TP

Titre EXO

Source EXO

Savoirs et compétences :

- ....

algo

ALG-000.tex

Votre robot dispose de nombreux récepteurs et enregistre tous les signaux qui l'entourent. Cependant vous avez remarqué que certains de ces signaux sont très bruités. Vous décidez donc d'écrire un programme qui atténue le bruit de ces signaux, en effectuant ce que l'on appelle un lissage.

Une opération de lissage d'une séquence de mesures (des nombres décimaux) consiste à remplacer chaque mesure sauf la première et la dernière, par la moyenne des deux valeurs qui l'entourent.

Par exemple, si l'on part de la séquence de mesures suivantes : 1 3 4 5

On obtient après un lissage : 1 2.5 4 5

Le premier et dernier nombre sont inchangés. Le deuxième nombre est remplacé par la moyenne du 1er et du 3e, soit $(1+4)/2 = 2.5$, et le troisième est remplacé par $(3+5)/2 = 4$.

On peut ensuite repartir de cette nouvelle séquence, et refaire un nouveau lissage, puis un autre sur le résultat, etc. Votre programme doit calculer le nombre minimum de lissages successifs nécessaires pour s'assurer que la valeur absolue de la différence entre deux valeurs successives de la séquence finale obtenue ne dépasse jamais une valeur donnée, `diffMax`.

On vous garantit qu'il est toujours possible d'obtenir la propriété voulue en moins de 5000 lissages successifs.

On cherche à définir la fonction suivante `def lissage(t:list[float], diffmax:float) -> int` : où

- l'entrée est une liste `t` contenant les mesures, qui sont des flottants, et un flottant `diffmax`;
- la sortie est un entier qui correspond au nombre minimal de lissages nécessaire.

▪ **Exemple** `lissage([1.292, 1.343, 3.322, 4.789, -0.782, 7.313, 4.212], 1.120) -> 13.` ▪

ALG-001.tex

Il existe de nombreuses traditions étranges et amusantes sur Algoréa, la grande course de grenouilles annuelle en fait partie. Il faut savoir que les grenouilles algoréennes sont beaucoup plus intelligentes que les grenouilles terrestres et peuvent très bien être dressées pour participer à des courses. Chaque candidat a ainsi entraîné sa grenouille durement toute l'année pour ce grand événement.
La course se déroule en tours et, à chaque tour, une question est posée aux dresseurs. Le premier qui trouve la réponse gagne le droit d'ordonner à sa grenouille de faire un bond. Dans les règles de la course de grenouilles algoréennes, il est stipulé que c'est la grenouille qui restera le plus longtemps en tête qui remportera la victoire. Comme cette propriété est un peu difficile à vérifier, le jury demande votre aide.
Ce que doit faire votre programme :

`nbg` numérotées de 1 à `nbg` sont placées sur une ligne de départ. À chaque tour, on vous indique le numéro de la seule grenouille qui va sauter lors de ce tour, et la distance qu'elle va parcourir en direction de la ligne d'arrivée.
Écrivez un programme qui détermine laquelle des grenouilles a été strictement en tête de la course à la fin du plus grand nombre de tours.

ENTRÉE : deux entiers `nbg` et `nbt` et un tableau `t`.
`nbg` est le nombre de grenouilles participantes.
`nbt` est le nombre de tours de la course.
`t` est un tableau ayant `nbt` éléments, et tel que chaque élément est un couple : (numéro de la grenouille qui saute lors de ce tour, longueur de son saut).
SORTIE : vous devez renvoyer un entier : le numéro de la grenouille qui a été strictement en tête à la fin du plus grand nombre de tours. En cas d'égalité entre plusieurs grenouilles, choisissez celle dont le numéro est le plus petit.

EXEMPLE :
entrée : (4, 6, [[2,2], [1,2], [3,3], [4,1], [2,2], [3,1]])
sortie : 2.

ALG-002.tex

Un marchand de légumes très maniaque souhaite ranger ses petits pois en les regroupant en boîtes de telle sorte que chaque boîte contienne un nombre factoriel de petits pois. On rappelle qu'un nombre est factoriel s'il est de la forme 1, 1×2, $1\times 2\times 3$, $1\times 2\times 3\times 4\cdots$ et qu'on les note sous la forme suivante :

$$n! = 1 \times 2 \times 3 \times \cdots \times (n-1) \times n$$

Il souhaite également utiliser le plus petit nombre de boîtes possible.
Ainsi, s'il a 17 petits pois, il utilisera :

2 boîtes de $3! = 6$ petits pois
2 boîtes de $2! = 2$ petits pois
1 boîte de $1! = 1$ petits pois.

ce qui donne bien $2 \times 3! + 2 \times 2! + 1 \times 1! = 12 + 4 + 1 = 17$.

D'une manière générale, s'il a `nbp` petits pois, il doit trouver une suite `a_1, a_2⋯, a_p` d'entiers positifs ou nuls avec `a_p`> 0 et telle que : `nbp=a_1 x 1!+a_2 x 2!+⋯+a_p x p!` avec `a_1+⋯+a_p` minimal.
Remarque mathématique : si à chaque étape on cherche le plus grand entier `k` possible tel que `k!` soit inférieur au nombre de petits pois restant, on est sûrs d'obtenir la décomposition optimale : en termes informatiques, on dit que l'algorithme *glouton* est optimal.

ENTRÉE : un entier, `nbp`, le nombre total de petits poids.
SORTIE : un couple constitué de l'entier `p` et du tableau `[ a_1, a_2, ⋯, a_p ]`.

EXEMPLE :
entrée : 17
sortie : `(3, [ 1, 2, 2 ] )`.

ALG-003.tex

Rien de tel que de faire du camping pour profiter de la nature. Cependant sur Algoréa, les moustiques sont particulièrement pénibles et il faut faire attention à l'endroit où l'on s'installe, si l'on ne veut pas être sans cesse piqué.

Vous disposez d'une carte sur laquelle est indiquée, pour chaque parcelle de terrain, si le nombre de moustiques est supportable ou non. Votre objectif est de trouver le plus grand camping carré évitant les zones à moustiques qu'il est possible de construire.

ENTRÉE : un tableau `t` de n éléments correspondant à des lignes, chacune de ces lignes contenant p élements, qui ne sont que des 0 et des 1. 0 signifie qu'il n'y a pas de moustiques et 1 qu'il y a des moustiques.
SORTIE : un entier : la taille maximale du côté d'un carré ne comportant que des 0 et dont les bords sont parallèles aux axes.

EXEMPLE 1 :

entrée : `t=`
```
[ [1, 0, 0, 1, 0, 0, 1],
  [0, 0, 0, 0, 0, 0, 0],
  [1, 0, 0, 0, 0, 0, 0],
  [0, 0, 0, 0, 0, 0, 0],
  [0, 1, 0, 0, 0, 0, 1],
  [1, 0, 0, 0, 1, 0, 1] ]
```
sortie : 4.

EXEMPLE 2 :

entrée : `t=`
```
[ [0, 0, 0, 1, 1, 1, 1],
  [0, 0, 0, 1, 1, 1, 1],
  [0, 0, 0, 1, 1, 1, 1],
  [1, 1, 1, 1, 1, 0, 0],
  [1, 1, 1, 1, 1, 0, 0] ]
```
sortie : 3.

ALG-004.tex

On ne s'intéresse pas ici à la validité d'un nombre écrit en chiffre romains, mais à sa valeur. On rappelle quelques principes de base. Les sept caractères de la numération romaine sont :

I	V	X	L	C	D	M
1	5	10	50	100	500	1000

Certaines lettres sont dites d'*unité*. Ainsi on dit que I est une unité pour V et X, X est une unité pour L et C, C est une unité pour D et M.

Pour trouver la valeur d'un nombre écrit en chiffres romains, on s'appuie sur les règles suivantes :

- toute lettre placée à la droite d'une lettre dont valeur est supérieure ou égale à la sienne s'ajoute à celle-ci ;
- toute lettre d'unité placée immédiatement à la gauche d'une lettre plus forte qu'elle indique que le nombre qui lui correspond doit être retranché au nombre qui suit ;
- les valeurs sont groupées en ordre décroissant, sauf pour les valeurs à retrancher selon la règle précédente ; 1
- la même lettre ne peut pas être employée quatre fois consécutivement sauf M.

Par exemple, DXXXVI = 536, CIX = 109 et MCMXL = 1940.

1. Écrire une fonction `valeur (caractere)` qui retourne la valeur décimale d'un caractère romain. Cette fonction doit renvoyer 0 si le caractère n'est pas l'un des 7 chiffres romains.
2. Écrire la fonction principale `conversion (romain)` qui permet de convertir un nombre romain en nombre décimal. Cette fonction doit prendre en argument une chaîne de caractères `romain`. Si cette chaîne est écrite en majuscule et correspond à un nombre romain correctement écrit, la fonction doit renvoyer le nombre décimal égal au nombre romain passé en argument. Sinon, la fonction doit renvoyer -1.

ALG-005.tex

Pour tout $n \in \mathbb{N}\backslash\{0,1\}$, on pose $S_n = \sum_{k=2}^{n} \frac{(-1)^k}{k \ln k}$. On admet que la suite $(S_n)_{n\in\mathbb{N}\backslash\{0,1\}}$ a une limite finie ℓ en $+\infty$, et que pour tout $n \in \mathbb{N}\backslash\{0,1\}$, $u_{2n+1} \leq \ell \leq u_{2n}$.

1. Écrire un script Python donnant une valeur approchée de ℓ à 10^{-8} près.
2. Démontrer que le résultat donné par cet script est correct. On donnera clairement les éventuels invariants et variants des boucles intervenant dans le programme.

ALG-006.tex

On appelle *nombre parfait* tout entier naturel non nul qui est égal à la somme de ses diviseurs stricts, c'est-à-dire de ses diviseurs autres que lui-même.

Par exemple, 26 n'est pas parfait, car ses diviseurs stricts sont 1, 2 et 13, et $1+2+13=16\neq 28$. Mais 28 est parfait, car ses diviseurs stricts sont 1, 2, 4, 7 et 14, et $1+2+4+7+14=28$.

QUESTION Écrire une fonction Python `parfait(n)` prenant en entrée un entier naturel non nul `n`, et renvoyant un booléen donnant la valeur de vérité de l'assertion « `n` est parfait ».

QUESTION Écrire les éventuels variants et invariants permettant de montrer que cette fonction renvoie le bon résultat.

ALG-007.tex

QUESTION Écrire un script Python permettant de calculer le plus petit entier naturel n tel que $n!>123456789$.

QUESTION Écrire les éventuels variants et invariants de boucle permettant de montrer que le code Python écrit précédemment donne le bon résultat.

QUESTION Montrer que les variants et/ou invariants donnés à la question précédente sont bien des variants/invariants de boucle et justifier que le script écrit termine et donne le bon résultat.

ALG-008.tex

On s'intéresse au problème du codage d'une suite de bits (représentée sous la forme d'un tableau de 0 ou 1), de manière à pouvoir réparer une erreur de transmission. On fixe un entier naturel non nul `k`. Un tableau de bits `b` sera codé avec un niveau de redondance `k` en répétant chaque bit $2k+1$ fois. Pour décoder un tableau avec un niveau de redondance `k`, on le découpe en blocs de $2k+1$ bits. Dans chaque bloc, on effectue un « vote » et l'on considère la valeur majoritaire.

Exemple : Avec $k=2$ (et donc un niveau de redondance de 2), le tableau

$$b=[0,1,0]$$

sera codé en

$$c=[\underbrace{0,0,0,0,0}_{5\text{ bits}},\underbrace{1,1,1,1,1}_{5\text{ bits}},\underbrace{0,0,0,0,0}_{5\text{ bits}}].$$

Imaginons qu'après transmission, le tableau reçu soit

$$c'=[\underbrace{0,0,0,1,0}_{1\text{ erreur}},\underbrace{0,1,1,0,1}_{2\text{ erreurs}},\underbrace{0,1,0,1,1}_{3\text{ erreurs}}].$$

On le décode alors en

$$b'=[0,1,1].$$

QUESTION Écrire une fonction `code(b,k)` renvoyant le tableau codant un tableau `b`, avec un niveau de redondance `k`.

QUESTION Écrire une fonction `decode(c,k)` renvoyant le tableau décodant un tableau `c`, avec un niveau de redondance `k`.

ALG-009.tex

On considère un fichier `points.csv` contenant n lignes, chaque ligne contenant n entiers parmi 0 ou 1, séparés par des virgules. Ce fichier représente donc un tableau de nombres. Par exemple, le fichier

```
0,1,0,0
1,1,0,1
1,0,1,0
0,0,0,0
```

sera représenté (en `Python`) par le tableau bidimensionnel `t` (construit comme un tableau de tableaux) :

```
t = [ [0,1,0,0],
      [1,1,0,1],
      [1,0,1,0],
      [0,0,0,0] ]
```

QUESTION Écrire une fonction `lit_fichier(nom_de_f` qui, à un tel fichier `nom_de_fichier`, renvoie le tableau associé.

On voit ce tableau comme décrivant des points dans le plan. Étant donné un tel tableau `t`, on considère que l'on a un point aux coordonnées `(i,j)` si et seulement si `t[i][j]` vaut 1. Par exemple, avec le tableau précédents, la liste `L` des points décrits est

```
L = [ (0,1) , (1,0) , (1,1) , (1,3) , (2,0) , (2,2) ]
```

QUESTION Écrire une fonction `lit_tableau(t)` qui, à un tel tableau bidimensionnel `t`, renvoie la liste des points décrits.

QUESTION Complexité ou invariants ?

QUESTION Écrire une fonction `d(a,b)` qui, pour deux couples d'entiers `a`, `b`, dont nous noterons les coordonnées repectivement (x_a,y_a) et (x_b,y_b), renvoie la valeur $(x_a-x_b)^2+(y_a-y_b)^2$.

On veut maintenant, étant donné un entier naturel k non nul et un couple d'entiers `c`, trouver les k couples de la liste de points les plus proches de `c`. S'il y a égalité entre plusieurs points, on n'en garde que k.

Par exemple [...]

On considère le code suivant.

```
def kNN(L,c,k,d):
    """k plus proches voisins du point c dans L
       d : fonction de distance"""
    v = []
    for j in range(L) :
        a = L[j]
        if len(v) < k :
            v.append(a)
        if d(a,c) < d(v[-1],c) :
            v[-1] = a
        i = len(v) - 1
        while i >= 1 and v[i] < v[i-1] :
            v[i-1],v[i] = v[i],v[i-1]
            i = i-1
    return v
```

Cycle 01

QUESTION Donner l'invariant pour la boucle while et faire montrer que c'en est un.

QUESTION Donner l'invariant pour la boucle for.

QUESTION Complexité.

ALG-010.tex

On s'intéresse au problème d'insertion d'un nombre dans un tableau de nombres trié par ordre croissant.

Soit `t` $= [t_0, \dots, t_{n-1}]$ un tableau de nombres trié par ordre croissant, c'est-à-dire que

$$t_0 \leq t_1 \leq \cdots \leq t_{n-1}.$$

On dit qu'un nombre x s'insère en position $i \in [\![0, n-1[\![$ dans le tableau `t` si $t_i \leq x \leq t_{i+1}$. Si $x < t_0$, alors x s'insère en position -1 dans `t` et, si $x > t_{n-1}$, alors x s'insère en position $n-1$ dans `t`. 0.1

QUESTION Écrire une fonction `position_insertion(t,x)` prenant en argument un tableau de nombres `t` trié par ordre croissant et un nombre `x` et renvoyant une position où `x` s'insère dans `t`.

ALG-011.tex

Soit $n \geq 2$ un entier. Un diviseur strict de n est un entier $1 \leq d \leq n-1$ qui divise n (c'est-à-dire que le reste de la division euclidienne de n par d est nul). 0.2

Deux entiers n_1 et n_2 sont dits *amicaux* si la somme des diviseurs stricts de n_1 vaut n_2 et si la somme des diviseurs stricts de n_2 vaut n_1.

QUESTION Écrire une fonction `amicaux(n,m)` qui prend en argument deux entiers naturels `n` et `m` et renvoie la valeur de vérité de « `n` et `m` sont amicaux ».

On pourra écrire une fonction auxiliaire, au besoin.

chaines

STR-000.tex

Écrire une fonction prenant en paramètre deux chaînes de caractères `texte` et `mot`, et recherchant la première occurence de `mot` dans `texte` (la fonction retournera l'indice de la première lettre de l'occurrence de `mot` dans `texte`. 0.3

STR-001.tex

Une chaîne de caractères $s_0 s_1 \dots s_{n-1}$ est un *palindrome* si elle est « symétrique » : $\forall k \in \{0, \dots, n-1\}, s_k = s_{n-1-k}$.

QUESTION Écrire une fonction `est_pal(s)` qui, à une chaîne de caractères `s`, renvoie le booléen `True` si `s` est un palindrome et `False` sinon.

QUESTION Écrire une fonction `max_pal(s)` qui, à une chaîne de caractères `s`, renvoie la chaîne `p` où `p` est la plus grande sous chaîne palindromique centrée (c'est la plus grande sous-chaîne palindromique de la forme $s_k s_{k+1} \dots s_{n-1-k}$).

complexite

COM-000.tex

Dans ce problème, on considère des tableaux d'entiers relatifs `t`$=[$`t`$_0$`,t`$_1$`,...,t`$_{n-1}$`]`, et on appelle *tranche* de `t` toute sous-tableau non vide `[t`$_i$`,t`$_{i+1}$`,...,t`$_{j-1}$`]` d'entiers consécutifs de ce tableau (avec $0 \leq i < j \leq n$) qu'on notera désormais `t[i:j]`.

À toute tranche `t[i:j]` on associe la somme `s[i : j]` $= \sum_{k=i}^{j-1}$ `t_k` des éléments qui la composent. Le but de ce problème est de déterminer un algorithme efficace pour déterminer la valeur minimale des sommes des tranches de `t`.

