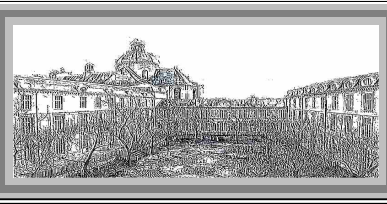
LYCEE CHARLEMAGNE
Lundi 13 mai
M.P.S.I.2



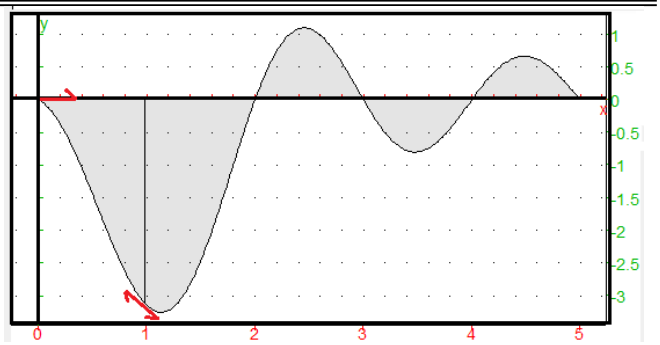
2023 2024

TD28

◦0◦ Trouvez un nombre qui est soixante dix sept fois plus grand que le chiffre de ses unités.

◦1◦ Le problème que Pavl Halmos aimait poser à ses étudiants : les concombres sont composés de quatre vingt dix neuf pour cent d'eau (si si, et nous, c'est plus que quatre vingt dix pour cent). Le Cours des Halles en bas de chez moi en a acheté cinq cent kilos. Mais après le week-end, ils ne sont plus formés que de quatre vingt dix huit pour cent d'eau. Combien de kilos lui reste-t-il à vendre ?

◦2◦ ♡ J'ai l'impression que $x \longmapsto \dfrac{\sin(\pi.x)}{\ln(x)}$ se prolonge par continuité en 0 et en 1. Prouvez le. Et donne moi l'équation de la tangente en 0 puis en 1.



◦3◦ ♠ Soit f une application uniformément continue de $\mathbb{Q}$ dans $\mathbb{R}$. On veut montrer qu'elle se prolonge en une application continue de $\mathbb{R}$ dans $\mathbb{R}$ "par densité de $\mathbb{Q}$".
On se donne un irrationnel a. Montrez qu'il existe une suite (r_n) de rationnels qui converge vers a. Montrez que les suites (r_n) et $(f(r_n))$ sont de Cauchy et convergent donc. L'élève A veut alors poser $f(a) = \lim\limits_{n\mapsto+\infty} f(r_n)$. L'élève B l'arrête en indiquant : mais si tu avais pris une autre suite (c_n), qui dit que tu aurais obtenu la même limite $f(a) = \lim\limits_{n\mapsto+\infty} f(c_n)$ *($f(a)$ dépendrait il du choix de la suite choisie ?)*. Que pensez vous de son objection ? Et que pensez vous de l'élève qui explique alors $(r_0,\ c_0,\ r_1,\ c_1,\ r_2, \ldots)$?
Montrez que f ainsi définie est à son tour uniformément continue.

◦4◦ Montrez que pour tout a, $\displaystyle\int_0^a \frac{Atan(a.t)}{t}.dt$ existe (on la note $F(a)$).
Montrez : $F'(a) = 2.\dfrac{Arctan(a^2)}{a}$ *(si si, il y a bien un 2, et l'astuce constatera à changer de variable pour qu'il n'y ait plus de a dans la fonction intégrée, juste dans les bornes)*.

◦5◦ ♡ Prolongez la aux bornes du domaine : $x \longmapsto \ln(x).\ln(1-x)$. Trouvez son maximum, donnez son axe de symétrie.
Admet elle un développement limité en 0 ?

◦6◦ Écrivez un script `Python` qui pour une permutation `sigma` donnée (sous forme de liste des images) déterminez le plus long cycle.
Par exemple, pour `[0, 3, 8, 1, 9, 6, 4, 10, 2, 7, 5]` il répondra `[4, 9, 7, 10, 5, 6]` (ou `[5, 6, 4, 9, 7, 10]`) puisque c'est le même.
Comme votre programme devra travailler sur des permutations très longues, il doit être optimisé pour avoir tous les points, et ne pas perdre de temps.

