DM 1 : Le président mégalomaniaque

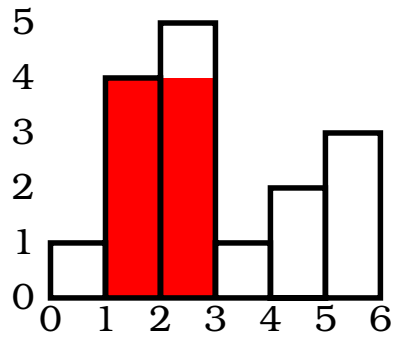
À rendre au plus tard le 9 Octobre 2018 à 10h15

Pour les calculs de complexité, les opérations arithmétiques (additions, multiplicatiosn, . . .), les constructeurs (ajouts d'un un élément en tête d'une file, . . .), les destructeurs (extraction d'un élément en haut d'une pile, . . .), les lectures et les affectations sont considérés comme s'effectuant en $O(1)$. Sauf contre-indication, on s'intéresse à la complexité dans le pire des cas.

Un président mégalomaniaque souhaite faire peindre le plus grand tableau de lui qu'il soit possible de faire. Il réquisitionne donc une rue de New York, et souhaite utiliser les façades des immeubles pour cet effet. Il vous ordonne de trouver le plus grand rectangle possible (en aire) pour ce majestueux tableau.

L'entrée du problème est un tableau d'entiers positifs, représentant les hauteurs des immeubles de la rue. La largeur des immeubles est toujours de 1. La sortie est un rectangle tenant sur la façade des immeubles (sans dépasser), et d'aire maximale. Par exemple :

Entrée	**Sortie**	
$[1;4;5;1;2;3]$	Début :	1
	Fin :	3
	Hauteur :	4
	Aire :	8



Question 1 : Trouvez un algorithme naïf résolvant le problème en temps $O(n^2)$, où n est le nombre d'immeubles. Prouvez sa correction et sa complexité.

Peinant à expliquer le sens du mot "quadratique" au président mégalomaniaque, il vous contraint à trouver un algorithme plus rapide. Jean-Maurice, votre meilleur ami, vous dit « J'ai une super idée ! C'est du diviser pour régner, et la complexité de diviser pour régner c'est toujours $O(n\ log(n))$. C'est très simple, tu prends le plus petit immeuble et tu coupes la rue à cet endroit là, puis tu regardes le plus grand rectangle à gauche (récursivement), le plus grand rectangle à droite (récursivement), et le rectangle de hauteur la hauteur du petit immeuble. Tu choisis le rectangle d'aire maximale parmi les 3, et c'est fini ! »

Question 2 : Écrivez l'algorithme proposé par Jean-Maurice et prouvez sa correction. Montrez que sa complexité est $O(n^2)$, où n est le nombre d'immeubles. Exhibez une situation dans laquelle l'algorithme ne s'exécute pas en temps $O(n\ log(n))$.

« Ah, mince. J'avais oublié ça. Le principal problème, c'est de trouver à chaque fois le plus petit immeuble. Donc moi, je m'occupe d'aller trouver les immeubles petits, et toi, tu t'occupes d'en déduire la meilleure position pour le tableau ».

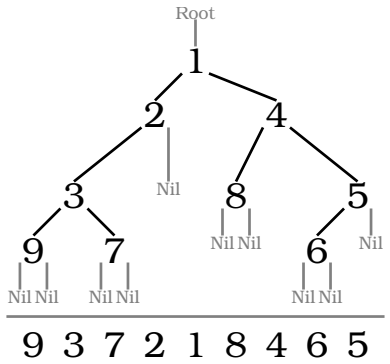
Question 3 : Pour cette question, on suppose que vous avez accès à une fonction "JM", qui cherche dans un tableau d'immeubles le plus petit de ces immeubles en temps constant $O(1)$. Comment devez-vous modifier l'algorithme de la Question 2 pour obtenir un algorithme en $O(n)$, où n est le nombre d'immeubles ?

Question Bonus : Comment pourriez vous implémentez la fonction "JM" afin d'obtenir une complexité globale en temps **moyen** $O(n\ log(n))$, où n est le nombre d'immeubles.
Indice : Expliquez comment pré-traiter le tableau d'immeubles de la manière suivante :

Tableau d'immeubles
$[9;3;7;2;1;8;4;6;5]$ $\rightarrow$ Immeubles triés par hauteur
$[(4,1);(3;2);(1;3);(6,4);(8,5);(7,6);(2,7);(5,8);(0,9)]$

$\rightarrow$

Arbre d'immeubles



Malheureusement, n'étant qu'un simple informaticien et non un artiste employé par le président, Jean-Maurice ne peut s'éloigner de vous sans se faire expulser par la sécurité. Néanmoins, il a une autre idée. « En fait, j'ai encore mieux. En calculant petit à petit, et ajoutant les immeubles un à un, on a un algorithme linéaire ! C'est très simple, tu mémorises le plus grand des rectangles pour les x premiers immeubles, puis on ajoute l'immeuble $x+1$, et on regarde s'il agrandit ce rectangle. »

Question 4 : Trouvez un contre-exemple à l'algorithme de Jean-Maurice.

« Ah, oui ... Du coup, il suffit de mémoriser le plus grand des rectangles actuels, et aussi autant de rectangles "ouverts" que nécessaire, c'est à dire les rectangle qui peuvent encore grossir. À chaque nouvel immeuble, on met à jour les rectangles ouverts, et le plus grand des rectangles. »

Question 5 : Trouvez un algorithme implémentant l'idée que viens d'avoir votre ami, et s'exécutant en temps $O(n^2)$, où n est le nombre d'immeubles. Prouvez sa correction, et sa complexité.