L'algorithme naïf

1. Définir une fonction `somme` prenant en paramètre un tableau `t` et deux entiers i et j, et retournant la somme `s[i : j]`.
2. En déduire une fonction `tranche_min1` prenant en paramètre un tableau `t` et retournant la somme minimale d'une tranche de `t`.
3. Montrer que la complexité de cet algorithme est en $\Theta(n^3)$, c'est-à-dire qu'il existe deux constantes $a, b \in \mathbb{R}_+^*$ telles que si `t` a n éléments, alors le nombre d'opérations effectuées dans le calcul de `tranche_min1(t)` est compris entre an^3 et bn^3.

Un algorithme de coût quadratique

1. Définir, sans utiliser la fonction `somme`, une fonction `mintranche` prenant en paramètres un tableau `t` et un entier `i`, et calculant la valeur minimale de la somme d'une tranche de `t` dont le premier élément est `t`$_i$, en parcourant une seule fois la liste a à partir de l'indice `i`.
2. En déduire une fonction `tranche_min2` permettant de déterminer la somme minimale des tranches de `t`, en temps quadratique, c'est-à-dire que la complexité de cet algorithme est en $\Theta(n^2)$. On justifiera que la complexité est précisément en $\Theta(n^2)$.

Un algorithme de coût linéaire

Étant donnée un tableau `t`, on note m_i la somme minimale d'une tranche quelconque du tableau `t[0 : i]`, et c_i la somme minimale d'une tranche de `t[0 : i]` se terminant par `t`$_{i-1}$.

Montrer que $c_{i+1} = \min(c_i + t_i, t_i)$ et $m_{i+1} = \min(m_i, c_{i+1})$, et en déduire une fonction `tranche_min3` de coût linéaire (c'est-à-dire dont la complexité est en $\Theta(n)$), calculant la somme minimale d'une tranche de `t`.

COM-001.tex

Un enjeu scientifique et technologique actuel est de savoir traiter des problèmes mettant en jeu un nombre très important de données. Un point crucial est souvent de pouvoir manipuler des matrices de très grandes dimensions, ce qui est *a priori* très coûteux, en temps de

calcul et en mémoire. On peut cependant souvent considérer que les matrices manipulées ne contiennent que « peu » d'éléments non nuls : c'est ce que l'on appelle les matrices *creuses*.

Nous nous intéressons ici à l'implémentation de deux algorithmes d'addition de matrices : l'un pour une représentation classique des matrices, l'autre pour une représentation des matrices creuses.

Soit $n \in \mathbb{N}^*$, on note $\mathcal{M}_n(\mathbb{R})$ l'ensemble des matrices carrées d'ordre n, à coefficients dans $\mathbb{R}$.

On représentera classiquement une matrice $M \in \mathcal{M}_n(\mathbb{R})$ par un tableau à double entrées. En Python, cela sera un tableau (type `list`) de longueur n, chaque élément de ce tableau représentant une ligne de M. Chaque élément de ce tableau est donc un tableau de longueur n, dont tous les éléments sont des nombres (types `int` ou `float`).

Cette même matrice M sera représentée de manière creuse en ne décrivant que ses cases non vides par un tableau de triplets `(i,j,x)`, où `x` est l'élément de M situé sur la i^e ligne et la j^e colonne. On pourra supposer que les éléments non nuls de M sont ainsi décrits ligne par ligne.

▪ **Exemple** La matrice $\begin{pmatrix} 1 & 0 & 5 \\ 0 & -2 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ sera représentée classiquement par le tableau

```
[ [1,0,5] , [0,-2,0] , [0,0,0] ]
```

et de manière creuse par le tableau

```
[ (0,0,1) , (0,2,5) , (1,1,-2) ].
```

▪

QUESTION Écrire une fonction `add(M,N)` prenant en argument deux représentations classiques `M` et `N` de deux matrices M et N (carrées, de même ordre) et renvoyant la représentation classique de $M+N$. On prendra soin d'écrire une fonction « optimale » en terme de complexité, spatiale et temporelle.

QUESTION Écrire une fonction `add_creuse(M,N)` prenant en argument deux représentations creuses `M` et `N` de deux matrices M et N (carrées, de même ordre) et renvoyant la représentation creuse de $M+N$. On prendra soin d'écrire une fonction « optimale » en terme de complexité, spatiale et temporelle.

QUESTION On suppose que M et N sont carrées, d'ordre n, représentées classiquement par `M` et `N`. Évaluer asymptotiquement la complexité temporelle de la fonction `add(M,N)`.

QUESTION On suppose que M et N sont carrées et contient chacune au plus p élements non nuls, représentées de manière creuse par `M` et `N`. Évaluer asymptotiquement la complexité temporelle de la fonction `add_creuse(M,N)`.

Pour simplifier, on ne justifiera pas que les éléments obtenus sont disposés dans le bon ordre.

QUESTION Discuter du choix de la représentation pertinente à utiliser pour additionner deux matrices.

COM-002.tex

On considère la suite u à valeurs dans $[\![0;64\,007[\![$ définie par

$$u_0 = 42, \quad \forall n \in \mathbb{N},\ u_{n+1} = 15\,091\,u_n \ [64\,007],$$

ainsi que, pour tout $n \in \mathbb{N}$, $S_n = \sum_{k=0}^{n} u_k$.

On propose l'algorithme suivant pour calculer les valeurs de S.

```
def u(n):
    """u_n, n : entier naturel"""
    v = 42
    # Inv : v = u_0
    for k in range(n):
        # Inv : v = u_k
        v = 15091 * v % 64007
        # Inv : v = 15091*u_k % 64007 = u_k+1
    # Inv : au dernier tour, k = n-1, donc v =
    return v

def S(n):
    """u_n, n : entier naturel"""
    s = u(0)
    # Inv : s = S_0
    for k in range(n):
        # Inv : s = S_k
        s = s + u(k+1)
        # Inv : v = S_k+u_k+1 = S_k+1
    # Inv : au dernier tour, k = n-1, donc s = S_n
    return s
```

1. Étudier les complexités des fonctions `u` et `S`, en fonction de n.
2. Écrire une fonction donnant la valeur de S_n en temps $O(n)$.

COM-003.tex

On considère la suite u définie par

$$u_0 = 2, \quad \forall n \in \mathbb{N},\ u_{n+1} = u_n^2.$$

On se considère la fonction suivante, permettant de calculer les valeurs de u.

```
def u(n):
    """u_n, n : entier naturel"""
    v = 2
    # Inv : v = u_0
    for k in range(n):
        # Inv : v = u_k
        v = v*v
        # Inv : v = u_k**2 = u_k+1
    # Inv : au dernier tour, k = n-1, donc v = u_n
    return v
```

Pour étudier le temps d'exécution d'une fonction, on pourra utiliser le morceau de code suivant.

```
import timeit

REPEAT=3
```

Cycle 01

```
def duree(f, x):
  """Calcule le temps mis par Python pour calculer f(x)
  Cette fonction effectue en fait le calcul de f(x) REPEAT fois
  et garde la valeur la plus petite
  (l'idée est d'éliminer les éventuelles perturbations provoquées
  par d'autres processus tournant sur la machine)"""
  t = timeit.Timer(stmt=lambda : f(x))
  time = min(t.repeat(REPEAT, number=1))
  return time
```

1. Étudier en fonction de n la complexité asymptotique de la fonction `u`, dans le modèle standard.
2. Tracer les temps de calculs de u_k pour $k \in [\![0;30[\![$ par la fonction `u`. Discuter le résultat.
   Indication : on pourra utiliser une échelle semi-logarithmique.
3. Proposer un modèle de complexité plus réaliste et étudier dans ce modèle n la complexité asymptotique de la fonction `u`. On pourra déterminer explicitement u_n.

COM-004.tex

On considère la fonction `Python` suivante.

```
def mystere(t):
    """t : tableau d'entiers"""
    m = 0
    for i in range(len(t)):
        for j in range(i+1,len(t)):
            if t[j] - t [i] > m :
                m = t[j]-t[i]
    return m
```

QUESTION Que contient le résultat renvoyé par `mystere(t)` ? Justifier.

QUESTION Quelle est la complexité de la fonction `mystere(t)`, en fonction de la longueur de `t`, que l'on notera n ?

COM-005.tex

On considère la fonction `Python` suivante.

```
def mystere(u,v):
    """u,v : tableaux d'entiers"""
    m = 0
    p,q = len(u),len(v)
    for i in range(p) :
        for a in range(q) :
            k = 0
            while i+k < p and a+k < q and u[i+k] == v[a+k] :
                k = k+1
            if k > m :
                m = k
    return m
```

QUESTION Que contient le résultat renvoyé par `mystere(u,v)` ? Justifier.

QUESTION Quelle est la complexité de la fonction `mystere(u,v)`, en fonction du maximum des longueurs de `u` et `v`, que l'on notera n ?

COM-006.tex

On considère le code suivant, qui crée une liste aléatoire `L` et qui la trie ensuite par ordre croissant par la méthode du «tri par insertion».

```
from random import randrange

def liste_triee(n):
    """Renvoie une liste d'éléments de range(100), de
       triée par ordre croissant """
    L = []
    for i in range(n):
        L.append(randrange(100))
    for i in range(1,n):
        # Inv : L[:i] est triée par ordre croissant
        # Idée : on fait redescendre L[i] pour l'insé
        j = i
        while j >= 1 and L[j] < L[j-1]:
            # On échange L[j-1] et L[j]
            L[j], L[j-1] = L[j-1], L[j]
            j = j-1
    return L
```

Un appel de `randrange(100)` renvoie un nombre tiré aléatoirement et uniformément dans $[\![0,100[\![$ et a une complexité en $O(1)$.

QUESTION Peut-on donner explicitement et exactement une complexité pour la boucle `while` des lignes 13 à 16 ? Que peut-on quand même proposer comme type de complexité pour cette boucle ?

Donner dans ce cas la complexité de cette boucle en fonction de `i`.

QUESTION Donner dans ce cas la complexité d'un appel de `liste_triee(n)` en fonction de `n`.

COM-007.tex

Beaucoup de problèmes technologiques actuels mettent en jeu des données de grandes dimensions et font souvent intervenir de grandes matrices.

Beaucoup de ces matrices sont *creuses*, c'est-à-dire qu'elles ne contiennent que peu de valeurs non nulles. Nous allons développer une méthode de codage de telles matrices et étudier leur intérêt.

Dans tout ce problème, on s'interdira d'utiliser les opérations entre vecteurs et matrices `numpy` (méthode `.dot()` par exemple).

Dans tout ce problème, on utilisera le type `array` de `numpy` pour représenter des vecteurs. On pourra supposer que la ligne suivante a été écrite :

```
from numpy import array, zeros
```

Dans tout ce problème, on veillera à l'optimalité des fonctions écrites en termes de complexité temporelle asymptotique. Les réponses clairement sous-optimales seront pénalisées.

Soit une matrice réelle $A \in \mathcal{M}_{n,p}(\mathbb{R})$ de dimension $n \times p$ et possédant s coefficients non nuls. Ce coefficient s est appelé *niveau de remplissage* de la matrice A. Comme toujours en Python, on numérotera les lignes et les colonnes en partant de 0.

Une telle matrice se code usuellement comme un tableau de nombres de dimension $n \times p$.

Le codage CSR (Compress Sparse Raw) code la matrice A par trois listes V, L et C définies comme suit.

- La liste V contient toutes les valeurs des coefficients non nuls de A, en les listant ligne par ligne (de la première à la dernière ligne puis de la première à la dernière colonne). Ainsi, V est de longueur s.
- La liste L est de longueur $n+1$, L_0 vaut toujours 0 et pour chaque $1 \leq i \leq n$, L_i vaut le nombre de coefficients non nuls dans les i premières lignes de A (*i.e.* de la ligne d'indice 0 à celle d'indice $i-1$, inclu).
- La liste C contient les numéros de colonne de chaque coefficients non nuls de A, en les listant ligne par ligne (de la première à la dernière ligne puis de la première à la dernière colonne). Ainsi, C est de longueur s.

Par exemple, avec

$$A = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 2 & 0 & 4 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix},$$

on a $n=3$, $p=4$, $s=4$ et sa représentation CSR est donnée par les trois listes

$$V = [-1,2,4,-1],$$
$$L = [0,1,3,4],$$
$$C = [1,0,2,3].$$

On remarquera qu'ajouter une colonne nulle à droite de A ne change pas sa représentation CSR.

QUESTION Déterminer la représentation CSR de la matrice

$$A = \begin{pmatrix} 1 & 0 & 3 \\ 0 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

QUESTION Donner une matrice dont la représentation CSR est

$$V = [2,-3,1,1],$$
$$L = [0,0,2,4],$$
$$C = [0,3,2,3].$$

QUESTION Soit (V,L,C) la représentation CSR de A, soit $0 \leq i \leq n-1$. Combien y a-t-il de coefficients non nuls sur la ligne n° i de A? Donner deux expressions Python (en fonction de V, L, C et i) permettant d'obtenir respectivement la liste des valeurs de ces coefficients et la liste des indices colonnes de ces coefficients.

On rappelle la formule du produit matriciel : pour une matrice $A \in \mathcal{M}_{n,p}(\mathbb{R})$ de coefficients $a_{i,j}$ et un vecteur $X \in \mathcal{M}_{p,1}(\mathbb{R})$ de coefficients x_j, si $0 \leq i \leq n-1$, alors la i^{e} coordonnée de AX est

$$\sum_{j=0}^{p-1} a_{i,j} x_j.$$

QUESTION Écrire une fonction `coeff_prod(V,L,C,X,i)` renvoyant la i^{e} coordonnée du produit AX, où le triplet (V,L,C) est la représentation CSR de A. On supposera que les dimension de A et X sont compatibles pour effectuer le produit AX.

Donner la complexité asymptotique de cette fonction, en fonction de n, p et ℓ_i, où ℓ_i est le nombre d'éléments non nuls sur la i^{e} ligne de A.

QUESTION Écrire une fonction `prod(V,L,C,X)` renvoyant le produit AX, où le triplet (V,L,C) est la représentation CSR de A. On supposera que les dimension de A et X sont compatibles pour effectuer le produit AX.

Donner la complexité asymptotique de cette fonction, en fonction de n, p et s.

QUESTION Écrire une fonction `prod_naif(A,X)` renvoyant le produit AX, où A est codée usuellement comme un tableau à double dimension.

Donner la complexité asymptotique de cette fonction, en fonction de n, et p.

QUESTION En les comparant, discuter des deux complexités précédentes, notamment en fonction du niveau de remplissage s.

On prend maintenant pour exemple celui de la dérivation discrète, typique en traitement du signal. Pour un vecteur de longueur n

$$X = \begin{pmatrix} x_0 \\ \vdots \\ x_{n-1} \end{pmatrix}$$

on définit la dérivée discrète de X comme le vecteur Y de longueur n définit par :

$$y_0 = x_1 - x_0,\ y_{n-1} = x_{n-1} - x_{n-2} \text{ et } \forall i \in [\![1, n-1[\![,\ y_i = \frac{1}{2}(x_{i+1} - x_{i-1}).$$

QUESTION Déterminer une matrice A vérifiant $Y = AX$.

QUESTION Écrire une fonction `CSR_A(n)` renvoyant les trois vecteurs `V`, `L` et `C` codant A au format CSR.

Indication : on pourra écrire une fonction pour la création de chaque vecteur

equadiffs

EQD-000.tex

On considère une équation différentielle $(\mathcal{E})$: $y' = F(y,t)$, où F est une fonction de $\mathbb{R}^n \times \mathbb{R}$ dans $\mathbb{R}^n$, et où l'inconnue y est une fonction de $\mathcal{C}^1(\mathbb{R},\mathbb{R}^n)$, avec la condition initiale $y(t_0) = y_0$.

QUESTION Décrire le principe de la méthode d'Euler. On donnera clairement la relation de récurrence qui est au coeur de cette méthode, en expliquant bien ce que représente la suite vérifiant cette relation de récurrence.

QUESTION Écrire en Python une fonction mettant en œuvre la méthode d'Euler et permettant de résoudre numériquement $(\mathcal{E})$ sur le segment $[a,b]$, où $a,b \in \mathbb{R}$, $a < b$.

Vous écrirez une docstring décrivant tous les arguments de cette fonction.

On considère l'équation différentielle d'inconnue $y \in \mathscr{C}^2(\mathbb{R})$ et les conditions initiales suivantes :

$$y'' + \cos(t)y' - t^2 y = \mathrm{e}^t, \qquad y(0) = 4,\ y'(0) = 2. \quad (\mathscr{E})$$

QUESTION Donnez une fonction F et une variable X telle que (**??**) soit équivalente à l'équation $X' = F(X,t)$. On précisera bien les ensembles de définition et d'arrivée de F, et l'ensemble auquel appartient la variable X, ainsi que la condition initiale à utiliser.

EQD-001.tex

1 Autour de la dynamique gravitationnelle

1.1 Présentation

Modéliser les interactions physiques entre un grand nombre de constituants mène à l'écriture de systèmes différentiels pour lesquels, en dehors de quelques situations particulières, il n'existe aucune solution analytique. Les problèmes de dynamique gravitationnelle et de dynamique moléculaire en sont deux exemples. Afin d'analyser le comportement temporel de tels systèmes, l'informatique peut apporter une aide substantielle en permettant leur simulation numérique. L'objet de ce sujet est l'étude de solutions algorithmiques en vue de simuler une dynamique gravitationnelle afin, par exemple, de prédire une éclipse ou le passage d'une comète.