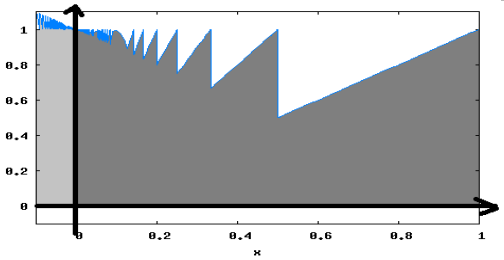
a	Pourquoi est il vrai que si f est continue à droite et continue à gauche en a, alors elle est continue en a.	.
b	Pourquoi est il vrai que si f est dérivable à droite et dérivable à gauche en a, alors pourquoi elle n'est pas forcément dérivable en a.	
c	Pourquoi est il vrai que si f est dérivable à droite et dérivable à gauche en a, alors elle est continue en a.	
d	Un $\mathcal{E}$ tracé par Enis est il homéomorphe à un $\mathcal{E}$ tracé par Ines ?	
e	Vrai ou faux : $x \longmapsto \ln(\lvert x\rvert)$ a pour dérivée $x \longmapsto \dfrac{1}{\lvert x\rvert}$?	
f	Vrai ou faux : f est dérivable en a si et seulement si $x \longmapsto f(x+a)$ est dérivable en 0.	
g	Vrai ou faux : si f est dérivable en a de dérivée $f'(a)$, alors $x \longmapsto f(-x)$ est dérivable en $-a$ de dérivée $f'(a)$.	
h	Vrai ou faux : si $x \longmapsto Arctan(f(x))$ est dérivable en a alors f est dérivable en a.	
i	Vrai ou faux : si $x \longmapsto (f(x))^2$ est dérivable en a alors f est dérivable en a.	
j	Vrai ou faux : si f est paire, alors f' est nulle en 0.	
et même http ://rogermansuy.fr/HX2/Derivables.html		

◦7◦

Homéomorphisme : application continue bijective dont la réciproque est aussi continue.

♡ Donnez les points de discontinuité sur $[0,\ 1]$ de $x \longmapsto x.\left[\dfrac{1}{x}\right]$. Prolongez cette application par continuité en 0.
♣ Exprimez l'intégrale de cette application de 0 à 1 comme somme d'une série numérique. Donnez sa valeur, en admettant $\displaystyle\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6}$.

◦8◦



◦9◦ Montrez que l'application $x \longmapsto \dfrac{x^3+3.x}{3.x^2+1}$ est croissante sur $\mathbb{R}$. Étudiez la suite u_0 donné et pour tout n $u_{n+1} = \dfrac{(u_n)^3+3.u_n}{3.(u_n)^2+1}$. Discuter suivant u_0.

◦10◦ Pour tout n, on définit $f_n = t \longmapsto \dfrac{t^n - t^{2.n}}{1-t}$. Montrez que f_n se prolonge par continuité en 1. On note alors I_n l'intégrale de 0 à 1 de f_n. Montrez $I_n = \displaystyle\sum_{k=1}^{n} \frac{1}{n+k}$. Déduisez par comparaison série intégrale que I_n converge vers $\ln(2)$ quand n tend vers l'infini.

◦11◦ On définit : $f = x \longmapsto e^{\frac{1}{\ln(x)}}$ de $]0,\ 1[$ dans $\mathbb{R}$. Prolongez la par continuité en 0 et en 1. est elle alors C^1 ? Est-ce un homéomorphisme de $[0,\ 1]$ dans $[0,\ 1]$?
Est-ce un difféomorphisme de $]0,\ 1[$ dans $]0,\ 1[$?
Est-ce un difféomorphisme de $]0,\ 1]$ dans $]0,\ 1]$?
Déterminez f^{-1}.

◦12◦ On définit $f = \theta \longmapsto \tan(\theta) + \dfrac{2}{2.\theta - \pi}$. Prolongez f par continuité en $\pi/2$[1] *(on notre $\overline{f}$ l'application prolongée, juste pour avoir la rigueur emmerdante des concours)*. Montrez que $\overline{f}$ est dérivable en $\pi/2$. Montrez que $\overline{f}$ est C^1 sur $[0,\ \pi/2]$.
Calculez $\displaystyle\int_0^{\pi/2} \overline{f}(\theta).d\theta$.
On pose $I_n = \displaystyle\int_0^{\pi/2} \overline{f}(\theta).\sin(n.\theta).d\theta$, Montrez que I_n tend vers 0 quand n tend vers l'infini *(by parts)*.

◦13◦ Limites en 0^+ et en l'infini de $\left(\dfrac{1+3^x}{2}\right)^{\frac{1}{x}}$.

◦14◦ On travaille avec les entiers de 0 à 4 pour les opérations modulo 5. Montrez que $\begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix}$ est nilpotente.

---

1. pensez à écrire $x = \dfrac{\pi}{2} + h$ quand x doit tendre vers $\pi/2$, car c'est en 0 que vous maitrisez vos développements limités

Combien y a-t-il de matrices de taille 2 sur 2 nilpotentes.
Combien y a-t-l de matrices 2 sur 2 de rang 2 ?
Combien y a-t-il de couples de matrices $(A,\ B)$ avec A, B et $A+B$ nilpotentes ?
Le rang est le nombre de colonnes linéairement indépendantes.

◦15◦ Auguste, Baptiste et Clara sortent de colle (d'anglais ou de physique) et indiquent leur note :

Auguste		j'ai 6	j'ai 2 de moins que Baptiste	j'ai 1 de plus que Clara
Baptiste		Clara a 9	je n'ai pas la plus mauvaise note	la différence entre Clara et moi est 3
Clara		Auguste a 7	j'ai moins qu'Auguste	Baptiste a 3 de plus qu'Auguste

Ce n'est pas cohérent tout ça. En effet, chacun a menti une fois et dit la vérité deux fois. Quelle est la note de chacun ?