Inquiet de n'avoir toujours pas mieux qu'un algorithme quadratique, Jean-Maurice vous rassure : « Le principal problème avec notre algorithme, c'est qu'on doit mettre à jour toutes les infos à chaque nouvel immeuble. Ça serait beaucoup plus rapides si on ne faisait que le minimum de mises à jour nécessaire. ».

Question 6 : Complétez l'algorithme suivant afin d'obtenir un algorithme correct de complexité linéaire $O(n)$, où n est le nombre d'immeubles. Prouvez la correction et la complexité.

```
Entrée :  T     // Tableau d'entiers de taille n
A ←       0     // Aire du plus grand rectangles
Pos ←     (0,0,0)     // (Abscisse début, Hauteur, Abscisse fin) du plus grand des rectangles
S ←       [ ]     // Pile des "coins haut gauche" des rectangles qui peuvent s'étendre
Pour      i = 0 à n − 1 faire

              // À compléter (une dizaine de lignes)

      Fin
Tant que  S ≠ [ ] faire
          (x,y) ← Extraire(S)     // Extrait l'élément en haut de la pile
          Si (n − x) × y > A alors
              A ← (n − x) × y
              Pos ← (x, y, n)
          Fin
      Fin
Sortie :  Pos     // Le plus grand rectangle
```

Rappel : Une pile est une structure de donnée disposant de 2 opérations :

- *`Ajouter`(S, (x,y)) : ajoute l'élément (x,y) sur le haut de la pile (ne renvoie rien)*
- *`Extraire`(S) : enlève l'élément qui est sur le haut de la pile et le renvoie*

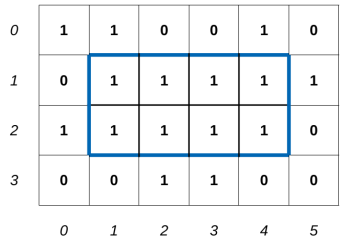
Le président a approuvé votre plan. Il est maintenant l'heure de l'appliquer. Vous allez aux archives municipales pour trouver les registres contenant les hauteurs des différents immeubles. Malheureusement, ils sont classés par date de construction, et non par numéro de rue. L'entrée du problème est maintenant un tableau de *paires d'entier*, le premier entier étant la position de l'immeuble sur la rue (supposées toutes distinctes, allant de 0 à $n-1$, où n est le nombre d'immeubles), et le deuxième étant la hauteur de l'immeuble. Jean-Maurice est terrifié : « Trier tout ça va nous prendre un temps fou ! Les tris c'est au mieux $O(n\ log(n))$! »

Question 7 : Trouver un algorithme permettant de trier la liste d'immeuble en fonction de leur position sur la rue, en temps linéaire $O(n)$, où n est le nombre d'immeubles. (*Indice :* On rappelle qu'à chaque position dans la rue correspond exactement un immeuble.)

Les travaux sont sur le point de commencer, la position parfaite pour le tableau est trouvée, lorsque vous aprenez qu'il y a dans la rue des députés qui refusent catégoriquement que la façade de leur appartement soit utilisée pour peindre un portrait du président.

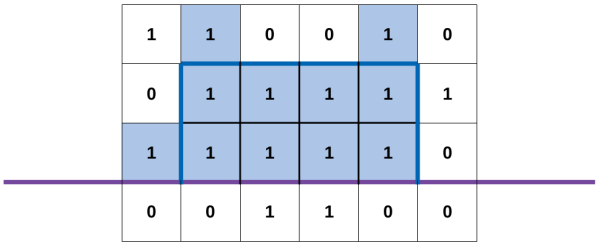
L'entrée du problème est maintenant une matrice $n \times m$ de 0 et de 1, représentant la présence ou l'absence d'un appartement utilisable. Le but est de trouver le plus grand rectangle (en aire) ne couvrant que des 1. Exemple :

Entrée	Sortie	
$[[1;1;0;0;1;0],$	Début :	$(2,1)$
$[0;1;1;1;1;1],$	Largeur :	4
$[1;1;1;1;1;0],$	Hauteur :	2
$[0;0;1;1;0;0]]$	Aire :	8



Question 8 : Quelle serait la complexité d'un algorithme naïf pour résoudre ce problème ?

Pensant que vous allez devoir reprendre tout votre travail de zéro pour pouvoir trouver un algorithme satisfaisant le président, vous êtes catastrophés. Heureusement votre ami vous rassure : « Regarde ce dessin, c'est un peu comme s'il suffisait de dire que la rue commence plus haut et d'ignorer les blocs qui flottent ! »



Question 9 : Vous décidez de suivre l'idée de Jean-Maurice, vous allez regarder toutes les hauteurs de rue possible et appliquer l'algorithme que vous aviez auparavant. Formalisez cette idée et prouvez qu'elle fonctionne.

Question 10 : Ecrire un algorithme qui transforme les lignes de la matrice d'entrée en lignes de "rue" s'effectuant en $O(n \times m)$. (Ci-dessous un exemple d'entrée et de sortie de l'algorithme en question).

Entrée	Sortie
$[[1;1;0;0;1;0],$	$[[1;1;0;0;1;0],$
$[0;1;1;1;1;1],$	$[0;2;1;1;2;1],$
$[1;1;1;1;1;0],$	$[1;3;2;2;3;0],$
$[0;0;1;1;0;0]]$	$[0;0;3;3;0;0]]$

Question 11 : Ecrivez l'algorithme qui prend en entrée la matrice de zéro et de un et renvoie le rectangle d'aire maximale. Quelle est sa complexité ?