1.2 Quelques fonctions utilitaires

QUESTIONDonner la valeur des expressions python suivantes :

- $[1,2,3]+[4,5,6]$
- $2*[1,2,3]$

QUESTION Écrire une fonction python $smul$ à deux paramètres, un nombre et une liste de nombres, qui multiple chaque élément de la liste par le nombre et renvoie une deuxième liste sans modifier la première. Par exemple, *smul(2, [1, 2, 3])* renvera *[2, 4, 6]*.

QUESTION Déterminer la complexité de cet algorithme en fonction de la taille de la liste.

2 Étude de schémas numériques

Soient y une fonction de classe $\mathscr{C}^2$ sur $\mathbb{R}$ et t_{min} et t_{max} deux réels tels que $t_{min} < t_{max}$. On note I l'intervalle $[t_{min}, t_{max}]$. On s'intéresse à une équation différentielle du second ordre de la forme :

$$\forall t \in I \qquad y''(t) = f(y(t)), \qquad (1)$$

où f est une fonction donnée, continue sur $\mathbb{R}$. De nombreux systèmes physiques peuvent être décrits par une équation de ce type.

On suppose connues les valeurs $y_0 = y(t_{min})$ et $z_0 = y'(t_{min})$.

2.1 Mise en forme du problème

Pour résoudre numériquement l'équation différentielle (**??**), on introduit la fonction $z : I \to \mathbb{R}$ définie par

$$\forall t \in I, z(t) = y'(t). \qquad (2)$$

On considère l'équation :

$$Y' = F(Y,t) \qquad (3)$$

QUESTION Pour quelle variable Y et quelle fonction F, l'équation (**??**) peut se mettre sous la forme de l'équation (**??**) ?

Soit n un entier strictement supérieur à 1 et $J_n = [\![0, n-1]\!]$. On pose $h = \dfrac{t_{max} - t_{min}}{n-1}$ et $\forall i \in J_n,\ t_i = t_{min} + i \cdot h$. On peut montrer que, pour tout entier $i \in [\![0, n-2]\!]$,

$$Y(t_{i+1}) = Y(t_i) + \int_{t_i}^{t_{i+1}} Y'(t)\,dt \qquad (4)$$

La suite du problème exploite les notations introduites dans cette partie et présente deux méthodes numériques dans lesquelles l'intégrale précédente est remplacée par une valeur approchée.

2.2 Schéma d'Euler explicite

Dans le schéma d'Euler explicite, chaque terme sous le signe intégrale est remplacé par sa valeur prise en la borne inférieure.

QUESTION Ecrire une fonction $euler(F, tmin, tmax, Y_0, n)$ qui renvoie deux listes de nombres correspondant aux valeurs associées aux suites $(y_i)_{i\in J_n}$, $(z_i)_{i\in J_n}$ ainsi que la liste du temps $(t_i)_{i\in J_n}$.

Pour illustrer cette méthode, on considère l'équation différentielle,

$$\forall t \in I, \quad y''(t) = -\omega^2 y(t) \qquad (5)$$

dans laquelle ω est un nombre réel.

QUESTION Expliciter en langage Python la fonction $F(Y,t)$ et la fonction $f(y)$ qui permettront de mettre en oeuvre ce problème.

La mise en oeuvre de la méthode d'Euler explicite génère le résultat graphique donné figure **??**. Dans un système d'unités adapté, les calculs ont été menés en prenant $y_0 = 3$, $z_0 = 0$, $t_{min} = 0$, $t_{max} = 3$, $\omega = 2\pi$ et $n = 100$.

QUESTION Écrire la suite d'instructions permettant à partir de la fonction F et après avoir défini la fonction $euler$ de la mettre en oeuvre et produire le tracé de la figure **??**.

2.3 Schéma de Verlet

Le physicien français Loup Verlet a proposé en 1967 un schéma numérique d'intégration d'une équation de la forme (**??**) dans lequel, en notant $f_i = f(y_i)$ et $f_{i+1} = f(y_{i+1})$, les relations de récurrence s'écrivent, pour tout entier $i \in [\![0, n-2]\!]$:

$$y_{i+1} = y_i + hz_i + \frac{h^2}{2} f_i \quad \text{et} \quad z_{i+1} = z_i + \frac{h}{2}(f_i + f_{i+1}). \tag{6}$$

QUESTION Écrire une fonction $verlet(f, tmin, tmax, Y_0, n)$ qui renvoie deux listes de nombres correspondant aux valeurs associées aux suites $(y_i)_{i\in J_n}$ et $(z_i)_{i\in J_n}$, ainsi que la liste du temps $(t_i)_{i\in J_n}$.

On reprend l'exemple de l'oscillateur harmonique défini par l'équation (**??**) et on compare les résultats obtenus à l'aide des schémas d'Euler et de Verlet.

QUESTION Écrire la suite d'instructions permettant à partir de la fonction f et après avoir défini la fonction $verlet$ de la mettre en oeuvre et produire le tracé de la figure **??**.

2.4 Comparaison qualitative des schémas numériques.

On rappelle que l'énergie pour un tel oscillateur est proportionnelle à :

$$\mathscr{E} = \frac{1}{2}(y')^2 + \frac{\omega^2}{2} y^2.$$

QUESTION Conclure sur la stabilité des deux schémas numériques.

EQD-002.tex

3 Etude d'un trafic routier

Ce sujet concerne la conception d'un logiciel d'étude de trafic routier. On modélise le déplacement d'un ensemble de voitures sur des files à sens unique (voir Figures **??** et **??**). C'est un schéma simple qui peut permettre de comprendre l'apparition d'embouteillages et de concevoir des solutions pour fluidifier le trafic.

3.1 Préliminaires

Dans un premier temps, on considère le cas d'une seule file, illustré par la Figure **??**. Une file de longueur n est représentée par n cases. Une case peut contenir au plus une voiture. Les voitures présentes dans une file circulent toutes dans la même direction (sens des indices croissants, désigné par les flèches sur la Figure **??**) et sont indifférenciées.

QUESTION Expliquer comment représenter une file de voitures à l'aide d'une liste de booléens.

QUESTION Donner une ou plusieurs instructions `Python` permettant de définir une liste A représentant la file de voitures illustrée par la Figure **??**.

QUESTION Soit L une liste représentant une file de longueur n et i un entier tel que $0 \leqslant i < n$. Définir en Python la fonction `occupe(L, i)` qui renvoie `True` lorsque la case d'indice i de la file est occupée par une voiture et `False` sinon.

QUESTION Combien existe-t-il de files différentes de longueur n ? Justifier votre réponse.

QUESTION Écrire une fonction `egal(L1, L2)` retournant un booléen permettant de savoir si deux listes $L1$ et $L2$ sont égales.

QUESTION Que peut-on dire de la complexité de cette fonction ?

3.2 Base de données

On modélise ici un réseau routier par un ensemble de croisements et de voies reliant ces croisements. Les voies partent d'un croisement et arrivent à un autre croisement. Ainsi, pour modéliser une route à double sens, on utilise deux voies circulant en sens opposés. La base de données du réseau routier est constituée des relations suivantes :

- Croisement(id, longitude, latitude)
- Voie(id, longueur, id_croisement_debut, id_croisement_fin)

Dans la suite on considère c l'identifiant (id) d'un croisement donné.

QUESTION Écrire la requête SQL qui renvoie les identifiants des croisements atteignables en utilisant une seule voie à partir du croisement ayant l'identifiant c.

QUESTION Écrire la requête SQL qui renvoie les longitudes et latitudes des croisements atteignables en utilisant une seule voie, à partir du croisement c.

QUESTION Que renvoie la requête SQL suivante ?

3.3 Simulation dynamique

On introduit les bibliothèques `Math` et `NumPy` à l'aide des lignes suivantes :

```
import math as m
import numpy as np
```

On s'intéresse maintenant à l'apparition des voitures en début de file. On se place alors dans le cas d'un parking lors d'une sortie d'usine entre 17 h et 19 h. Afin de simplifier le problème on considère qu'à $t = 0$, il est 17 h et à $t = 1$, il est 19 h.

Tous les employés de l'usine ne sortent pas au même moment. On donne sur la Figure **??** l'évolution de $Q(t)$ qui représente le nombre de véhicules quittant le stationnement par unité de temps, pour $t \in [0,1]$. On considère que cette fonction est nulle en dehors de cet intervalle. On appelle `db_stat_max` le maximum de Q entre 0 et 1.

$$\forall t \in [0,1],\ Q(t) = \texttt{db\_stat\_max} \cdot \mathrm{e}^{\left(1 - \frac{1}{4t(1-t)}\right)}$$

QUESTION Écrire une fonction `Q(t)` permettant d'obtenir, pour tout réel t, le nombre de véhicules quittant le stationnement par unité de temps. On considère `db_stat_max` comme une variable globale. Faire attention aux cas où `t = 0` et `t = 1`.

Cycle 01

QUESTION Écrire une fonction `integrale(f, a, b, n)` permettant, avec la méthode des trapèzes, d'estimer l'intégrale de `f` entre `a` et `b` à partir de `n` points équirépartis.

QUESTION On appelle $N(t)$ le nombre de véhicules ayant quitté le stationnement jusqu'à l'instant t. Écrire une fonction `Nb(t, n)` renvoyant une valeur approchée de ce nombre, en utilisant n points.

La suite de cette partie permettant d'obtenir $N(t)$ par une autre méthode, la fonction `Nb` ne sera plus utilisée.

La vitesse de sortie des véhicules est limitée par le nombre de véhicules sortants. En effet, quand peu de véhicules sortent du parking, ces derniers peuvent circuler à vitesse maximale V_{max}. En revanche, lorsque beaucoup de véhicules sont en mouvement, ces derniers se ralentissent les uns les autres. On appelle K la concentration de véhicules cherchant à rejoindre la sortie du parking à un instant donné et K_{sat} le nombre de véhicules à partir duquel la vitesse de sortie est minimale (V_{min}).

L'expression de la vitesse des véhicules V en fonction de $K \in [0, K_{sat}]$ est donnée par

$$V(K) = \frac{V_{max} - V_{min}}{2} \cdot \left(1 + \cos\left(\pi \cdot \frac{K}{K_{sat}}\right)\right) + V_{min}.$$

QUESTION Écrire une fonction `V(K)` prenant en entrée un flottant `K` et renvoyant la valeur de la vitesse correspondante. On considère `Vmin`, `Vmax` et `Ksat` comme des variables globales.

La conservation du nombre total de véhicules conduit au système différentiel suivant :

$$\frac{\mathrm{d}N}{\mathrm{d}t}(t) = Q(t) \tag{7}$$
$$\frac{\mathrm{d}K}{\mathrm{d}t}(t) = Q(t) - K(t) \cdot V(K) \tag{8}$$
$$\frac{\mathrm{d}S}{\mathrm{d}t}(t) = K(t) \cdot V(K) \tag{9}$$

où $S(t)$ désigne le nombre de véhicules sortis du parking à l'instant t.

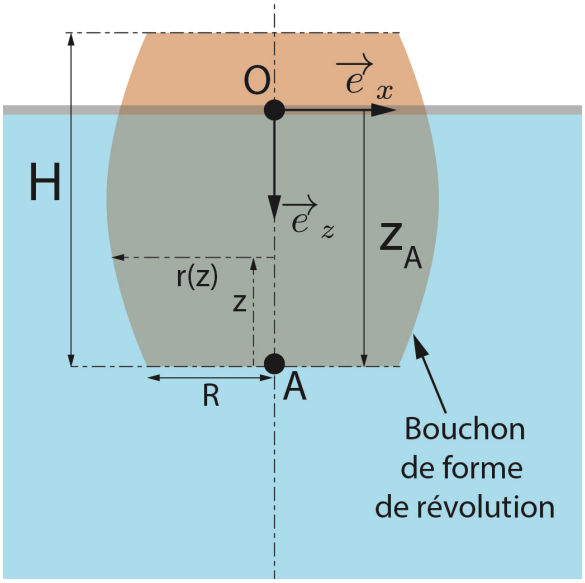
QUESTION Rappeler la relation de récurrence de la méthode d'Euler explicite pour un problème défini par son équation différentielle $y'(t) = F(y(t), t)$ avec $h = \Delta t$ le pas de temps supposé constant.

On considère que cette méthode est implémentée dans une fonction `odeint(F, Y0, T)` où `F` est la fonction associée au problème de Cauchy, `Y0` est un tableau `NumPy` contenant les valeurs initiales de la fonction Y et `T` un tableau `NumPy` contenant les différents pas de temps.

QUESTION Commenter en quelques lignes les allures des courbes de la Figure **??**.

EQD-003.tex

Ce sujet concerne la dynamique d'un bouchon en liège flottant dans un verre d'eau (voir figure suivante).



On note :

- R le rayon de la base du bouchon ;
- ρ_e la masse volumique de l'eau ;
- ρ_b la masse volumique du bouchon ;
- $z_A(t)$ la position verticale du bas du bouchon selon la direction $\vec{e}_z$ qui dépend du temps t par rapport à O (point situé sur la surface de l'eau) ;
- H la hauteur du bouchon.

Le bouchon possède une symétrie de révolution. Ainsi, son rayon, noté $r(z)$, dépend de la coordonnée z de la manière suivante :

$$\begin{array}{rccl} r : & [0,H] & \to & \mathbb{R}, \\ & z & \mapsto & R \cdot \left[1 + 0,1 \cdot \sin\left(\pi \cdot \dfrac{z}{H}\right)\right]. \end{array}$$

Le volume immergé du bouchon dépend de la position z_A et est donné par la relation suivante :

$$V_i(z_A) = \begin{cases} 0 & \text{si} \quad z_A < 0 ; \\ \pi \cdot \displaystyle\int_{z=0}^{z_A} (r(z))^2 \,\mathrm{d}z & \text{si} \quad 0 \le z_A \le H ; \\ \pi \cdot \displaystyle\int_{z=0}^{H} (r(z))^2 \,\mathrm{d}z & \text{si} \quad z_A > H. \end{cases}$$

On note V le volume total du bouchon correspondant à $V_i(H)$.

Le théorème de la résultante dynamique appliqué au bouchon selon la direction $\vec{e}_z$ donne :

$$M \cdot \frac{\mathrm{d}^2 z_A}{\mathrm{d}t^2}(t) = P - F_p(z_A(t)) + F_v\left(z_A(t), \frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right). \quad (\mathscr{E})$$

Les grandeurs suivantes interviennent dans l'équation (**??**).

- $M = \rho_b \cdot V$, la masse du bouchon (égale au produit de sa masse volumique avec son volume).
- $P = \rho_b \cdot V \cdot g$, le poids du bouchon.
- g, l'accélération de la pesanteur.

- F_p, la force de poussée. Elle est égale au produit de la masse volumique de l'eau (ρ_e) avec g et du volume immergé ($V_i(z_A(t))$), *i.e.*

$$F_p(z_A(t)) = \rho_e \cdot g \cdot V_i(z_A(t)).$$

- $F_v\left(z_A(t), \frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right)$ est la force de frottement visqueux. Elle est donnée par la formule

$$F_v\left(z_A(t), \frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right) = -\frac{1}{2}\rho_e \cdot C_x \cdot S\left(z_A(t), \frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right) \cdot \left(\frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right)^2.$$

- $S\left(z_A(t), \frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right)$ est la surface apparente du bouchon vis-à-vis du fluide et est définie de la manière suivante.
  - Si le bouchon est totalement hors de l'eau, cette surface est nulle.
  - Si le bouchon descend et est (au moins partiellement) immergé :
    si $0 \leq z < H/2$, $S\left(z_A(t), \frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right) = \pi \cdot r(z)^2$,
    si $z \geq H/2$, $S\left(z_A(t), \frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right) = \pi \cdot r(H/2)^2$.
  - Si le bouchon remonte et est (au moins partiellement) immergé :
    si $0 \leq z < H/2$, $S\left(z_A(t), \frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right) = 0$.
    si $H/2 \leq z \leq H$, la surface à prendre en compte est celle de la couronne : $S\left(z_A(t), \frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right) = -\pi \cdot \left(r(H/2)^2 - r(z)^2\right)$
    si $z > H$, $S\left(z_A(t), \frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right) = -\pi \cdot r(H/2)^2$.
    La force de frottement s'opposant au mouvement, si le bouchon remonte, la résultante de cette force selon la direction $\overrightarrow{e}_z$ est positive, d'où le signe $-$ placé ici.
- C_x est le coefficient de trainée aérodynamique.

Dans toute la suite de ce devoir, on pourra supposer que les grandeurs suivantes (et uniquement celles-ci) ont été définies.

```
R = 1E-2 # m, rayon minimum du bouchon de révolution
H = 4.5*1E-2 # m, hauteur du bouchon
rho_eau = 1000 # kg / m**3 masse volumique de l'eau
rho_b = 240 # kg / m**3, masse volumique du liège
g = 9.81 # m / s**2, accélération de la pesanteur en
Cx = 1 # Coefficient de trainée aérodynamique
N = 1000 # Nombre de trapèzes pour les calculs d'intégrales
```

QUESTION Écrire la fonction `T(f,a,b,N)` permettant de donner l'estimation de $\int_{z=a}^{b} f(z)\mathrm{d}z$ par la méthode des trapèzes, avec N trapèzes sur le segment $[a,b]$.

QUESTION Écrire une fonction `volume_immerge(z)` qui renvoie le volume immergé du bouchon en fonction de la profondeur du bas du bouchon (noté z) en utilisant la fonction définie à la question précédente.

QUESTION Écrire l'instruction permettant de calculer le volume total du bouchon, que l'on affectera à la variable `V`.

QUESTION Écrire une fonction `Fv(z,zp)` qui renvoie la force de frottement visqueux $F_v\left(z_A(t), \frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right)$.

QUESTION Exprimer $\frac{\mathrm{d}^2 z_A}{\mathrm{d}t^2}(t)$ en fonction de ρ_e, ρ_b, $V_i(z_A(t))$, $F_v\left(z_A(t), \frac{\mathrm{d}z_A}{\mathrm{d}t}(t)\right)$ et $z_A(t)$.