◦16◦ ♡ Déterminez $\lim_{x\to 4} \dfrac{x^3-4^3}{x^5-4^5}$.

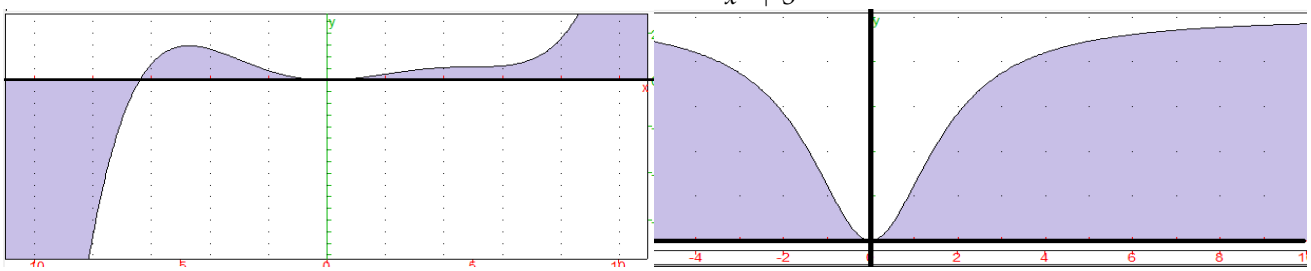
a et b strictement positifs donnés, déterminez si elle existe la limite de $\dfrac{x^a-\mu^a}{x^b-\mu^b}$ quand x tend vers μ avec $\mu=\dfrac{a+b}{2}$.

◦17◦ ♡ Comparez les probabilités de "*avoir au moins un* 6 *avec six dés*" et "*avoir au moins deux* 6 *avec douze dés*" (et pourquoi pas "*avoir au moins un* 12 *avec douze dés*" ?).

◦18◦ ♡ Soit f de classe C^3 sur un voisinage de a. Calculez $lim_{h\to 0}\dfrac{1}{h^2}\cdot\begin{vmatrix} f(a-h) & f(a) \\ f(a) & f(a+h)\end{vmatrix}$ (indication : DL d'ordre 2).

◦19◦ ♡ Donnez un intervalle le plus grand possible sur lequel $x\longmapsto 3.x^5-20.x^4-110.x^3+900.x^2$ est convexe.

Donnez un intervalle le plus grand possible sur lequel $x\longmapsto \dfrac{x^2}{x^2+3}$ est convexe.



◦20◦ ♣☺ On définit : $I=\displaystyle\int_0^{\pi/4}\ln(1+\tan(t)).dt$, $J=\int_0^{\pi/4}\ln(\cos(x)).dx$, $K=\int_0^{\pi/4}\ln(\cos(x-\pi/4)).dx$ et $L=\displaystyle\int_0^{\pi/4}\ln(\sin(x)+\cos(x)).dx$.
Montrez : $I=L-J$, $J=K$. Calculez $L-K$ par trigonométrie. Déduisez la valeur de I.

◦21◦ On suppose que f convexe de $\mathbb{R}$ dans $\mathbb{R}$ admet un minimum local en a et aussi en b. Montrez alors $f(a)=f(b)$ et montrez que f est constante sur $[a,\ b]$.

◦22◦ Montrez que $x\longmapsto\ln(1+e^x)$ est convexe.
Déduisez pour a et b positifs : $1+\sqrt{a.b}\leqslant\sqrt{(1+a).(1+b)}$ et plus généralement
Déduisez : $1+\left(\displaystyle\prod_{k=1}^{n}x_k\right)^{\frac{1}{n}}\leqslant\left(\prod_{k=1}^{n}(1+x_k)\right)^{\frac{1}{n}}$ pour des x_k strictement positifs.
Déduisez $\sqrt[n]{a_1\dots a_n}+\sqrt[n]{b_1\dots b_n}\leqslant\sqrt[n]{(a_1+b_1)\dots(a_n+b_n)}$ pour des réels strictement positifs a_j et b_i.
$1+\sqrt[n]{a_1\dots a_n}\leqslant\sqrt[n]{(1+a_1)\dots(1+a_n)}$.

◦23◦ Montrez que si f est convexe, alors pour tout a positif, $t\longmapsto e^{a.f(t)}$ est convexe.

Réciproquement, on suppose que pour tout a positif, $t\longmapsto e^{a.f(t)}$ est convexe. Montrez alors que $t\longmapsto\dfrac{e^{a.f(t)}-1}{a}$ est convexe. Déduisez que f est convexe.

◦24◦ f est de classe C^2 et convexe. Montrez que $x \longmapsto \int_{t=x-1}^{x+1} f(t).dt$ est aussi convexe.

◦25◦ On définit $f = t \longmapsto (t^2+1).e^{-t}$. Donnez l'intervalle le plus grand possible sur lequel f est croissante et convexe.
Même question avec décroissante et convexe.
Même question avec décroissante ou convexe.

