mémoire



numéro 1 : Ah non, 34< 42 = max donc, le maximum ne change pas



numéro 2 : ok, 67>42 =max donc, le maximum change



numéro 3 : ok, 2<67 =max donc, le maximum ne change pas

On a trouvé le maximum : on voit que l'on va créer une nouvelle variable et que l'on va devoir faire une boucle sur les indices (0 à 3) et un test (est-ce que le nombre est plus grand que le max?).

Correction page suivante

Cela donne l'algorithme suivant :

```
max = T[0]     # élément n°0 du tableau
Pour i allant de 0 jusqu'à la taille du tableau - 1, on fait :
     si T[i] > max :
          max = T[i]
renvoyer max
```

b) Le test se fait avec le tableau vu dans le cours :

i	T[i]	T[i] > max	max
0	3	FALSE	3
1	5	TRUE	5
2	1	FALSE	5
3	8	TRUE	8
4	2	FALSE	8

Le programme renvoie 8.

c) On applique le principe du cours et on compte le nombre de boucles imbriquées. Il n'y en a qu'une seule qui dépend de la taille du tableau.

Donc, la complexité est en $\mathcal{O}(n)$.

Exercice 6 :

Dans cet exercice, on compte les indices à partir de 1 jusqu'à la taille du tableau n (3 dans l'exemple).

a)

$T[2][1] = 1$

$T[1][3] = 1$

$T[3][3] = -5$

b)

On a une double boucle : pour chaque valeur de i, nous allons parcourir TOUTES les valeurs de j.

c)

Nous avons une double boucle, donc on applique la propriété du cours. La complexité est en $\mathcal{O}(n^2)$.

On remarque d'ailleurs que pour chaque valeur de i, on parcourt TOUTES les valeurs de j. Donc n^2.

i	j	T[i][j]	T[i][j] ≥ 0	somme
1	1	-5	FALSE	0
	2	2	TRUE	2
	3	1	TRUE	3
2	1	1	TRUE	4
	2	-4	FALSE	4
	3	3	TRUE	7
3	1	7	TRUE	14
	2	9	TRUE	23
	3	-5	FALSE	23

Correction page suivante

d) On va recycler le même algorithme que dans cet exercice.
La trace correspond à faire la somme des éléments diagonaux. Les indices de ces éléments sont ainsi : 1,1 — 2,2 — 3,3 …
Ils sont donc de la forme **i,i**

Ainsi, on va pouvoir écrire l'algorithme suivant :
Données :
T : tableau d'entiers de taille n par n
Tr : trace (somme des éléments diagonaux)

```
Tr = 0
pour i allant de 1 à n, faire :
      Tr = Tr + T[i][i]
renvoyer Tr
```

Correction page suivante