On souhaite résoudre cette équation différentielle (équation **??**) avec pour conditions initiales :

$$\begin{cases} z_A(t=0) = -0,2 \\ \frac{\mathrm{d}z_A}{\mathrm{d}t}(t=0) = 0 \end{cases} \quad \text{(CI)}$$

QUESTION Définir l'expression de $Z(t)$, Z_0 et de $F(Z(t),t)$ pour que l'équation différentielle (**??**) avec les conditions initiales (**??**) soit équivalente au problème de Cauchy d'ordre 1 :

$$Z'(t) = F(Z(t),t) \quad \text{et} \quad Z(0) = Z_0. \qquad (\mathscr{F})$$

Écrire une suite d'instructions permettant de définir une telle fonction `F(Z,t)` ainsi que la condition initiale `Z0`.

QUESTION Écrire une fonction `euler(F, tmin, tmax, Z0, h)` prenant en argument la fonction `F` définie précédemment, `tmin` et `tmax` définissant l'intervalle de résolution, le vecteur `Z0` définissant les consitions initiales ainsi que `h` le pas de discrétisation temporelle et permettant de résoudre de manière approchée l'équation (**??**) par la méthode d'Euler.

On donne sur la figure **??** le résultat de l'application de la méthode d'euler pour simuler le comportement de la chute libre d'un bouchon à partir de 20 cm au dessus du niveau de l'eau.

QUESTION Commenter brièvement la cohérence physique d'une telle courbe. Pouvez-vous fournir une explication et une solution au problème observé, tout en restant dans le cadre de la méthode d'Euler?

fichiers

FIC-000.fex

Sur un réseau social, vous avez un certain nombre d'amis. Vous pouvez aller sur la page de vos amis afin de savoir combien ils ont d'amis chacuns. Le réseau va également vous indiquer combien d'amis vous avez en commun avec chacun de vos amis. Cependant, vous aimeriez en savoir un peu plus. Plus précisément, vous vous demandez combien il y a de gens qui font partie des amis de vos amis et qui ne font pas déjà partie de vos amis.

Compter à la main prendrait bien trop de temps, donc vous décidez d'écrire un algorithme pour cela. Votre algorithme utilisera comme données le fichier `amis.csv`, dont les lignes comportent deux entiers chacune, séparés par une virgule, qui sont deux identifiants de personnes amies l'une de l'autre.

Notez que l'amitié est toujours réciproque : si I_1 est ami de I_2 alors I_2 est ami de I_1. On donnera soit (I_1, I_2) soit (I_2, I_1) dans le fichier, mais pas les deux. De plus, une

personne n'est jamais amie d'elle-même, *i.e.* I_1 et I_2 sont toujours différents.

Prenons comme exemple le fichier suivant.

```
0,1
0,2
0,3
1,2
1,3
2,3
1,4
2,5
2,4
3,6
7,8
```

Alors, les amis de la personne d'identifiant 0 sont les personnes d'identifiants 1, 2 et 3. Les amis des personnes d'identifiants 1, 2 et 3 sont les personnes d'identifiants 0, 1, 2, 3, 4, 5 et 6. Les amis d'amis de la personne d'identifiant 0 qui ne sont pas des amis directs de la personne d'identifiant 0 (ni cette même personne) sont donc les personnes d'identifiant 4, 5 et 6. Il y en a donc 3.

QUESTION Combien d'amis d'amis de la personne d'identifiant α ne sont pas des amis de la personne d'identifiant α, ni α ?

FIC-001.tex

QUESTION Écrire une fonction `resume(nom_de_fichier)` qui prend en argument une chaîne de caractères `nom_de_fichier` et qui renvoie le triplet `(L,m,c)` où, pour le fichier dont le chemin est `nom_de_fichier`; :

- `L` est le nombre de lignes du fichier;
- `m` est le nombre de mots (sous chaîne maximale non vide de caractères consécutifs, sans blanc) du fichier;
- `c` est le nombre de caractères du fichier (blancs compris).

On rappelle qu'un blanc est un retour chariot (`\n`), une tabulation (`\t`) ou une espace.

FIC-002.tex

On suppose que l'on dispose d'un fichier `nombres.txt` contenant des entiers naturels séparés pas des blancs (on rappelle qu'un blanc est soit un espace, soit une tabulation `\t` soit un retour à la ligne `\n`).

Pour faciliter votre travail, un exemple de tel fichier est disponible sur le site de classe. Nous vous encourageons fortement à tester vos fonctions sur des exemples dont vous aurez calculé le résultat à la main.

QUESTION Écrire une fonction `somme()`, sans argument, renvoyant la somme de tous les entiers contenus dans ce fichier.

QUESTION Pour chaque ligne du fichier, on effectue le produit des entiers de cette ligne. Écrire une fonction `moyenne()`, sans argument, renvoyant la moyenne de tous ces produits.

QUESTION Si i est un entier inférieur au nombre de lignes du fichier, on note s_i la somme des entiers de la ligne i. Écrire une fonction `inversion()`, sans argument, retournant le nombre de couples (i, j) tels que $i < j$, et $s_j < s_i$.

Les invariants de boucle seront impérativement précisés en commentaires, dans le script renvoyé.

QUESTION Dans cette dernière question, on suppose que le fichier contient n lignes, et que chaque ligne ne contient qu'un entier. Quelle est en fonction de n la complexité de l'algorithme `inversion()` ?

On supposera que les opérations de lecture dans le fichier se font en temps constant. Ainsi, l'opération consistant à lire tous les entiers du fichier et à les stocker un par un dans un tableau sera supposée avoir une complexité en $O(n)$.

Pour plus de clarté, vous recopierez le code de la fonction `inversion()` avant d'étudier sa complexité.

FIC-003.tex

Le fichier gilberte.txt contient un extrait du premier volume d'*À l'ombre des jeunes filles en fleurs* de Marcel Proust, où il évoque la fin de sa relation avec Gilberte Swann, son premier amour.
Dans toute la suite, les questions porteront sur les lignes allant du numéro α inclus au numéro $\alpha + 300$ exclu. Comme toujours en Python, la première ligne du texte porte le numéro 0.
On pourra remarquer qu'aucun mot n'est coupé lors d'un retour à la ligne, ce qui simplifiera les choses.

QUESTION Combien de fois apparaît le mot « Gilberte » ?

On appellera ici *paragraphe* toute partie du texte telle que :

- le premier caractère est une tabulation;
- le caractère précédant cette tabulation est un retour à la ligne, sauf s'il s'agit du début du texte;
- le dernier caractère est un retour à la ligne;
- le caractère suivant ce retour à la ligne est une tabulation, sauf s'il s'agit de la fin du texte.

QUESTION Combien y a-t-il de paragraphes ?

Proust est célèbre pour la longueur de ces phrases. Vous allez donc chercher la phrase la plus longue de la portion du texte que vous devez étudier.
On rappelle que l'on appelle *blanc* un caractère qui est un espace, un retour chariot ou une tabulation.
On appellera *point* les caractères « . », « ! » et « ? ».
On appellera ici *phrase* toute portion du texte telle que :

- le premier caractère est un blanc;
- le caractère précédant ce blanc est un retour à la ligne ou un point, sauf s'il s'agit du début du texte;
- le dernier caractère est un point;
- le caractère suivant ce point est un blanc, sauf s'il s'agit de la fin du texte.

Attention, une phrase peut contenir un point autre que son dernier caractère, par exemple dans les phrases :

Je m'écriai « Gilberte ! » en la voyant.

Ou

J'étais perplexe ...

Un autre exemple pour être sûr que tout est compris : dans le texte :

Bonjour. Ça va ?
Comment te portes-tu ?
Très bien,
je te remercie.

il y a quatre phrases, la phrase la plus longue est la dernière, et elle comporte 28 caractères (ne pas oublier les deux blancs au début (retour chariot et tabulation), le retour chariot au milieu et le point à la fin !).

Et une dernière remarque : votre portion de texte risque de commencer au milieu d'une phrase et de finir au milieu d'une autre : pour simplifier les choses, la portion de votre texte commençant au premier caractère de votre texte et finissant là où commence la phrase suivante ne sera pas prise en compte, même si par hasard c'était une phrase complète. Et si votre texte ne finit pas par un point suivi d'un blanc, vous ne tiendrez pas compte de la phrase incomplète qui clôt votre portion de texte.

QUESTION Quel est le nombre de caractères de la phrase la plus longue ?

QUESTION Quel est le nombre de caractères de la phrase la plus courte ?

Le texte d'*À la recherche du temps perdu* étant tombé dans le domaine public depuis 1987, vous décidez de publier vous-mêmes la partie du texte qui vous est échue aujourd'hui, dans une édition au format particulier : ayant hérité de votre oncle épicier d'une grande quantité de rouleaux pour caisse enregistreuse, vous décidez de faire imprimer le texte sur ce papier. Dans un souci de lisibilité, vous décidez également d'imprimer ce texte dans une police de taille standard : chaque ligne comportera au plus neuf caractères, sans compter les retours chariot.
La règle est la suivante : les retours à la ligne du texte de base sont tous supprimés, sauf ceux marquant un changement de paragraphe. Les retours à la ligne de la nouvelle édition seront donc ceux des changements de paragraphe et ceux imposés par la taille maximale de neuf caractères de chaque ligne. S'il le faut, les mots seront coupés par un retour à la ligne sans scrupule. Par contre, les espaces en début de ligne seront supprimés.
Par exemple, le texte :

Marcel Proust est un enfant à la santé fragile. Toute sa vie il a des difficultés respiratoires.
Très jeune, il fréquente des salons aristocratiques.

sera réédité de la manière suivante :

Marcel P
roust est
un enfant
à la sant
é fragile
. Toute s
a vie il
a des dif
ficultés
respirato
ires.
Très jeu
ne, il fr
équente d
es salons
aristocra
tiques.

QUESTION Donner la 71-ème ligne (donc la ligne numéro 70) de votre texte réédité : on signalera les tabulations en début de ligne et les espaces par le symbole [0.4cm]

FIC-004.tex

Écrire une fonction `moyennes(fichier_notes,fichier_moyen` lisant les notes écrites dans `fichier_notes` et écrivant dans le `fichier_moyennes` le prénom de chaque élève, suivi par la moyenne de ses notes.

Par convention, la première ligne de chaque fichier comporte les titres des colonnes et la première colonne contient le prénom de chaque étudiant. Un exemple (fichiers `notes.csv` et `moyennes.csv`) sera mis en ligne sur le site de classe.

On respectera le format `.csv`, le séparateur par défaut sera donc les virgules `(',')`.

FIC-005.tex

Analyse préliminaire : filtrage numérique

Filtre numérique passe bas du premier ordre

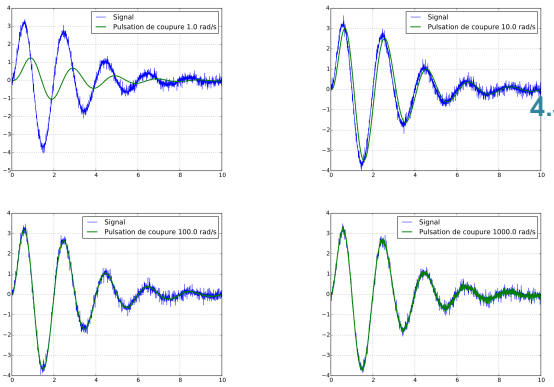
Soit un filtre linéaire du premier ordre. Son équation différentielle est de la forme :

$$s(t)+\tau\frac{\mathrm{d}s(t)}{\mathrm{d}t}=Ke(t)$$

En utilisant un schéma d'Euler implicite, on a l'approximation suivante : $\frac{\mathrm{d}s(t)}{\mathrm{d}t}=\frac{s_k-s_{k-1}}{h}$. En conséquences,

$$s_k+\tau\frac{s_k-s_{k-1}}{h}=Ke_k \Leftrightarrow s_k=\frac{hKe_k+\tau s_{k-1}}{h+\tau}$$

La fréquence d'échantillonnage est ici de 1 KHz. Le pas de dérivation est de 0,001 s. On réalise alors différents filtrages en faisant varier la pulsation de coupure du filtre.



On observe que lorsque la pulsation de coupure diminue, le signal est de plus en plus lissé, limitant ainsi le bruit. En revanche, le signal est atténue et déphasé.

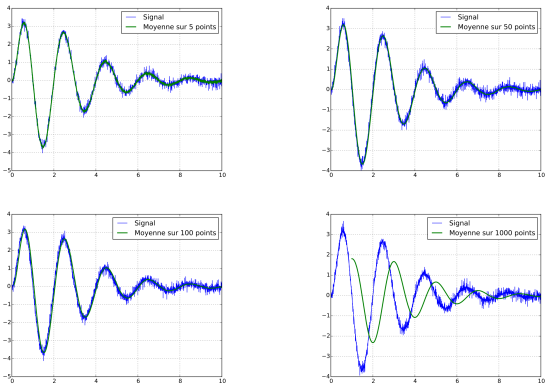
On donne l'algorithme permettant de réaliser ce filtre à partir d'un tableau "$signal$" contenant des valeurs pour chaque instant t correspondant au tableau "t" avec une fréquence de coupure "$freq$".

```
def filtrage_passe_bas(freq,t,signal):
    tau = 1/freq # Coupure du filtre
    K=1
    res=[signal[0]]
    for i in range(1,len(signal)):
        h=t[i]-t[i-1]
        res.append((h*K*signal[i]+tau*res[-1])/(h+tau))
    return res
```

4.2 Filtrage numérique par moyenne glissante

Une autre solution pour lisser une courbe est de réaliser une moyenne glissante. Si on réalise ce filtrage en temps réel, cela signifie qu'un point lissé à l'échantillon n est la moyenne des $n-1$ échantillons précédents et de l'élément en cours.

Il faut donc attendre l'acquisition de $n-1$ échantillons avant de disposer de la courbe.



On donne l'algorithme permettant de réaliser ce filtre à partir d'un tableau "$signal$" contenant des valeurs pour chaque instant t correspondant au tableau "t" avec une taille de fenêtre de filtrage "$freq$".

```
def filtrage_moyenne(signal,fenetre):
    filtrageg =signal[0:fenetre-1]
    for i in range(fenetre-1,len(signal)):
        s = sum(signal[i-fenetre+1:i+1])/fenetre
        filtrageg=np.concatenate([filtrageg,np.array
    return filtrageg
```

4.3 Références

- Cours de Xavier Pessoles support de cours de PSI* La Martinière Monplaisir.
- Patrick Beynet, *Supports de cours de TSI 2*, Lycée Rouvière, Toulon.
- David Crochet (créer à partir de KmPlot), CC-BY-SA-3.0, via Wikimedia Commons `https://upload.wikimedia.org/wikipedia/commons/6/6f/Fourier_d%27un_carr%C3%A9.svg`.

5 Analyse des performances d'un cycliste

5.1 Analyse d'un fichier gpx

5.1.1 Lecture du fichier

QUESTION Ecrire un programme permettant de savoir si un mot se trouve dans une chaine de caractère.

Un fichier gpx permet d'écrire l'ensemble des données issues d'un appareil du type gps ou montre connecté. Il recueille les 3 coordonnées spatiales (altitude par rapport au niveau de la mer, latitude et longitude) en fonction du temps. Ce fichier contient donc un ensemble de coordonnées spatiales en fonction du temps. Ouvrir le fichier gpx et analyser son contenu. On remarquera en particulier que le fichier est organisé d'un certaine manière. Ce fichier comporte différentes balises [1].

Le fichier ($< gpx >$) peut contenir :

- Des métadonnées ($< metadata >$), décrivant le contenu du fichier GPX par entre autre :
  - un nom ($< name >$)
  - une description ($< desc >$)
  - l'auteur du fichier ($< author >$) comprenant son nom, une adresse mail et un lien vers son site web.
  - ...
- Une liste de traces ou track ($< trk >$) chacune décrite par :
  - un nom ($< name >$);
  - le type d'itinéraire($< type >$)
  - des segments de trace ($< trkseg >$), le passage d'un segment à un autre indique une extinction du récepteur GPS ou une perte de réception. Un segment de trace est constitué :
    d'une liste ordonnée de points de trace ($< trkpt >$) dans laquelle figure respectivement la latitude ('lat') ainsi que la longitude ('long') en °.
    avec son altitude par rapport au niveau de la mer en mètres ($< ele >$)
    un horodatage ($< time >$)

Dans ce problème on pourra travailler sur un des deux fichiers gpx donné sur le site de la classe dans "info/mpsi2/Programmes montrés dans le cours"(rumilly.gpx, mont_dor.gpx).

QUESTION Ecrire un programme permettant de stocker sous la forme de 4 tableaux nommés $tp, lat, long$

1. source : wikipedia : `https://fr.wikipedia.org/wiki/GPX_(format_de_fichier)`

et lat représentant respectivement pour chaque point de mesure le temps en s par rapport à l'instant initial, la latitude en °, la longitude en ° ainsi que l'altitude par rapport au niveau de la mer en m.

5.2 Superposition du tracé sur une carte

Pour des traces petites (comme un parcours cycliste) on peu assimiler les longitude et latitude à des coordonnées cartésiennes et il n'est donc pas nécessaire de réaliser des projections cartographiques.

QUESTION A l'aide des fonctions contenues dans le module "$matplotlib.pyplot$" tracer un graphe décrivant la liste des points des différentes coordonnées extraites précédemment en mettant en abscisses les valeurs de longitude et en ordonnées celles de latitude. On obtient alors la trace gps du parcours réalisé.