◦26◦ ♡ Sur quel(s) intervalle(s) $t \longmapsto (1-t^2).e^t$ est elle convexe ?

◦27◦ ♡ Calculez ces sommes multiples pour n donné | $A_n = \sum_{0 \leqslant j < i \leqslant n} \frac{j}{i}$ | $B_n = \sum_{i+j=n} i.j$ | $C_n = \sum_{\substack{0 \leqslant i \leqslant n \\ 0 \leqslant j \leqslant n}} Max(i,\ j)$ |

Si vous n'avez pas confiance, écrivez un script Python qui prend en entrée n et les calcule.

◦28◦ ♡ Montrez pour tout x réel : $e^x \geqslant 1 + x + \frac{x^2}{2} + \frac{x^3}{6}$.

◦29◦ Soit f deux fois dérivable avec f'' positive. Montrez que si $f'(a)$ est nul, alors f admet un minimum en a.

◦30◦ Source : d'après un oral de Centrale. On considère $n \geqslant 2$ joueurs, numérotés de 1 à n, participant à un tournoi où chacun affronte tous les autres, sans égalité possible dans une rencontre. On définit la matrice $A = (a_i^k)_{i \leqslant n, k \leqslant n}$ de la manière suivante : $a_i^i = 0$, $a_i^k = 1$ si i a gagné contre k et $a_i^k = -1$ si k a gagné contre i. Écrivez une procédure renvoyant pour n donné une matrice de tournoi aléatoire.

Voici une procédure *(lourde, de complexité n!)* de calcul de déterminant, mais il manque une sous-procédure ; écrivez la :

```
def det(M) :
....if len(M)==0 :
........return(1)
....S, signe = 0, 1
....for i in range(len(M)) :
........S+=signe*M[i][0]*det(delRowCol(M,i,0))
....signe=-signe
....return(S)
```

On constate pour n impair que ces déterminants sont toujours nuls. Prouvez le.

$J_n = (1)_{i \leqslant n, k \leqslant n}$. Calculez $\det(J_n - I_n)$.

Soient M et N deux matrices coefficients entiers telles que $M - N$ ait tous ses coefficients pairs. Montrez que $\det(M)$ et $\det(N)$ sont deux entiers de même parité.

Déduisez que les matrices de tournois aléatoires en taille paire sont inversibles.

◦31◦ Petits exercices du cours :
a - La réciproque d'une application convexe est elle convexe ? concave ?
Qu'allez vous utiliser ? Déterminant ? Trois cordes ? Petite inégalité ? Grande inégalité ?
Et d'ailleurs, la réciproque existe-t-elle ?

b - Si f est convexe et g convexe et croissante, alors $g \circ f$ est convexe. Pour démontrer ça, qu'allez vous utiliser ? déterminant ? Trois cordes ? Petit inégalité ? grande inégalité ?

c - Une application convexe de $\mathbb{R}$ dans $\mathbb{R}$ peut elle être majorée ?

d - Pouvez vous construire f convexe de minimum 1 atteint à la fois en 2 et en 3 ?

e - Si le graphe de f (dérivable) est toujours au dessus de ses tangentes, peut on déduire que f est convexe.

◦32◦ *Le but de cette épreuve (filière PSI, mais grand concours) est de décider s'il existe, entre deux villes données, un chemin passant par exactement k villes intermédiaires distinctes, dans un plan contenant au total n villes, reliées par m routes. L'algorithme d'exploration*

naturel s'exécute en un temps $O(n^k.m)$. L'objectif est d'obtenir un algorithme qui s'exécute en un temps $O(f(k).n.(n+m))$, qui croît polynomialement en la taille $n+m$ du problème, quelle que soit la valeur de k demandée.

La complexité ou temps d'exécution d'un programme Π (fonction ou procédure [2]) est le nombre d'opérations élémentaires (additions, multiplications, affectations, tests...) nécessaires à l'exécution de Π. Lorsque cette complexité dépend de plusieurs paramètres n, m et k, on dira que Π a une complexité $O(\phi(n,m,k))$ lorsqu'il existe quatre constantes absolues A, n_0, m_0 et k_0 telles que la complexité de Π soit inférieure ou égale $A.\phi(n,m,k)$ pour tout $n \geqslant n_0$, $m \geqslant m_0$ et $k \geqslant k_0$. Lorsqu'il est demandé de préciser la complexité d'un programme, le candidat devra la justifier si elle ne se traduit pas directement de la lecture du code.