Pour superposer cette trace à une carte, on peut utiliser le module "$folium$" qu'il faut installer avec la commande :

```
pip install folium
```

Puis importer avec :

```
import folium
```

La fonction "Map" permet de créer un objet carte centrée avec en point possédant une latitude et une longitude. Dans les arguments de cette fonction, il faut préciser les coordonnées en latitude et en longitude (en °) ainsi le niveau d'agrandissement initial avec l'entier z.

```
macarte = folium.Map(location=[lat,long], zoom_start=z)
```

Etant donnée une trace GPS comme celle importée précédemment, on peut connaître les latitudes min,max et les longitudes min,max qui lui correspondent. On peut se servir de ces dernière pour définir la carte initiale.

Pour superposer à cette carte la trace gps extraite précédemment on peut utiliser la fonction "$PolyLine$" qui est utilisable de la façon suivante :

```
folium.PolyLine(points).add_to(macarte)
```

Où la variable "$points$" est une liste de tuple contenant pour chaque coordonnée du fichier gpx respectivement la latitude et la longitude.

La méthode "save" appliqué à l'objet "macarte" et prenant en argument une chaine de caractère correspondante au nom du fichier "html" permet de créer une page html représentant la carte définie précédemment.

QUESTION mettre en oeuvre les fonctions du module "$folium$" pour créer une page HTML superposant la trace GPS sur une carte avec un centrage et un niveau d'agrandissement satisfaisant. Vous pourrez visualiser la page créée en l'ouvrant un navigateur internet (Firefox, Google Chrome, etc...)

5.2.1 Traitement du fichier

On donne l'expression de la vitesse et de l'accélération en coordonnées sphérique d'un point matériel M dans son mouvement par rapport au référentiel terrestre R_0 :

$$\overrightarrow{V}(M/R_0) = (\dot{r})\overrightarrow{e}_r + (r\cdot\dot{\theta})\overrightarrow{e}_\theta + (r\cdot\sin\theta\dot{\varphi})\overrightarrow{e}_\varphi$$

$$\overrightarrow{a}(M/R_0) = (\ddot{r} - r\cdot\dot{\theta}^2 - r\cdot\sin^2\theta\dot{\varphi}^2)\overrightarrow{e}_r + (2*\dot{r}\dot{\theta} + r\cdot\ddot{\theta} - r\cdot\sin$$
$$+ (2r\cos\theta\dot{\theta}\dot{\varphi} + 2\dot{r}\sin\theta\dot{\varphi}$$

On peut relier les paramètres r, θ et φ aux coordonnées gps :

$$\begin{cases} r = alt + R \\ \theta = \dfrac{\pi}{2} - lat \\ \varphi = long \end{cases}$$

avec R le rayon de la terre supposé constant et égal à $6371km$

QUESTION Écrire un programme prenant en argument un vecteur $y(t)$ dépendant du temps ainsi qu'un vecteur t de mêmes longueurs et renvoyant le vecteur dy correspondant à l'estimation de $\dfrac{dy(t)}{dt}$ par différence finie. On pourra la noter $diff(y,t)$.

QUESTION Appliquer cette méthode pour donner des estimation de $\dot{r}$, $\dot{\theta}$, $\dot{\varphi}$ que pouvons-nous dire quant à la qualité de l'estimation de la dérivée?

QUESTION Utiliser les fonctions de filtrage numérique décrites en première partie pour améliorer les résultats.

QUESTION Écrire une fonction $vitesse$ prenant en argument les tableaux r, $theta$ et phi et renvoyant 4 tableau Vr, $Vtheta$ et $Vphi$ et V correspondant respectivement aux 3 coordonnées du vecteur vitesse dans le système de coordonnées sphérique ainsi que la norme du vecteur vitesse.

QUESTION Mettre en évidence à l'aide d'un tracé l'éventuelle corrélation entre la vitesse du cycliste et le profil en altitude du circuit.

5.3 Modélisation des performances mécaniques

5.3.1 Modélisation de la puissance

En considérant l'ensemble (cycliste+velo) comme un solide à masse ponctuelle (cela revient à négliger les effets d'inertie de la rotation des roues) on peut estimer la puissance que le cycliste doit développer en mouvement (P_c) à l'aide du théorème de l'énergie cinétique :

$$P_c = P_{inertie} + P_{aero} + P_{Pes} + P_{roul}$$

- $P_{inertie}$ est la puissance due aux effets d'inertie et elle provient de la variation de l'énergie cinétique.
- P_{aero} est la puissance qui résulte des effets aéronautiques.
- P_{pes} est la puissance qui provient de l'action mécanique de pesanteur.

- P_{roul} est la puissance due à la résistance au roulement.

On se placera dans le référentiel terrestre supposé galiléen (R_0). Le cycliste ayant réalisé l'activité possède une masse $M = 70kg$. On prendra l'accélération de la pesanteur $g = 9,81m \cdot s^{-2}$.

5.3.2 Puissance due à la résistance au roulement

QUESTION Déterminer une fonction P_{roul} qui prend en argument la masse d'un cycliste, un vecteur V comprenant la norme de la vitesse de déplacement du cycliste ($V = \|\overrightarrow{V}(M/R_0)\|$) associé à une trace GPX, la pression dans les pneus et qui renvoie la puissance instantanée de roulement.

5.3.3 Puissance due aux effets d'inertie

La puissance due aux effets d'inertie peut s'estimer de la façon suivante :

$$P_{inertie} = M \cdot V \cdot \frac{dV}{dt}$$

QUESTION Déterminer une fonction $Pinertie$ prenant en arguments la masse d'un cycliste, un vecteur V comprenant la norme de la vitesse de déplacement du cycliste ($\|\overrightarrow{V}(M/R_0)\|$) associé à une trace GPX, la pression dans les pneus et qui renvoie la puissance instantanée de roulement.

5.3.4 Puissance aéronautique

QUESTION Déterminer une fonction $Paero$ prenant en arguments, la vitesse du vent (Vv),sa direction (d_v), le produit $C_d \times A_p$, un vecteur V comprenant les composantes de la vitesse de déplacement du cycliste ($V(M/R_0)$) en coordonnées sphériques associées à une trace GPX et renvoyant l'estimation de la puissance aérodynamique instantanée.

5.3.5 Puissance due aux effets de pesanteur

La puissance de pesanteur va dépendre du dénivelé qui correspond à la variation de la pente en fonction de la distance parcouru.

QUESTION Écrire une fonction permettant de calculer l'intégrale par rapport au temps d'une fonction $f(t)$ avec la méthode des trapèze. Elle prendra en arguments deux tableaux de même dimension f et t et renverra l'estimation de $I = \int_{t'=0}^{t} f(t')dt'$ sous la forme d'un tableau ayant la même dimension que t et f.

QUESTION Utiliser cette fonction pour donner une estimation de la distance parcouru en fonction du temps que l'on notera x.

La pente instantanée peut s'estimer par $\frac{dr(t)}{dx}$.

QUESTION Écrire une fonction prenant en argument les tableaux r et x et renvoyant l'estimation de la pente sous la forme d'un tableau de même dimension. Pour vérifier la cohérence de cette estimation, tracer sur une même figure deux graphes représentant en fonction du temps $r(t)$ et $\frac{dr(t)}{dx}$.

La puissance de pesanteur peut s'estimer de la façon suivante :

$$P_{pes} = M \cdot g \cdot \sin\left[\arctan\left(\frac{dr(t)}{dx}\right)\right] \cdot \|\overrightarrow{V}(M/R_0)\|$$

QUESTION Déterminer une fonction $Ppes$ prenant en arguments, la masse du cycliste, la coordonnées $r(t)$, un vecteur V comprenant la norme de la vitesse de déplacement du cycliste ($\|\overrightarrow{V}(M/R_0)\|$) associé à une trace GPX et renvoyant l'estimation de la puissance de pesanteur instantanée.

Puissance totale et profile de puissance record

Puissance totale

QUESTION A l'aide des questions précédentes tracer en fonction du temps les différentes puissances instantanées. Estimer la puissance totale instantanée que doit fournir le cycliste (Pc) et la superposer sur ce graphe.

QUESTION Comment interpréter les valeurs négatives. On veillera à ne pas en tenir compte pour la suite.

L'énergie dépensée sur le circuit par le cycliste peut s'estimer de la façon suivante :

$$E = \int_{t=0}^{t_f} P_c(t')dt'$$

QUESTION Mettre en oeuvre la méthode des trapèze pour calculer l'énergie totale consommée par le cycliste.

Profile de puissance record

QUESTION Proposer une méthode pour obtenir ce graphe en tenant compte des données gps issues du fichier gpx.

QUESTION A partir de l'estimation de la puissance instantanée précédente déterminer le PPR.

FIC-006.tex

Vous trouverez sur le site de classe un fichier `cats.csv`. Il contient des mesures de corpulence de chats. Vous y trouverez trois colonnes : le sexe de chaque chat (colonne `Sex`), son poids en kilogrammes (colonne `Bwt`) et le poids de son cœur en grammes (colonne `Hwt`).

Écrire un script Python permettant de lire ce fichier et d'écrire dans un autre fichier, pour chaque sexe, le poids moyen et le poids du cœur moyen.

FIC-007.tex

Vous trouverez sur le site de classe un fichier `sainte_lyon.csv`. Il contient la liste des 31 premiers concurrents de la course de trail nocture **La Saintélyon** de l'édition 2017 reliant Saint-Etienne à Lyon. Vous y trouverez deux colonnes : le nom de chaque concurrent (colonne `Nom`) et le temps mis pour parcourir le parcours de 72 km (colonne `Temps`) au format $heures : minutes : secondes$.

Écrire un script Python permettant de lire ce fichier et d'écrire dans un autre fichier, pour chaque nom, le temps en secondes et la vitesse en km/h.

FIC-008.tex

Le fichier sur lequel vous allez travailler a été chiffré en utilisant le chiffre de César. C'est un code par décalage très simple, qui fonctionne avec une clef $c \in [\![0,26[\![$. En numérotant les lettres de 0 (lettre `a`) à 25 (lettre `z`), chaque lettre de numéro r est remplacée par la lettre de numéro $(r+c)\%26$.

Par exemple, si la clef est 2 (lettre `c`), le `a` est transformé en `c`, le `b` en `d`, ..., le `x` en `z`, le `y` en `a` et le `z` en `b`.

Indication : dans cet exercice, vous pourrez introduire `alphabet="abcdefghijklmnopqrstuvwxyz"`. Ainsi, si $i \in [\![0,26[\![$, `alphabet[`i`]` sera la lettre n^o i.

QUESTION Appliquez la tranformation de clef 24 (lettre `y`) à votre texte. Quelle est le $(\alpha+50)^e$ mot de ce nouveau texte (on numérote à partir de zéro) ?

Pour déchiffrer un texte chiffré avec la clef c, il suffit d'appliquer la tranformation de clef $(26-c)\%26$. Cette dernière clef sera appelée *clef de déchiffrement* du texte. Par exemple, si la clef de chiffrement est 3 (lettre `d`), alors la clef de déchiffrement sera 23 (lettre `x`).

La cryptanalyse du chiffre de César, c'est-à-dire l'obtention de la clef à partir du texte chiffré, peut se faire par analyse de fréquences.

La fréquence d'une lettre dans un texte est le rapport entre le nombre d'occurences dans ce texte de cette lettre et le nombre total de lettres du texte.

QUESTION Quelle est la fréquence de la lettre `e` dans votre texte chiffré ?

Pour comparer deux tableaux de fréquences de lettres $t=[t_0,\dots,t_{25}]$ et $u=[u_0,\dots,u_{25}]$, on utilise la distance

$$d(t,u)=\sum_{k=0}^{25}(t_k-u_k)^2.$$

Le fichier `frequences.txt` contient une liste de fréquences des lettres, que l'on supposera être celle du français. Vous le retrouverez dans le tableau **??**.

QUESTION Quelle est la distance entre le tableau des fréquences des lettres de votre texte chiffré et celui des fréquences données dans le fichier `frequences.txt` ?

L'algorithme de déchiffrement est le suivant : pour chaque clef $c \in [\![0,26[\![$, on applique le code de César de clef c au texte (ce qui donne un second texte), puis l'on calcule le tableau des fréquences des lettres de ce second texte, et enfin l'on calcule la distance entre ce tableau et celui donné par le fichier `frequences.txt`.

On sélectionne alors la clef qui minimise les distances calculées ci-dessus.

QUESTION Quelle est la clef de déchiffrement de votre texte ?

QUESTION Quelle est le $(\alpha+100)^{\text{e}}$ mot du texte déchiffré (on numérote à partir de zéro) ?

nombres

NBR-000.tex

QUESTION Trouver la somme de tous les entiers qui sont la somme des factorielles de leurs chiffres, en écriture décimale. On écrira une fonction `enigme()` renvoyant le résultat demandé.

NBR-001.tex

QUESTION Que renvoie la fonction suivante ? Pourquoi ?

```
def mystere():
    a,b,n = 1,1.,0
    while a==b:
        a,b,n = 2*a,2*b,n+1
    return n
```

NBR-002.tex

QUESTION Que renvoie la fonction suivante ? Pourquoi ?

```
def mystere():
    a,b,n = 1,1.,0
    while a==b:
        a,b,n = 2*a+1,2*b+1,n+1
    return n
```

NBR-003.tex

En mathématiques, la méthode de la dichotomie à partir d'un segment $[a_0, b_0]$ donne deux suites infinies de réels (a_n) et (b_n) vérifiant $\forall n \in \mathbb{N}$, $a_n < b_n$. Est-ce le cas en informatique ?

Justifier la réponse et l'illustrer par un script.

NBR-004.tex

Un diviseur strict d'un entier naturel n est un diviseur positif de n différent de n. Un nombre est déficient s'il est strictement plus grand que la somme de ses diviseurs stricts.

Par exemple, 15 a pour diviseurs stricts 1, 3 et 5. Comme $1+3+5<15$, 15 est déficient.

QUESTION Calculer le nombre de nombres déficients de 10^6 (inclu) à 10^6+10^5 (exclu).

NBR-005.tex

On admet que pour tout entier $n \geq 1$ il existe une unique suite $a_1, \dots, a_p$ vérifiant :

- $n = \sum_{k=1}^{p} a_k \times k!$
- $a_p \neq 0$ et $\forall k \in [\![1,p]\!]$, $0 \leq a_k \leq k$.

On appellera *écriture en base factorielle* de l'entier n la chaîne $a_1 - a_2 - \cdots - a_p$.

Par exemple, $42 = 0\times 1! + 0\times 2! + 3\times 3! + 1\times 4!$, l'écriture de 42 en base factorielle est donc $0-0-3-1$.

QUESTION Donner l'écriture en base factorielle de $\alpha^3+\alpha^2+\alpha+10^5$.

plot

PLT-000.tex

On dispose en `Python` de la fonction `randrange` dans la bibliothèque `random`. Elle prend deux arguments entiers a et b, et renvoie un entier tiré uniformément dans $[\![a;b[\![$. On peut considérer que deux appels successifs de cette fonction donnent des tirages indépendants l'un de l'autre.

Une variable aléatoire R suit la loi de *Rademacher* si R ne peut prendre que 1 ou -1 comme valeurs, avec $P(R=1) = P(R=-1) = \dfrac{1}{2}$.

Étant donné une suite $(R_n)_{n\in\mathbb{N}}$ de variables aléatoires indépendantes et suivant chacune la loi de Rademacher, on considère la suite $(S_n)_{n\in\mathbb{N}}$ définie par

$$S_0 = 0 \quad \text{et} \quad \forall n \in \mathbb{N}^*,\ S_n = \sum_{k=1}^{n} R_n.$$

Notamment, si $n \in \mathbb{N}$, $S_{n+1} = S_n + R_{n+1}$. On peut voir cette suite $(S_n)_{n\in\mathbb{N}}$ comme une *marche aléatoire* sur $\mathbb{Z}$, et représenter son évolution en traçant les points (k, S_k) pour k dans une plage raisonnable.

Étant donné une telle suite $(S_n)_{n\in\mathbb{N}}$ et deux entiers naturels a, b avec $a < b$, on dit que la marche aléatoire S réalise une excursion hors de zéro entre a et b si

$$S_a = S_b = 0 \quad \text{et} \quad \forall k \in [\![a+1; b[\![,\ S_k \neq 0.$$

Vous trouverez sur le site de la classe :

- un fichier `marche.txt` contenant un exemple de telle marche ($n = 500$, les valeurs sont séparées par des espaces) ;
- un fichier `excursions.txt` contenant les excursions pour cette marche (une excursion par ligne) ;
- un fichier `d02m-nom-marche.png` représentant cette marche (vous devrez renvoyer une figure similaire) ;
- un fichier `d02m-nom-excursions.png` représentant les excursions hors de zéro de cette marche (vous devrez renvoyer une figure similaire).

QUESTION Écrire une fonction `R()`, sans argument, renvoyant une réalisation d'une variable aléatoire de loi de Rademacher.

QUESTION Écrire une fonction `marche(n)`, prenant en argument un entier naturel n, et renvoyant une liste contenant une réalisation de $[S_0, \dots, S_n]$.

QUESTION Écrire une fonction `excursions(m)`, prenant en argument une liste d'entiers `m`, et renvoyant la liste des excursions pour la marche `m`. Par exemple, si

$$\texttt{m} = [0,1,0,-1,-2,-1,-2,-1,0,1,2],$$

la fonction `excursions(m)` renverra la liste

$$\texttt{les\_e} = [\,[0,1,0]\,,\,[0,-1,-2,-1,-2,-1,0]\,].$$

QUESTION Écrire une fonction `trace_marche(m,nom_de_fichier)` ne renvoyant rien et enregistrant dans le fichier `nom_de_fichier` le tracé de la marche aléatoire dont les valeurs sont données dans `m`.

Vous enverrez à votre enseignant un tracé produit par cette fonction pour $n = 500$.

QUESTION Écrire une fonction `trace_excursions(les_e,nom_de_fichier)` ne renvoyant rien et enregistrant dans le fichier `nom_de_fichier` le tracé des excursions données dans `les_e`.

Vous enverrez à votre enseignant un tracé produit par cette fonction, pour la même marche que la question précédente.

Remarque : on n'hésitera pas à retirer quelques réalisations de la marche de manière à avoir un tracé lisible et un nombre d'excursions raisonnable.

PLT-001.tex

Une fonction pseudo-périodique est une fonction qui permet de décrire des phénomènes physiques du type «oscillatoire-amortis». Cette fonction peut-être le produit d'une fonction harmonique sinusoïdale et d'une fonction exponentielle décroissante, de la forme :

$$f : x \to \sin(x) \cdot \exp(\alpha \cdot x)$$

Le coefficient α doit être un réel négatif strictement pour avoir un régime amorti. On prendra dans la suite $\alpha = -0,1$.

On souhaite obtenir le tracé ci-dessous (voir figure **??**), faisant apparaître la courbe représentative de la fonction f sur le segment $[0,6\pi]$, mais également les courbes enveloppes $(x \mapsto \pm\exp(\alpha \cdot x))$ et la courbe représentative de la fonction harmonique non amortie $(x \mapsto \sin x)$.

QUESTION Donner les instructions permettant de réaliser ce tracé. On veillera à bien préciser les modules utilisés et à utiliser les instructions permettant d'obtenir le tracé complet.

programmation dynamique

PDY-000.tex

Dans une pyramide de nombres, en partant du sommet, et en se dirigeant vers le bas à chaque étape, on cherche à maximiser le total de la somme des nombres traversés.

$$\begin{array}{c}
\underline{\mathbf{2}} \\
4\ \underline{\mathbf{5}} \\
1\ 7\ \underline{\mathbf{8}} \\
2\ 3\ 1\ \underline{\mathbf{6}} \\
9\ 4\ 6\ \underline{\mathbf{5}}\ 2
\end{array}$$

Sur cet exemple, la somme totale maximale est 26, obtenue en parcourant les nombres soulignés.

On peut voir la pyramide comme un graphe, parcourir les 16 chemins, et choisir celui qui a le plus grand total. Quand la pyramide a n niveaux, il y a 2^{n-1} chemins, ce qui rend vite cet algorithme inexploitable.
Un algorithme récursif est aussi possible, mais alors certains calculs sont effectués plusieurs fois, et là encore l'algorithme est trop long pour des pyramides de taille conséquente.
La méthode la plus rapide est celle utilisant la *programmation dynamique*. L'idée est résumée dans la séquence suivante :

$$\begin{array}{cccc}
2 & 2 & 2 & 2 \\
4\ 5 & 4\ 5 & 4\ 5 & 20\ 24 \\
1\ 7\ 8 & 1\ 7\ 8 & 12\ 16\ 19 & \\
2\ 3\ 1\ 6 & 11\ 9\ 7\ 11 & & \\
9\ 4\ 6\ 5\ 2 & & &
\end{array}$$

À vous de comprendre cette idée et d'écrire un programme calculant ce total maximum pour des pyramides que l'on définit de la manière suivante : chaque pyramide est donnée dans un tableau, dont les éléments sont les lignes de la pyramide, représentées elles-mêmes dans un tableau. Par exemple, la pyramide de l'exemple sera représentée dans le tableau `[ [ 2 ], [ 4, 5 ], [ 1, 7, 8 ], [ 2, 3, 1, 6 ], [ 9, 4, 6, 5, 2 ] ]`.

Vous pourrez utiliser la fonction suivante pour construire ces pyramides :

```
def pyramide(alpha,n):
    p = []
    x = alpha
    for i in range(n):
        l = []
        for j in range(i+1) :
            y = x % 10
            l.append(y)
            x = (15091 * x) % 64007
        p.append(l)
    return p
```

Enfin, on rappelle que si $x \in \mathbb{N}$, on note $x\%10$ le reste de la division euclidienne par 10.

QUESTION Donner la somme maximale pour la pyramide ayant 20 lignes, définie par les $u_k\%10$ (lus de haut en bas, de gauche à droite, la première ligne est $[u_0\%10]$, la seconde $[u_1\%10,\ u_2\%10]$ *etc.*).

QUESTION Donner la somme maximale pour la pyramide ayant 100 lignes, définie par les $u_k\%10$ (lus de haut en bas, de gauche à droite, la première ligne est $[u_0\%10]$, la seconde $[u_1\%10,\ u_2\%10]$ *etc.*).

python bases

PYB-000.tex

Évaluer les expressions suivantes en repérant auparavant celles qui donnent des résultats de type `int`.

a)
```
4+2
```
b)
```
25-3
```
c)
```
-5+1
```
d)
```
117*0
```
e)
```
6*7-1
```
f)
```
52*(3-5)
```
g)
```
5*(-2)
```
h)
```
22/(16-2*8)
```
i)
```
42/6
```
j)
```
18/7
```
k)
```
(447+3*6)/5
```
l)
```
0/0
```

Calculer les restes et les quotients des divisions euclidiennes suivantes :

a) $127 \div 8$
b) $54 \div 3$
c) $58 \div 5$
d) $58 \div (-5)$
e) $-58 \div 5$
f) $-58 \div (-5)$
g) $17583 \div 10$
h) $17583 \div 100$
i) $17583 \div 10^4$
j) $(2^7 + 2^4 + 2) \div 2^5$
k) $(2^7 + 2^4 + 2) \div 2^7$
l) $(2^7 + 2^4 + 2) \div 2^{10}$

Calculer les nombres suivants avec une expression Python en repérant auparavant ceux qui donnent un résultat de type `int`.

a) 3^5
b) 2^{10}
c) $(-3)^7$
d) -3^7
e) 5^{-2}
f)
g) $7^{(5^4)}$ $(7^5)^4$
h) 5^{7+6}
i) $5^7 + 6$
j) $2^{(10^4)}$

PYB-001.tex

Calculer les restes et les quotients des divisions euclidiennes suivantes :

a) $127 \div 8$
b) $54 \div 3$
c) $58 \div 5$
d) $58 \div (-5)$
e) $-58 \div 5$
f) $-58 \div (-5)$
g) $17583 \div 10$
h) $17583 \div 100$
i) $17583 \div 10^4$
j) $(2^7 + 2^4 + 2) \div 2^5$
k) $(2^7 + 2^4 + 2) \div 2^7$
l) $(2^7 + 2^4 + 2) \div 2^{10}$

Calculer les nombres suivants avec une expression Python en repérant auparavant ceux qui donnent un résultat de type `int`.

a) 3^5
b) 2^{10}
c) $(-3)^7$
d) -3^7
e) 5^{-2}
f)
g) $7^{(5^4)}$ $(7^5)^4$
h) 5^{7+6}
i) $5^7 + 6$
j) $2^{(10^4)}$

PYB-002.tex

Évaluer les expressions suivantes.

a)
```
4.3+2
```
b)
```
2.5-7.3
```
c)
```
42+4.
```
d)
```
42+4
```
e)
```
42.+4
```
f)
```
12*0.
```
g)
```
11.7*0
```
h)
```
2,22/(1.6-2*(447+3*6))/5
```
i)
```
42/6
```
j)
```
1,8/7
```
k)
```
0/0
```

PYB-003.tex

Calculer, sans utiliser la fonction `sqrt` ni la division flottante `/`, les nombres suivants.

a) $\dfrac{1}{7,9}$
b) $\sqrt{6,2}$
c) $\dfrac{1}{\sqrt{3,5}}$
d) $2\sqrt{2}$

De base, on ne peut réaliser que des calculs élémentaires avec Python. Cependant, il est possible d'avoir accès à des possibilités de calcul plus avancées en utilisant une *bibliothèque*. Par exemple, la bibliothèque `math` permet d'avoir accès à de nombreux outils mathématiques. On peut donc taper

```
from math import sqrt, log, exp, sin, cos, tan, pi
```

pour avoir accès à toutes ces fonctions.
Calculer les nombres suivants (on n'hésitera pas à consulter l'aide en ligne).

a) e^2
b) $\sqrt{13}$
c) $\cos\left(\dfrac{\pi}{5}\right)$
d) $e^{\sqrt{5}}$
e) $\ln 2$
f) $\ln 10$
g) $\log_2 10$
h) $\tan\left(\dfrac{\pi}{2}\right)$

PYB-004.tex

Les expressions suivantes sont-elles équivalentes ?

a)
```
8.5 / 2.1
```
b)
```
int(8.5) / int(2.1)
```
c)
```
int(8.5 / 2.1)
```

Et celles-ci ?

a)
```
float(8 * 2)
```
b)
```
8 * 2
```
c)
```
8. * 2.
```

Prévoir la valeur des expressions suivantes puis vérifier cela (avec IDLE).

a)
```
1.7 + 1.3
```
b)
```
2 - 1
```
c)
```
2. - 1
```
d)
```
2 - 1.
```
e)
```
(2 - 1).
```
f)
```
.5 + .5
```
g)
```
4 / (9 - 3**2)
```
h)
```
4 / (9. - 3**2)
```

PYB-005.tex

Déterminer de tête la valeur des expressions suivantes avant de le vérifier (avec IDLE).

a)
```
0 == 42
```
b)
```
1 = 1
```
c)
```
3 == 3.
```
d)
```
0 != 1
```
e)
```
0 < 1
```
f)
```
4. >= 4
```
g)
```
0 !< 1
```
h)
```
2*True + False
```
i)
```
-1 <= True
```
j)
```
1 == True
```
k)
```
False != 0.
```
l)
```
True and False
```
m)
```
True or False
```
n)
```
True or True
```
o)
```
(2 == 3-1) or (1/0 == 5)
```
p)
```
(1/0 == 5) or (2 ==
```
q)
```
True or True and Fa
```
r)
```
False or True and F
```
s)
```
not (1 == 1 or 4 ==
```
t)
```
(not 1 == 1) or 4 =
```
u)
```
(1/0 == 5) or True
```

PYB-006.tex

Dans IDLE, cliquer sur File/New File. Une nouvelle fenêtre apparaît. Dans cette fenêtre, taper les lignes suivantes.

```
3*2
print(2*3.)
17*1.27
```

Enregistrer le document produit puis, toujours dans cette fenêtre, exécutez-le (touche F5). Observez le résultat dans l'interpréteur interactif. Modifiez les instructions pour que tous les résultats de calcul s'affichent dans l'interpréteur interactif.

PYB-007.tex

QUESTIONDans chaque cas, indiquez le type que vous utiliseriez pour modéliser les grandeurs suivantes dans leur contexte scientifique usuel. Vous justifierez brièvement chaque réponse.

a) La taille d'un individu en mètres.
b) Le tour de taille d'un manequin, en millimètres.
c) Le nombre d'Avogadro.
d) Le nombre de Joules dans une calorie.
e) Le nombre de secondes dans une année.
f) Le plus grand nombre premier représentable avec 20 chiffres en écriture binaire.

PYB-100.tex

Prévoir les résultats des expressions suivantes, puis le vérifier grâce à l'interpréteur interactif d'IDLE.

a)
```
(1,2)
```
b)
```
(1)
```
c)
```
(1,)
```
d)
```
(,)
```
e)
```
()
```
f)
```
()+()
```
g)
```
()+() == ()
```
h)
```
(1,2)+3
```
i)
```
(1,2)+(3)
```
j)
```
(1,2)+(3,)
```
k)
```
(1,2)+(3,4,5)
```
l)
```
len((1,7,2,"zz",[]))
```
m)
```
len(())
```
n)
```
len(("a","bc")+("cde",""))
```

PYB-101.tex

Pour chaque séquence d'instruction, prévoir son résultat puis le vérifier grâce à l'interpréteur interactif d'IDLE.

a)
```
t = (2,"abra",9,6*9,22)
print(t)
t[0]
t[-1]
t[1]
t[1] = "cadabra"
```
b)
```
res = (45,5)
x,y = res
(x,y) == x,y
(x,y) == (x,y)
print x
print(y)
x,y = y,x
print(y)
```
c)
```
v = 7
ex = (-1,5,2,"","abra",8,3,v)
5 in ex
abra in ex
(2 in ex) and ("abr" in ex)
v in ex
```

PYB-102.tex

Prévoir les résultats des expressions suivantes, puis le vérifier grâce à l'interpréteur interactif d'IDLE.

a)
```
"abba"
```
b)
```
abba
```
c)
```
""
```
d)
```
"" == " "
```
e)
```
""+""
```
f)
```
""+"" == ""
```
g)
```
"May"+" "+"04th"
```
h)
```
"12"+3
```
i)
```
"12"+"trois"
```
j)
```
len("abracadabra")
```
k)
```
len("")
```
l)
```
len("lamartin"+"201
```

PYB-103.tex

Pour chaque séquence d'instruction, prévoir son résultat puis le vérifier grâce à l'interpréteur interactif d'IDLE.

a)
```
t = "oh oui youpi !"
print(t)
t[0]
t[-1]
t[1]
t[2]
t[1] = "o"
```
b)
```
ex = "abdefgh"
"a" in ex
a in ex
"def" in ex
"adf" in ex
```

PYB-104.tex

Prévoir les résultats des expressions suivantes, puis le vérifier grâce à l'interpréteur interactif d'IDLE.

a)
```
[1,2,3,"a"]
```
b)
```
123a
```
c)
```
[]
```
d)
```
[]+[]
```
e)
```
[]+[] == []
```
f)
```
[1,2] + [5,7,9]
```
g)
```
[0,0]+[0]
```
h)
```
len(["a","b"])
```
i)
```
len([])
```
j)
```
len([[]])
```
k)
```
len([[[]]])
```
l)
```
len([0,0]+[1])
```

PYB-105.tex

Pour chaque séquence d'instruction, prévoir son résultat puis le vérifier grâce à l'interpréteur interactif d'IDLE.

a)
```
t = [1,2,3,4,5,6]
u = ["a","b","c","d"]
print(t+u)
t[0]
t[-1]
z = t[3]
print(z)
t.append(7)
print(t)
```

b)
```
ex = ["sin","cos","tan","log","exp"]
"log" in ex
log in ex
"l" in ex
z = ex.pop()
print(z)
z in ex
print(ex)
```

c)
```
u = [1,2,3,4,5,6]
L = u
u = [1,2,3,42,5,6]
print(L)
```

d)
```
u = [1,2,3,4,5,6]
L = u
u[3] = 42
print(L)
```

PYB-106.tex

Calculer cette suite d'expressions.

```
[i for i in range(10)]
[compt**2 for compt in range(7)]
[j+1 for j in range(-2,8)]
```

Sur ce modèle, obtenir de manière synthétique :

a) la liste des 20 premiers entiers naturels impairs ;
b) la liste de tous les multiples de 5 entre 100 et 200 (inclus) ;
c) La liste de tous les cubes d'entiers naturels, inférieurs ou égaux à 1000.
d) une liste contenant tous les termes entre −20 et 5 d'une progression arithmétique de raison 0,3 partant de −20.

PYB-107.tex

a) Affecter à `v` la liste `[2,5,3,-1,7,2,1]`
b) Affecter à `L` la liste vide.
c) Vérifier le type des variables crées.
d) Calculer la longueur de `v`, affectée à `n` et celle de `L`, affectée à `m`.
e) Tester les expressions suivantes : `v[0]`, `v[2]`, `v[n]`, `v[n-1]`, `v[-1]` et `v[-2]`.
f) Changer la valeur du quatrième élément de `v`.
g) Que renvoie `v[1:3]` ? Remplacer dans `v` les trois derniers éléments par leurs carrés.
h) Que fait `v[1] = [0,0,0]` ? Combien d'éléments y a-t-il alors dans `v` ?

PYB-108.tex

QUESTIONQuel type choisiriez-vous pour représenter les données suivantes ? Vous justifierez brièvement chaque réponse.

a) Le nom d'une personne.
b) L'état civil d'une personne : nom, prénom, date de naissance, nationalité.
c) Les coordonnées d'un point dans l'espace.
d) L'historique du nombre de 5/2 dans la classe de MP du lycée.
e) Un numéro de téléphone.
f) *Plus difficile :* l'arbre généalogique de vos ancêtres.

PYB-200.tex

Dans les cas où c'est possible, affecter les valeurs aux variables correspondantes à l'aide de l'interpréteur interactif d'IDLE. On notera `var` ← `a` pour dire que l'on affecte la valeur `a` à la variable `var`.

a) `ArthurDent` ← `42`
b) `4` ← `0.`
c) `L` ← `[]`
d) `list` ← `[1,2,3]`
e) `int` ← `5`
f) `s` ← `""`
g) `True` ← `1`
h) `ok` ← `ok`
i) `x` ← `"x"`
j) `a` ← `1<0`
k) `lam` ← `1/0`
l) `or` ← `"xor"`

PYB-201.tex

On considère les affectations `a` ← `-1.` et `b` ← `5`. Prévoir la valeur de chacune de ces expressions, puis le vérifier à l'aide de l'interpréteur interactif d'IDLE.

a) `a * a`
b) `a ** a`
c) `a ** a == aa+b == 5`
d) `a * b`
e) `a+6>=b`
g) `b<100 | a**2==-`
h) `b-3`

PYB-202.tex

On part des affectations suivantes : `a` ← `5` et `b` ← `0`. Pour la suite d'instructions suivante, prévoir ligne à ligne le résultat affiché par l'interpréteur interactif de Python ainsi que l'état des variables. Le vérifier grâce à l'interpréteur interactif d'IDLE, en prenant soin de partir d'une nouvelle session.

```
a*b
x = a**b + a
print(x)
print(y)
z = x
x = 5
print(z)
a = a+a**b
print(a)
```

PYB-203.tex

Affecter des valeurs toutes différentes aux variables `a`, `b`, `c` et `d`.

Á chaque fois, effectuer les permutations suivantes de manière naïve (c'est-à-dire, sans utiliser de `tuple`).

a) Échanger les contenus de `a` et de `b`.
b) Placer le contenu de `b` dans `a`, celui de `a` dans `c` et celui de `c` dans `b`.
c) Placer le contenu de `a` dans `d`, celui de `d` dans `c`, celui de `c` dans `b` et celui de `b` dans `a`.