On suppose qu'on dispose d'une fonction `CreerTableau(n)` *qui crée un tableau de taille* `n`*, indexé de* `0` *à* `n-1` *(les valeurs contenues dans le tableau initialement sont arbitraires [3]. L'instruction* `b=CreerTableau(n)` *créera/affectera un tableau de taille* `n` *dans la variable* `b`*. On pourra créer des tableaux de tableaux : exemple*

```
a = CreerTableau(p)
for i in range(p) :
....a[i] = CreerTableau(q)
```

On accède par `a[i][j]` *à la case d'indice j du tableau d'indice* `i` *dans le tableau ainsi créé.*

On souhaite stocker en mémoire une liste non ordonnées d'au plus n entiers sans redondance (aucun entier n'apparaît plusieurs fois). Nous nous proposons d'utiliser un tableau `liste` *de longueur $n+1$ tel que* `liste[0]` *contient le nombre d'éléments de la liste (comme en Pascal ?)*
`liste[i]` *contient le i^{eme} élément de la liste (non triée) pour $1 \leqslant i \leqslant liste[0]$ [4]. Il y a des exemples plus loin, profitez.*
Sinon, vous l'aurez compris, nulle part vous n'utiliserez **`append`**...

◐ **1** ◑ Écrivez une fonction **`CreerListeVide(n)`** qui crée, initialise et renvoie un tableau de longueur **`n+1`** correspondant à la liste « vide » ayant une capacité totale de n éléments.

◐ **2** ◑ Écrivez une fonction **`EstDansListe(liste, x)`** qui renvoie **`True`** si l'élément **`x`** apparaît dans la liste représentée par le tableau **`liste`**, et renvoie **`False`** sinon.
■ Quelle est la complexité de votre fonction dans le pire cas en fonction du nombre maximal n d'éléments de la liste.

◐ **3** ◑ Écrivez une procédure **`AjouteDansListe(liste, x)`** qui modifie de façon appropriée le tableau **`liste`** pour y ajouter **`x`** si x n'appartient pas déjà à la liste, et ne fait rien sinon.
■ Quel est le comportement de votre procédure si la liste est plein initialement (on ne demandera pas de traiter ce cas).
■ Quelle est la complexité en temps de votre procédure dans le pire cas en fonction du nombre maximal d'éléments de la liste ?

Un plan P est défini par un ensemble de n villes numérotées de 1 à n et un ensemble de m routes (toutes à double sens) reliant chacune deux villes ensemble. On dira que deux villes x et y sont voisines lorsqu'elles sont reliées par une route, ce que l'on notera $x \sim y$. On appellera chemin de longueur k toute suite de villes $v_1, v_2 \dots v_k$ telle que $v_1 \sim v_2 \sim \dots \sim v_k$. On représentera les villes d'un plan par des ronds contenant leur numéro et les routes par des traits reliant les villes voisines [5].
Un plan P est donc un tableau `plan` *où*

- `plan[0]` *contient un tableau à deux éléments :* `plan[0][0] = n` *(contient le nombre de villes du plan)*
  `plan[0][1] = m` *(contient le nombre de routes du plan)*
- *pour chaque ville x de $\{1, 2, \dots n\}$,* `plan[x]` *contient un tableau à n éléments représentant la liste à au plus $n-1$ éléments des villes voisines de la ville* `x` *(ordre de lecture sans importance).*

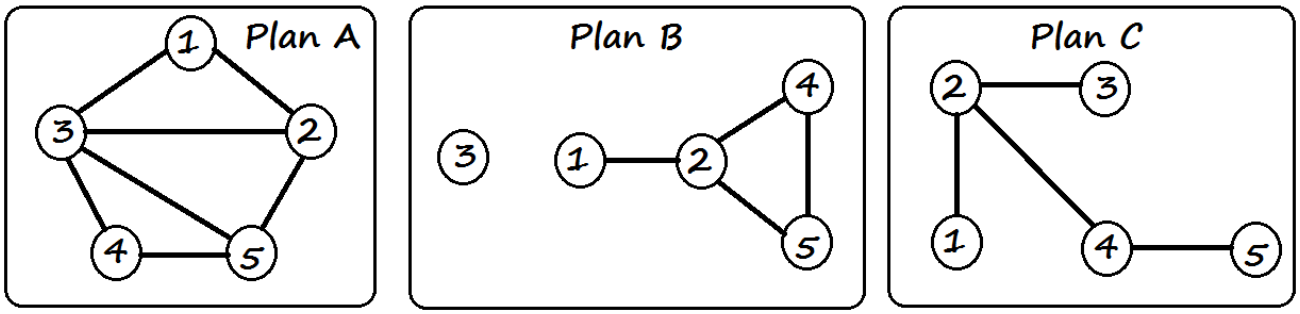
L'un des trois plans du dessin est représenté par le tableau ci contre

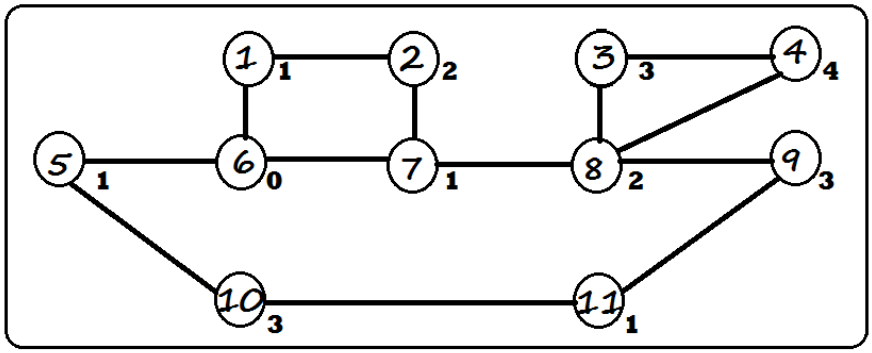
```
plan=[  [5,  4],
        [1,  2,  *,  *,  *  ],
        [3,  4,  1,  5,  *  ],
        [0,  *,  *,  *,  *  ],
        [2,  2,  5,  *,  *  ],
        [2,  4,  2,  *,  *  ]]
```

◐ **4** ◑ Donnez le tableau des deux autres.