Reprendre cet exercice en effectuant chaque permutation en une instruction à l'aide d'un `tuple`.

PYB-204.tex

a) Affecter à la variable `mon_age` l'âge que vous aviez il y a 13 ans.
b) Écrire l'opération qui vous permet d'actualiser votre âge, tout en conservant la même variable.
c) Que donne l'interpréteur après exécution des expressions suivantes ? Pourquoi ?

```
mon-age = 18
2013_mon_age = 18
True = 18
```

d) À partir d'une nouvelle session d'IDLE, exécuter les expressions suivantes et commenter le résultat.

```
age = 5
age = Age + 14
```

PYB-205.tex

Combien d'affectations sont suffisantes pour permuter circulairement les valeurs des variables $x_1, \cdots, x_n$ sans utiliser de variable supplémentaire ? Et en utilisant autant de variables supplémentaires que l'on veut ?

Mêmes questions en remplaçant suffisantes par nécessaires.

PYB-206.tex

Supposons que la variable x est déja affectée, et soit $n \in \mathbb{N}$. On veut calculer x^n sans utiliser la puissance, avec uniquement des affectations, autant de variables que l'on veut, mais avec le moins de multiplications possible. Par exemple, avec les 4 instructions :

```
>>> y1 = x * x
>>> y2 = y1 * x
>>> y3 = y2 * x
>>> y4 = y3 * x
```

on calcule x^5, qui est la valeur de `y4`.
Mais 3 instructions suffisent :

```
>>> y1 = x * x
>>> y2 = y1 * y1
>>> y3 = y2 * x
```

En fonction de n, et avec les contraintes précédentes, quel est le nombre minimum d'instructions pour calculer x^n ?

PYB-207.tex

Voici des affectations successives des variables a et b. Dresser un tableau donnant les valeurs de a et b à chaque étape.

```
>>> a = 1
>>> b = 5
>>> a = b-3
>>> b = 2*a
>>> a = a
>>> a = b
```

PYB-208.tex

Écrire une séquence d'instructions qui échange les valeurs de deux variables x et y.

PYB-209.tex

Écrire, sans variable supplémentaire, une suite d'affectation qui permute circulairement vers la gauche les valeurs des variables x, y, z : x prend la valeur de y qui prend celle de z qui prend celle de x.

PYB-300.tex

Ouvrir IDLE, écrire la fonction suivante dans un fichier, l'enregistrer, taper *run* (F5) puis utiliser la fonction dans l'interpréteur interactif d'IDLE. Décrire ensuite précisément ce que réalise cette fonction.

```
def split_modulo(n):
  """A vous de dire ce que fait cette fonction !"""
  return (n%2,n%3,n%5)
```

PYB-301.tex

QUESTIONÉcrire une fonction `moy_extr(L)` qui prend en argument une liste `L` et renvoie en sortie la moyenne du premier et du dernier élément de `L`.

PYB-302.tex

Écrire une fonction `norme` qui prend en argument un vecteur de $\mathbb{R}^2$ donnée par ses coordonnées et renvoie sa norme euclidienne. Vous devrez spécifier clairement le type de l'argument à l'utilisateur via la *docstring*.

PYB-303.tex

Écrire une fonction `lettre` qui prend en argument un entier `i` et renvoie la $\texttt{i}^{\text{e}}$ lettre de l'alphabet.

PYB-304.tex

Écrire une fonction `carres` qui prend en argument un entier naturel `n` et qui renvoie la liste des `n` premiers carrés d'entiers, en commençant par 0.

PYB-305.tex

QUESTIONOn cherche à écrire une fonction prenant en argument une liste d'entiers et incrémentant de 1 le premier élément de cette liste.

a) Écrire une telle fonction `incr_sans_effet_de_bord`, qui ne modifie pas la liste initiale et renvoie en sortie une nouvelle liste.

b) Écrire une telle fonction `incr_avec_effet_de_bord`, qui modifie la liste initiale et ne renvoie rien en sortie (ponctuer par un `return None`).

PYB-306.tex

QUESTION Écrire une fonction `appartient(a,c,r)` prenant en argument

- un couple de nombres `a` ;
- un couple de nombres `c` ;
- un nombre positif `r` ;

et renvoyant la valeur de vérité de « le point de coordonnées `a` est dans le disque fermé de centre de coordonnées `c` et de rayon `r` ».

PYB-307.tex

1. Ecrire une fonction qui à un nombre entier associe le chiffre des unités.
2. Ecrire une fonction qui à un nombre entier associe le chiffre des dizaines.
3. Ecrire une fonction qui à un nombre entier associe le chiffre des unités en base 8.

PYB-400.tex

QUESTIONIndenter de deux manières différentes la suite d'instructions suivante afin que la variable `t` contienne `True` pour une indentation, puis `False` pour l'autre.

```
x = 0
y = 5
t = False
if x>=1:
t = True
if y <= 6:
t = True
```

PYB-401.tex

QUESTIONRéécrire la suite d'instructions suivante de manière plus appropriée.

```
from random import randrange
n = randrange(100) # Un entier aléatoire entre 0 et 99
if n <= 10:
    print("Trop petit")
else:
    if n >= 50:
        print("Trop grand")
    else:
        print("Juste comme il faut")
```

PYB-402.tex

a) Écrire une fonction `neg(b)` qui renvoie la négation du booléen `b` sans utiliser `not`.

b) Écrire une fonction `ou(a,b)` qui renvoie le ou logique des booléen `a` et `b` sans utiliser `not`, `or` ni `and`.

c) Écrire une fonction `et(a,b)` qui renvoie le et logique des booléen `a` et `b` sans utiliser `not`, `or` ni `and`.

PYB-403.tex

Indenter de deux manières différentes la suite d'expressions suivante de manière à ce qu'à son exécution, le programme affiche soit la liste de tous les éléments de L inférieurs ou égaux à m, soit juste le dernier.

```
from random import randrange
L = [randrange(100) for i in range(100)] # 100 valeurs entre 0 et 99.
m = 50
for x in L:
if x <= m:
p = x
print(p)
```

PYB-404.tex

QUESTIONQue fait la fonction suivante ? La corriger pour qu'elle coïncide avec le but annoncé.

```
def inv(n):
    """Somme des inverses des n premiers entiers naturels non nuls"""
    s = 0
    for k in range(n):
        x = 1/k
    s = s+x
    return s
```

PYB-405.tex

Décrire ce que fait cette suite d'instructions.

```
from random import randrange
n = randrange(1000) # Un entier entre 0 et 999
L = [randrange(100) for i in range(n+1)] # n+1 valeu
p = 0
for x in L:
    if x <= 10:
        p = p + x**2
```

Et celle-ci ?

```
from random import randrange
n = randrange(1000) # Un entier entre 0 et 999
L = [randrange(100) for i in range(n+1)] # n+1 valeu
p = 0
for i in range(n+1):
    if L[i] <= 10:
        p = p + i**2
```

PYB-406.tex

Écrire une suite d'instructions permettant de calculer la somme des racines carrées des cinquante premiers entiers naturels non nuls.

PYB-407.tex

Écrire la suite d'instructions suivantes dans un fichier, l'enregistrer puis l'exécuter (F5). À l'instant qui vous convient, presser Ctrl+C.

```
a = 1
while a>0:
    a = a+1
```

PYB-408.tex

Que fait la fonction suivante ? La corriger pour qu'elle coïncide avec le but annoncé.

```
def sqrt_int(n):
    """Renvoie la partie entière de la racine carrée
    s = 0
    while s**2 <= n:
        s = s+1
        s = s-1
    return s
```

PYB-409.tex

QUESTIONUn taupin se lance dans un marathon d'exercices, mais se fatigue vite. Il réalise le i^e exercice en $\sqrt{i}$ minutes. Combien d'exercices arrive-t-il à faire en 4 heures ? Pour faciliter la correction, on écrira une fonction `nb_exos()` ne prenant pas d'argument et renvoyant le résultat demandé.

PYB-410.tex

Expliquer et justifier ce que fait la fonction suivante.

```
def cesar(k):
    """?????"""
    alphabet = "abcdefghijklmnopqrstuvxyz"
    s = ""
    for i in range(26):
        s = s + alphabet[i+k % 26]
    return s
```

PYB-500.tex

QUESTIONÉcrire une fonction `racine(n)` prenant en argument un entier naturel `n` et renvoyant sa racine carrée comme un entier si c'est un carré parfait, comme un flottant sinon.

PYB-501.tex

QUESTIONDans le jeu de la bataille navale, on représente chaque case par un couple d'entiers entre 0 et 9.

Un navire a ses extrémités sur les cases `a` et `b`. Un joueur tire sur la case `x`.

Écrire une fonction `touche(a,b,x)` qui renvoie un booléen indiquant si le navire est touché ou non.

PYB-502.tex

QUESTIONUn banquier vous propose un prêt de 400 000 euros sur 40 ans «à 3% par an» — ce qui, dans le langage commercial des banquiers, veut dire 0,25% par mois — avec des mensualités de 1431,93 euros. Autrement dit, vous contractez une dette de 400 000 euros. Chaque mois, cette dette augmente de 0,25% puis est diminuée du montant de votre mensualité. À la fin des 40×12 mensualités, il ne vous reste plus qu'à vous acquitter d'une toute petite dette, que vous rembourserez aussitôt.

a) Écrire une fonction `reste_a_payer(p, t, m, d)` renvoyant le montant de cette somme à rembourser immédiatement après le paiement de la dernière mensualité, où p est le montant total du prêt en euros (dans l'exemple, 400 000), t son taux mensuel (dans l'exemple, $0,25\times 10^{-2}$), m le montant d'une mensualité en euros (dans l'exemple, 1431,93) et d la durée en années (dans l'exemple, 40).
Indice : dans le cas donné dans cet énoncé, vous devez trouver un montant restant d'un peu moins de 7,12 euros.

b) Écrire une fonction `somme_totale_payee(p, t, m, d)` renvoyant la somme totale (mensualités plus le dernier paiement) que vous aurez payé au banquier.

c) Écrire une fonction `cout_total(p, t, m, d)` renvoyant le coût total du crédit, c'est-à-dire le total de ce que vous avez payé moins le montant du prêt.

PYB-503.tex

QUESTIONUn banquier vous propose de vous prêter p euros, à un taux de $12t$% par an — ce qui, dans le langage commercial des banquiers, veut dire t% par mois — avec des mensualités de m euros. Autrement dit, vous contractez une dette de p euros. Chaque mois, cette dette augmente de t% puis est diminuée du montant de votre mensualité. Lorsque votre dette, augmentée du taux, est inférieure à la mensualité, il suffit de régler le solde en une fois.

Écrire une fonction `duree_mensualite(p,t,m)` renvoyant le nombre de mensualités nécessaires au remboursement total du prêt.

Attention : que se passe-t-il si la mensualité est trop petite?

Indice : dans le cas où le prêt est `p` $= 4\times 10^5$*, le taux est* `t` $= 0,25\times 10^{-2}$ *et la mensualité est* `m` $= 1431,93$*, on trouvera une durée de remboursement de* 480 *mois.*

PYB-504.tex

QUESTIONÉcrire une fonction `comb(p,n)` renvoyant $\binom{n}{p}$ (nombre de combinaisons de p éléments parmi n). On pourra bien entendu introduire une fonction auxiliaire[2].

PYB-505.tex

QUESTIONOn appelle suite de Fibonacci la suite F définie par $F_0=0$, $F_1=1$ et pour tout $n\in\mathbb{N}$, $F_{n+2}=F_{n+1}+F_n$.

Écrire une fonction `fib(n)` calculant et renvoyant la valeur de F_n.

Pensez à vérifier le résultat de votre fonction en 0, 1 et en 5 (vous calculerez à la main F_5 *avant* de le faire calculer par votre fonction).

PYB-506.tex

QUESTIONOn pose $u_0=1$ et pour tout $n\in\mathbb{N}$,

$$u_{n+1}=\frac{1}{2}\left(u_n+\frac{n+1}{u_n}\right)$$
$$\text{et } v_n=\sum_{k=0}^{n}\frac{1}{u_k^5}$$

Écrire une fonction `f(n)` renvoyant la valeur de v_n.

On peut montrer que $(v_n)_{n\in\mathbb{N}}$ converge.

Vous pouvez calculer u_n pour de grandes valeurs de n.

Attention : on fera attention à ce que le calcul de `f` de demande pas trop de (re)calculs inutiles. Pour fixer les idées, vous pouvez considérer que `f(10**6)` doit être calculé en (largement) moins d'une minute.

PYB-507.tex

QUESTIONÉcrire une fonction `somme1(n)` et une fonction `somme2(n)` prenant en argument un entier naturel `n` et renvoyant respectivement

$$\sum_{1\leq i,j\leq n}\frac{1}{i+j^2}, \tag{10}$$
$$\sum_{1\leq i<j\leq n}\frac{1}{i+j^2}. \tag{11}$$

2. C'est-à-dire, comme l'indique l'étymologie, une autre fonction dont le but sera de vous aider à répondre à la question.

Au besoin, on introduira des fonctions auxiliaires.

PYB-508.tex

On considère la suite u définie par $u_0 \in [-2;2]$ et $\forall n \in \mathbb{N}, u_{n+1} = \sqrt{2-u_n}$. On rappelle que u converge vers 1.

QUESTION Écrire une fonction `valeur_u(n,u0)` qui, à un entier naturel `n` et un flottant `u0`, renvoie u_n.

QUESTION Écrire une fonction `approche_u(eps,u0)` qui, à deux flottants `eps` et `u0`, renvoie le plus petit rang $n \in \mathbb{N}$ tel que $|u_n - 1| \leq$ `eps`.

PYB-509.tex

QUESTION Écrire une fonction `comb(n,p)` calculant $\binom{n}{p}$ pour deux entiers naturels n, p vérifiant $0 \leq p \leq n$, en utilisant la formule :

$$\binom{n}{p} = \frac{n}{p} \times \frac{n-1}{p-1} \times \cdots \times \frac{n-p+1}{1}.$$

On veillera à ce que le résultat donné par la fonction `comb` soit d'un type convenable.

Remarque : On ne demande pas de vérifier que les arguments de cette fonction vérifient les conditions imposées.

QUESTION Justifier que la fonction écrite donne bien le bon résultat, notamment à l'aide d'un invariant de boucle.

PYB-510.tex

Définir la fonction f qui à x associe $\begin{cases} 2 & \text{si} \quad x \in [-4,-2] \\ -x & \text{si} \quad x \in [-2,0] \\ 0 & \text{si} \quad x \in [0,4] \end{cases}$

PYB-511.tex

Ecrire une fonction calculant le produit des entiers impairs de 1 à $2n+1$.

PYB-512.tex

Soit $(a,b,c) \in \mathbb{R}^2$, $a \neq 0$.

Ecrire une fonction qui renvoie les solutions $ax^2 + bx + c = 0$ si celles-ci sont réelles, une phrase disant qu'il n'y a pas de solutions réelles sinon.

Modifier pour introduire le cas de la racine double.

PYB-513.tex

Écrire une fonction `somme_chiffres(n)` qui prend en argument un entier naturel `n` et qui renvoie la somme des chiffres de `n` (écrit en base dix).

On pourra utiliser toutes les fonctions de conversion présentes dans Python, mais la fonction rendue devra comporter au moins une boucle non triviale.

PYB-514.tex

Que renvoie la fonction suivante ? Le justifier, notamment à l'aide d'un invariant.

```
def mystere(n,p):
    """Précondition : n entier positif, p entier"""
    if p < 0 or p > n :
        return 0
    else :
        f = 1
        for i in range(p) :
            f = f * (n + 1 - p + i) // (i + 1)
        return f
```

PYB-515.tex

Pour tout $n \in \mathbb{N}^*$, on définit

$$H_n = \sum_{k=1}^{n} \frac{1}{k}.$$

QUESTION Écrire une fonction `H_depasse(M)` prenant en entrée un nombre `M` et renvoyant en sortie le plus petit entier naturel non nul n vérifiant $H_n \geq$ `M`.

PYB-516.tex

On considère la fonction suivante.

```
def mystere(L) :
    """Précondition : L est une liste de nombres"""
    x,n,i = L[0],len(L),1
    while i<n and x > L[i] :
        L[i-1],L[i] = L[i],L[i-1]
        i = i+1
    return None
```

QUESTION Montrer que «`x = L[i-1]`» est un invariant de boucle pour la boucle `while` de la fonction `mystere`.

QUESTION Donner un variant de boucle pour la boucle `while` de la fonction `mystere`. Que peut-on en déduire ?

QUESTION Si `L` est une liste de nombres, que fait `mystere(L)` ? Le justifier, notamment à l'aide des questions précédentes (vous pourrez cependant écrire un ou plusieurs autres invariants, au besoin).

Remarque : vous avez tout intérêt à utiliser cette fonction et à observer son fonctionnement *avant* de répondre à ces questions.

PYB-517.tex

On considère la fonction suivante.

```
def mystere(a,b) :
    """Précondition : a,b sont des entiers, a>0, b>1
    k,p = 0,1
    while a % p == 0 :
        k = k+1
        p = p*b
    return k-1
```

QUESTION Dresser un tableau de valeurs décrivant les valeurs des variables `k` et `p` en entrée des trois premiers tours de la boucle `while` de la fonction `mystere(a,b)`.

On pourra au besoin faire intervenir les variables `a` et `b`.

QUESTION En s'aidant de la question précédente, écrire un invariant de boucle pour la boucle `while` de la fonction `mystere(a,b)`. On justifiera la réponse.

QUESTION Donner un variant de boucle pour la boucle `while` de la fonction `mystere(a,b)`. On justifiera la réponse.