---

2. *les fonctions renvoient une valeur (entier, réel, tableau, booléen...) ; les procédures modifient des variables globales, mais ne renvoient rien*
3. *avec Python, des* `None`*, qui n'occupent pas de place en mémoire, dans les notations du devoir, de simples* `*`
4. *ici pas de décalage pythonien, la case d'indice 0 est prise*
5. *comme si vous aviez voulu faire le contraire !*





Graphe coloré.
A côté de chaque ville est indiquée sa couleur.
On cherche donc ici un chemin de la ville [ ] à la ville [ ].

◐ 5 ◑ Écrivez une fonction `CreePlanSansRoute(n)` qui crée, remplit et renvoie le tableau de tableaux correspondant au plan à `n` villes n'ayant aucun route.

◐ 6 ◑ Écrivez une fonction `EstVoisine(plan, x, y)` qui renvoie `True` si les villes `x` et `y` sont voisines dans le plan codé par `plan`, et `False` sinon.
Vous avez évidemment le droit d'utiliser les procédures déjà créées.

◐ 7 ◑ Écrivez une procédure `Ajoute(plan, x, y)` qui modifie le tableau `plan` pour ajouter une route entre les deux villes `x` et `y` (supposées distinctes, oui) si elle n'était pas déjà présente, et ne fait rien sinon.
■ Y a-t-il un risque de dépassement de la capacité des listes ?

◐ 8 ◑ Écrivez une procédure `AfficheToutesLesRoutes(plan)` qui affiche la liste des routes codées par le tableau plan, chaque route ne devant être mentionnée qu'une fois. Par exemple, (1-2), (2-4), (2-5), (4-5) pour le plan indiqué et retenu plus haut.
■ Quelle est la complexité en temps de votre procédure dans le pire des cas ?

Recherche d'un chemin arc en ciel.

Étant données deux villes distinctes s et t, nous rechercherons un chemin de s à t, passant par exactement k villes toutes distinctes. L'objectif de cette partie est de la suivante est de construire une fonction qui va détecter en un temps linéaire en n.(n+m) l'existence d'un tel chemin avec une probabilité indépendante de la taille $n+m$ du plan.
Le principe de l'algorithme est d'attribuer à chaque ville de $\{1, 2, \dots n\} - \{s, t\}$ une couleur aléatoire codée par un entier aléatoire uniforme $couleur[i] \in \{1, \dots k\}$ stocké dans un tableau `couleur` de taille $n+1$ (la case 0 n'est pas utilisée). Les villes s et t reçoivent respectivement les couleurs spéciales 0 et $k+1$. L'objectif de cette partie est d'écrire une procédure qui détermine s'il existe un chemin de longueur $k+2$ allant de s à t dont la j^{ieme} ville intermédiaire a reçu la couleur j. dans l'exemple de la figure suivante, le chemin $6 \sim 7 \sim 8 \sim 3 \sim 4$ de longueur $5 = k+2$ qui relie $s = 6$ à $t = 4$ vérifie cette propriété pour les couleurs indiquées.
On suppose l'existence d'une fonction `EntierAleatoire(k)` qui renvoie un entier aléatoire uniforme entre 1 et k.
Bref P(EntierAleatoire(k)=c)=1/k pour tout c de 1 à k.

◐ 9 ◑ Écrivez une procédure `ColoriageAleatoire(plan, couleur, k, s, t)` qui prend en argument un `plan` de n villes, un tableau `couleur` de taille n+1, un entier `k`, et deux villes `s` et `t` de $\{1, 2, \dots n\}$ et remplit le tableau `couleur` avec une couleur aléatoire uniforme dans $\{1, \dots k\}$ choisie indépendamment pour chaque ville de $\{1, \dots n\} - \{s, t\}$ et les couleurs 0 et $k+1$ pour s et t respectivement.

Nous cherchons maintenant à écrire une fonction qui calcule l'ensemble des villes de couleur c voisines d'un ensemble de villes donné. dans l'exemple de la figure 2, l'ensemble des villes de couleur 2 voisines des villes $\{1, 5, 7\}$ est $\{2, 8\}$.

◐ **10** ◑ Écrivez une fonction `VoisinesDeCouleur(plan, couleur, i, c)` qui crée et renvoie le tableau codant la liste sans redondance des villes de couleur `c` voisines de la ville `i` dans le tableau `plan` colorié par le tableau `couleur`.

◐ **11** ◑ Écrivez une fonction `VoisinesDeLaListeDeCouleur(plan, couleur, liste, c)` qui crée et renvoie un tableau codant la liste sans redondance des villes de couleur `c` voisines d'une des villes présentes dans la liste sans redondances `liste` dans le plan `plan` colorié par la table `couleur`.
■ Quelle est la complexité de votre fonction dans le pire cas en fonction de n et m ?

◐ **12** ◑ Écrivez une fonction `ExisteCheminArcEnCiel(plan, couleur, k, s, t)` qui renvoie `True` si il existe dans le plan `plan` un chemin $s \sim v_1 \sim \ldots \sim v_k \sim t$ tel que `couleur[`v_j`]=j` pour tout j de $\{1, \ldots k\}$ et renvoie `False` sinon.
■ Quelle est la complexité de votre fonction dans le pire cas en fonction de k, n et m ?

Si les couleurs des villes sont choisies aléatoirement et uniformément dans $\{1, \ldots k\}$, la probabilité que j soit la couleur de la j^{ieme} ville d'une suite fixée de k villes vaut $1/k$, indépendamment pour tout j. Ainsi, étant données deux villes distinctes `s` *et* `t`*, s'il existe dans le plan* `plan` *un chemin de s à t passant par exactement k villes intermédiaires toutes distinctes et si le coloriage couleur est choisi aléatoirement conformément à la procédure* `ColoriageAleatoire(plan, couleur, k, s, t)`*, la procédure* `ExisteCheminArcEnCiel(plan, couleur, k, s, t)` *répond True avec probabilité au moins k^{-k} ; et répond toujours* `False` *sinon. Ainsi, si un tel chemin existe, la probabilité qu'une parmi k^k exécutions indépendantes de* `ExisteCheminArcEnCiel` *réponde* `True` *est supérieure ou égale à $1-(1-k^{-k})^{k^k} = 1-\exp\left(k^k.\ln(1-k^{-k})\right) \geqslant 1-\frac{1}{e} > 0$ (admis).*

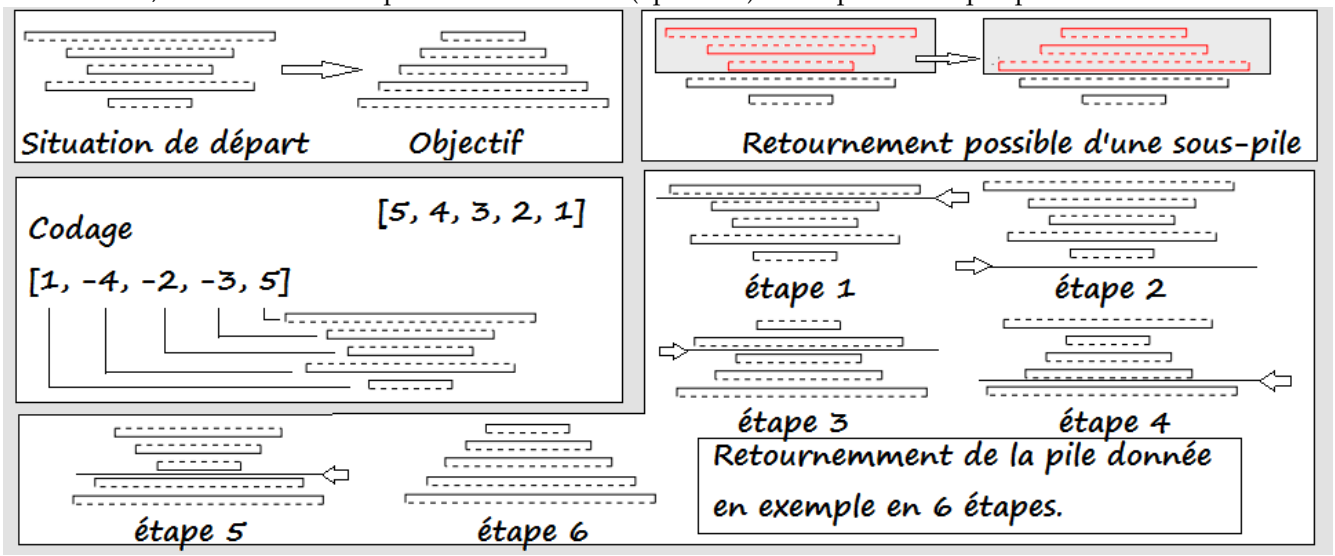
◐ **13** ◑ Écrivez une fonction `ExisteCheminSimple(plan, k, s, t)` qui renvoie `True` avec probabilité au moins $1-\frac{1}{e}$ s'il existe un chemin de `s` à `t` passant par exactement `k` villes intermédiaires toutes distinctes dans le plan `plan` et renvoie toujours `False` sinon.
■ Quelle est sa complexité en fonction de k, n et m dans le pire des cas ? Exprimez la sous la forme $O(f(k).g(n,m))$ pour f et g bien choisies.

◐ **14** ◑ Expliquez comment modifier votre programme pour renvoyer le chemin s'il est détecté avec succès.

◦33◦ On se propose d'étudier un algorithme de tri sur des listes, dit « algorithme du crêpier ». Oui, du marchand de crêpes. Le crêpier fabrique des crêpes qui ont des diamètres tous différents, et de plus chacune a deux faces : une face dorée, et une face blanche, pas assez cuite. Il les a posées dans le désordre, et chacune dans n'importe quel sens. Son objectif : une belle pile de crêpes, toutes orientées dans le même sens, face blanche vers le haut, de la plus grande à la plus petite.

Ce qu'il a le droit de faire : attraper le dessus du paquet de crêpes, sur une certaine épaisseur, puis retourner ce sous-paquet.
(les n crêpes de ce paquet sont alors changées de sens, et la sous-pile elle même est renversée).

Sur le dessin, on donne un exemple de retournement (optimal ?) d'une pile de cinq crêpes.



Une pile de crêpes sera codée par une liste d'entiers relatifs tous distincts en valeur absolue, de la crêpe du bas à la crêpe du haut. La valeur absolue de l'entier donne son diamètre, et le signe donne son orientation (deux bonnes raisons qu'il n'y ait pas de crêpe codée 0).
La pile finale devra donc être faite d'entiers positifs, du plus grand au plus petit.

I∼0) Donnez le codage de chacune des six piles de l'exemple représenté dans le cadre ci dessus.

I∼1) Écrivez une procédure `Melange` qui prend en entrée un entier n et retourne une pile de n crêpes, désordonnée.
Exemple : `Melange(5)` pourra donner `[1, -4, -2, -3, 5]`.
Rappel : le module `randrange` contient les fonctions `randrange`, `random` et `shuffle`.
`shuffle(L)` mélange une liste `L` : exemple `L = [3, 6, 4, 2, 0]`
`shuffle(L)`
`print(L)` : `[0, 3, 2, 4, 6]`
`randrange(a,b)` donne un entier entre `a` (inclus) et `b` (exclu)
`random()` donne un flottant au hasard entre `0` et `1`.
Un bonus si vous écrivez une procédure qui n'utilise pas `shuffle` mais juste des `randrange`.

I∼2) Écrivez une procédure `Test` qui prend en entrée une liste de crêpes `L` et retourne un booléen pour dire si la pile est triée, avec ses crêpes dans le bon sens.
Exemple : `Test( [6, 4, 3, 2, 1])` donne `True`,
`Test( [7, -5, 4, 3, 2, -1])` donne `False`,
`Test([7, 4, 5, 3, 2, 1])` donne `False`.

I∼3) Écrivez une procédure `Retourne` qui prend en entrée une liste `L` et un index `n` et retourne la pile de crêpes `L` retournée à partir de l'indice `n`.
Exemple : `Retourne([6, 7, -5, 4, 2, -3], 3)` donne `[6, 7, -5, 3, -2, -4]`.

Rappel : on peut couper une liste d'un indice `a` (inclus) à un indice `b` (exclu), de plus une option indique avec quel pas on avance dans la liste (par défaut, c'est bien sûr 1).
Exemple : `L = [1, 3, 5, 7, 8, 2, 6, 5, 9]`
`L[2 : 5]` donne `[5, 7, 8]`
`L[ : 4]` donne `[1, 3, 5, 7]`
`L[2 : 7 : 2]` donne `[5, 8, 6]`
`L[9 : 1 : -2]` donne `[9, 6, 8, 5]`
`L[1 : 9 : -2]` donne `[ ]`

I∼4) Que fait cette procédure :

```
def Scooby(L) :
....M, i = abs(L[0]), 0
....for k in range(len(L)) :
........if abs(L[k]) > M :
............M, i = L[k], k
....return i
```

Que retournera `Scooby([2, 5, -8, 7, 1, 3, 4, -6])` ?

	[-7, 2, 5, 1, 3, 4, -6]
Retourne(L, 0)	*[6, -4, -3, -1, -5, -2, 7]*
Retourne(L, 6)	*[6, -4, -3, -1, -5, -2, -7]*
Retourne(L,)	*[7, 2, 5, 1, 3, 4, -6]*
Retourne(L,)	*[7, 6, -4, -3, -1, -5, -2]*
Retourne(L, 5)	
Retourne(L,)	*[7, 6, -4, -3, -1, 2, -5]*
Retourne(L,)	*[7, 6, 5, , , ,]*
Retourne(L, 6)	*[7, 6, 5, , , ,]*
	[7, 6, 5, 4, 3, 2, 1]

Complétez l'algorithme à gauche.

Quelle est la pile de cinq crêpes qui sera la plus longue à retourner par algorithme du crêpier, et en combien de retournements ? (la plus courte est la pile déjà triée `[5, 4, 3, 2, 1]`).

Écrivez un script qui prend en entrée une liste `L` et la trie par algorithme du crêpier.