QUESTION Déduire des questions précédentes qu'un appel de la fonction `mystere(a,b)` renvoie un résultat et déterminer le résultat alors renvoyé. On justifiera la réponse.

sql

SQL-000.tex

Les classes de MPSI du lycée LMM organisent un tournoi d'intelligences artificielles de morpion. Une partie de morpion se déroule sur un damier de trois cases par trois cases. Un premier joueur choisit une case. Puis, le deuxième joueur choisit une autre case. Puis, le premier joueur choisit une troisième case, et l'on continue ainsi jusqu'à ce que :

- soit un joueur aligne trois cases, dans ce cas il gagne ;
- soit le damier est rempli, dans ce cas il y a match nul.

Les professeurs de ces classes ont donc créé une base de données afin de gérer ce tournoi. Cette base contient deux tables :

- une table contenant les informations des joueurs : identifiant du joueur, pseudonyme, date de naissance, classe ;
- une table contenant les informations des parties : identifiant de la partie, identifiant du joueur n° 1, identifiant du joueur n° 2, résultat de la partie, coups joués, date de la partie.

Voici les instructions SQL ayant permis de créé ces deux tables.

```
CREATE TABLE joueurs (
          idj INTEGER,
          pseudo VARCHAR(50),
          datenaiss DATE,
          classe VARCHAR(50),
          PRIMARY KEY (idj)
          );

CREATE TABLE parties (
          idp INTEGER,
          idj1 INTEGER,
          idj2 INTEGER,
          res INTEGER,
          coups INTEGER,
          date DATETIME,
          PRIMARY KEY(idp),
          FOREIGN KEY(idj1) REFERENCES joueurs,
          FOREIGN KEY(idj2) REFERENCES joueurs
          );
```

Quelques conventions et rappels.

- Si le résultat d'une partie vaut 0, il y a match nul, s'il vaut 1 le joueur 1 a gagné et s'il vaut 2 le joueur 2 a gagné.
- Pour une partie, on lit dans l'entier `coups` les coups effectués par les joueurs. Ainsi, si `coups = 54892`, alors
  - le joueur 1 joue dans la case 5 ;
  - le joueur 2 joue dans la case 4 ;
  - le joueur 1 joue dans la case 8 ;
  - le joueur 2 joue dans la case 9 ;
  - le joueur 1 joue dans la case 2.

  Ainsi, dans cette partie, le joueur 1 a gagné (alignement sur la deuxième colonne).
- Les dates (type `DATE`) sont au format `AAAA-MM-JJ`.
- Les dates-heures (type `DATETIME`) sont au format `AAAA-MM-JJ HH:MM:SS`.
- Les dates et dates-heures sont écrites comme des chaînes de caractères (exemple : `"1515-09-13"`) et peuvent être comparées entre elles.
- Si une requête ne donne pas qu'une réponse (exemple : plusieurs lignes vérifient un critère), on inscrira toutes les réponses.
- Pour effectuer une jointure d'une table `T` avec elle même, on pourra lui donner un alias avec l'instruction `AS`. Voici un exemple.

```
SELECT *
FROM T
JOIN T AS Tbis ON T.[...] = Tbis.[...] ;
```

QUESTION Quel est le pseudonyme de l'étudiant d'identifiant n° 42 ?

QUESTION Quel est l'identifiant de l'étudiant dont le pseudonyme est `"Galois"` ?

QUESTION Combien de parties ont été jouées ?

QUESTION Combien y a-t-il eu de matchs nuls ?

QUESTION Combien de parties différentes (*i.e.* de successions de coups différentes) ont été jouées ?

QUESTION Quel est l'identifiant de l'étudiant le plus jeune ? Les donner tous s'il y en a plusieurs.

QUESTION Combien de parties ont été jouées par l'étudiant de pseudonyme `"Tux"` ?

QUESTION Combien de parties ont été perdues par l'étudiant de pseudonyme `"Tux"` ?

QUESTION Combien de parties ont été gagnées par des étudiants de MPS1 contre des étudiants de MPS2 ?

QUESTION Quels sont les pseudonymes des joueurs ayant joué le plus de parties en tant que joueur n° 1 ? Les donner tous s'il y en a plusieurs.

QUESTION Combien de parties ont été jouées en sept coups ou moins, en 2018 et entre deux étudiants d'une même classe ?

SQL-001.tex

La base de données[3] `medocs.sqlite` contient 5 tables :

LABORATOIRES : Contient une liste de laboratoires pharmaceutiques. Cette table possède deux attributs :
- **id :** Identifiant du laboratoire dans la base de données.
- **laboratoire :** Nom du laboratoire.

CIS_COMPO : Contient la liste des compositions qualitatives et quantitatives des médicaments de la base de données, substance par substance. Cette table contient 8 attributs :
- **code_CIS :** Code CIS (code identifiant de spécialité) d'un médicament, vous pouvez le considérer comme un identifiant de médicament.
- **désignation :** Désignation de l'élément pharmaceutique.
- **code_substance :** Code de la substance.
- **dénomination_substance :** Dénomination de la substance.
- **dosage :** Dosage de la substance.
- **ref_dosage :** Référence de ce dosage. Exemple : `"(pour) un comprimé"`.
- **nature_compo :** Nature du composant (principe actif : « SA » ou fraction thérapeutique : « ST »).
- **numéro_liaison :** Numéro permettant de lier, le cas échéant, substances actives et fractions thérapeutiques.

HAS_Liens : Contient les liens vers les avis de la commission de transparence (CT) de la Haute Autorité de la Santé (HAS). Cette table contient 2 attributs :
- **code_HAS :** Code de dossier HAS.
- **lien :** Lien vers la page d'avis de la CT.

CIS_bdpm : Cette table contient la liste des médicaments commercialisés, ou en arrêt de commercialisation depuis moins de trois ans. Cette table contient 11 attibuts :
- **code_CIS :** Code CIS (code identifiant de spécialité) d'un médicament, vous pouvez le considérer comme un identifiant de médicament.
- **dénomination :** Dénomination du médicament.
- **forme :** Forme pharmaceutique.
- **voie :** Voies d'administration (avec un séparateur «; » entre chaque valeur quand il y en a plusieurs).
- **statut :** Statut administratif de l'autorisation de mise sur le marché (AMM).
- **procédure :** Type de procédure d'autorisation de mise sur le marché (AMM).
- **commercialisation :** État de commercialisation.
- **date_AMM :** Date d'AMM (format JJ/MM/AAAA).
- **statutBdM :** Valeurs possibles : « Alerte » (icône rouge) ou « Warning disponibilité » (icône grise).
- **numéro_autorisation :** Numéro de l'autorisation européenne.
- **titulaire :** Numéro du laboratoire titulaire.
- **surveillance :** Surveillance renforcée (triangle noir) : valeurs « Oui » ou « Non ».

CIS_HAS_SMR : Cette table contient l'ensemble des avis de SMR (Service médical rendu) de la HAS. Cette table contient 6 attributs :
- **code_CIS :** Code CIS (code identifiant de spécialité) d'un médicament, vous pouvez le considérer comme un identifiant de médicament.
- **code_HAS :** Code de dossier HAS.
- **motif :** Motif d'évaluation.
- **date_avis :** Date de l'avis de la Commission de la transparence (format AAAAMMJJ).
- **valeur :** Valeur du SMR.
- **libellé :** Libellé du SMR.

Question.

QUESTION `code_CIS` est-il une clé primaire de la table `CIS_bdmp`?

QUESTION `code_HAS` est-il une clé primaire de la table `CIS_HAS_SMR`?

QUESTION Donner le nom du laboratoire dont le numéro d'identification est α.

QUESTION Donner le nombre de médicaments produits par ce laboratoire.

QUESTION Donner le nombre médicaments de ce laboratoire dont l'AMM a été donnée le 1er janvier 2000 ou après.

QUESTION Donner le code CIS du médicament de ce laboratoire ayant la plus ancienne AMM. S'il y en a plusieurs, on donnera le code CIS le plus petit.

QUESTION Quel est le lien internet vers la page d'avis de la CT sur ce dernier médicament (on ne donnera que la série de chiffres à la fin de cette adresse, qui sont au nombre de 6 ou 7 suivant les adresses)?

QUESTION Quelle est la somme des numéros de liaison des médicaments de ce laboratoire?

QUESTION Donner le code CIS du médicament de ce laboratoire ayant le plus de substances différentes. S'il y en a plusieurs, on donnera le code CIS le plus petit.

QUESTION Cette question peut se traiter en interrogeant la base de données depuis `Python` : combien de codes CIS contiennent votre numéro α comme sous-chaîne? Si votre α est compris entre 0 et 9, vous le ferez précéder d'un 0. Par exemple les α valant 8 ou 27 sont contenus dans 60008927 mais pas dans 60002875.

SQL-002.tex

Un professeur d'informatique a créé une base de données pour gérer les notes de ses interrogations hebdomadaires. Pour cela, il a créé trois tables : `etudiants`, `interros`, `notes`. Voici les commandes SQL ayant permis de créer ces tables.

```
CREATE TABLE etudiants (
    -- table des données des étudiants
    id INTEGER, -- identifiant de l'étudiant
```

3. Source : http ://base-donnees-publique.medicaments.gouv.fr/telechargement.php

```
    nom VARCHAR NOT NULL, -- nom de l'étudiant
    prenom VARCHAR NOT NULL, -- prénom de l'étudiant
    date_naissance DATE NOT NULL, -- date de naissance de l'étudiant, format AAAA-MM-JJ
    PRIMARY KEY (id)
    );

CREATE TABLE interros (
    -- table des données des interros
    id INTEGER, -- identifiant de l'interro
    titre VARCHAR, -- titre de l'interro
    sujet VARCHAR, -- sujet de l'interro
    date DATE, -- date du jour où a été donnée l'interro, format AAAA-MM-JJ
    PRIMARY KEY (id)
    );

CREATE TABLE notes (
    -- table des notes des étudiants aux interros
    id_etudiant INTEGER NOT NULL, -- identifiant de l'étudiant
    id_interro INTEGER NOT NULL, -- identifiant de l'interro
    note INTEGER NOT NULL, -- note obtenue
    PRIMARY KEY (id_etudiant, id_interro),
    FOREIGN KEY (id_etudiant) REFERENCES etudiants,
    FOREIGN KEY (id_interro) REFERENCES interros
    );
```

QUESTION Peut-on donner plusieurs notes au même étudiant et pour la même interrogation?

QUESTION Donner une requête SQL traduisant l'opération suivante, exprimée dans le vocabulaire d'algèbre relationnelle usuel :

$$\pi_{\texttt{sujet}}(\sigma_{\texttt{id=1}}(\texttt{interros})).$$

QUESTION Écrire une requête SQL permettant d'obtenir la liste des noms et prénoms des étudiants de la classe.

QUESTION Écrire une requête SQL permettant d'obtenir la liste des prénoms des étudiants de la classe, sans doublon.

QUESTION Les enfants du professeur se sont amusés à rentrer des données factices, que le professeur aimerait retrouver afin de les effacer ensuite. Écrire une requête SQL permettant d'obtenir la liste des identifiants des étudiants ayant pour nom `"reinedesneiges"`.

QUESTION Écrire une requête SQL permettant d'obtenir la date de naissance de l'étudiant le plus jeune de la classe.

QUESTION Écrire une requête SQL permettant d'obtenir la liste des noms et prénoms des étudiants ayant obtenu au moins un 20 à une des interrogations.

QUESTION Écrire une requête SQL permettant d'obtenir la liste des noms, prénoms d'étudiants et titres d'interrogations pour chaque note de 0 obtenue.

QUESTION Écrire une requête SQL permettant d'obtenir la liste des noms et prénoms des étudiants, avec la moyenne des notes de chaque étudiant.

QUESTION Écrire une requête SQL permettant d'obtenir la liste des noms et prénoms des étudiants, suivis pour chaque étudiant du nombre d'interrogations rendues.

QUESTION Écrire une requête SQL permettant d'obtenir le nom, le prénom et le nombre de copies rendues par l'étudiant ayant rendu le plus de copies (on suppose qu'il n'y en a qu'un). On rappelle que l'instruction `LIMIT k` permet de tronquer une table à ses `k` premières lignes.

QUESTION Écrire une requête SQL permettant d'obtenir la liste des noms et prénoms des étudiants ayant rendu au moins 10 copies, suivis du nombre de copies rendues.

QUESTION Écrire une requête SQL permettant d'obtenir la liste des titres des interrogations, suivie pour chaque interrogation du nombre d'étudiants qui n'ont pas rendu de copies.

QUESTION Écrire une requête SQL permettant d'obtenir la liste des titres des interrogations pour lesquelles tous les étudiants ont rendu une copie.

SQL-003.tex

On s'intéresse dans cet exercice à la gestion des données produites par la caisse enregistreuse d'une boulangerie (fictive), un jour donné. Sur chaque ticket produit par la caisse figure une ligne par produit vendu, indiquant notamment le nombre d'unités vendues par produit et le prix de chaque produit.

Chaque étudiant dispose d'un fichier `bdd_boulangerie_`α`.sqlit` possédant trois tables. Voici les commandes ayant permis de créer ces tables.

```
CREATE TABLE tickets (
        -- table des tickets
        id INTEGER,
        heure TIME NOT NULL,
        paiement VARCHAR(10) NOT NULL,
        PRIMARY KEY (id)
        );

CREATE TABLE produits (
        -- table des produits
        id INTEGER,
        nom VARCHAR(50) NOT NULL,
        prix FLOAT,
        PRIMARY KEY (id)
        );

CREATE TABLE lignes_tickets (
        -- table des lignes des tickets
        id INTEGER,
        idt INTEGER,
        idp INTEGER,
        quantite INTEGER NOT NULL,
        PRIMARY KEY (id),
        FOREIGN KEY (idt) REFERENCES tickets,
        FOREIGN KEY (idp) REFERENCES produits
        );
```

La table `produits` recence les produits vendus par la boulangerie et indique pour chaque produit son nom et son prix (en €).

La table `tickets` recence pour chaque ticket l'heure d'enregistrement du ticket et le moyen de paiment (carte bleue, liquide ou chèque).

La table `lignes_tickets` indique pour chaque ticket (identifiant `idt`) et chaque produit (identifiant `idp`) le nombre d'unités du produit acheté sur ce ticket.

QUESTION Combien d'unités ont été vendues par la boulangerie ce jour là ?

QUESTION Quel est le chiffre d'affaire de la boulangerie ce jour là ?

QUESTION Quel est l'identifiant du produit dont ont été vendues le plus d'unités ? S'il y en a plusieurs, mettez le plus petit.

QUESTION Combien de tickets ne contiennent qu'un produit vendu (quelqu'en soit le nombre d'unités) ?

QUESTION Combien de tickets ne contiennent que des produits vendus en une unité ?

QUESTION Quel est l'heure du dernier ticket enregistré ?

QUESTION Quelle est la valeur en € du premier ticket enregistré ?

La banque de la boulangerie prélève une commission de 1% sur chaque transaction effectuée par carte bleue.

QUESTION Quel est le montant total prélevé par la banque à la boulangerie ce jour là ?

QUESTION Combien y a-t-il de tickets dont la valeur est supérieure ou égale à 10 € ?

systemes

SYS-000.tex

Soit A une matrice carrée telle que toute sous-matrice principale (sous-matrice carrée calée dans le coin supérieur gauche) soit inversible.

On rappelle que l'opération $L_i \leftarrow L_i + \alpha L_j$ effectuée sur les lignes d'une matrice A, revient à effectuer le produit $T(i,j,\alpha) \times A$, où $T(i,j,\alpha)$ est la matrice de transvection égale à l'identité, à l'exception du coefficient (i,j) qui vaut α.

Et l'opération $C_i \leftarrow C_i + \alpha C_j$ effectuée sur les colonnes d'une matrice A, revient à effectuer le produit $A \times T(j,i,\alpha)$.
On peut montrer qu'alors la matrice A admet une décomposition $A = LU$, où L est une matrice triangulaire inférieure et U une matrice triangulaire supérieure. La matrice L est obtenue en appliquant à la matrice identité les opérations sur les colonnes correspondant aux inverses des matrices d'opérations sur les lignes permettant de trianguler A par la méthode de Gauss (sans échange de ligne : toute sous-matrice de A étant principale, on peut montrer que si la matrice A est rendue triangulaire pour ses k premières colonnes, alors le coefficient diagonal de la colonne suivante n'est pas nul et peut être choisi comme pivot).
Plus précisément : si $T_1, \dots, T_p$ sont les p transvections successives à appliquer à A pour la rendre triangulaire supérieure, nous obtenons : $(T_p \times \dots \times T_1) \times A = U$. Il faut alors poser $L = (T_p \times \dots \times T_1)^{-1} = I_n \times T_1^{-1} \times \dots \times T_p^{-1}$.

L'intérêt réside dans le fait qu'on ramène la résolution de $AX = B$ à la résolution de 2 systèmes triangulaires. Ceci s'avère notamment intéressant lorsqu'on a plusieurs systèmes à résoudre associés à la même matrice A.

Nous allons écrire un programme permettant de calculer cette décomposition LU, et le comparer à l'algorithme du pivot de Gauss classique.
Pour cela, nous écrirons les matrices sous forme de tableaux bi-dimensionnels de type `array`, en utilisant le module `numpy`, dont voici quelques exemples d'utilisations :

```
import numpy as np
A=np.array([[1, 2], [3, 4]])
B=np.eye(2)                          # B = I_2
B[1,0]=2
C = A.dot(A) + 2*B.dot(A)            # calcule C = A^2 + 2BA.
```

Nous utiliserons aussi les fonctions définies dans le fichier `random_matrix`, disponible sur le site de la classe, ainsi que la fonction `clock` du module `time` pour chronométrer les temps d'éxécution.

1. Écrire une fonction `trans_ligne` qui à un quadruplet `(A,i,j,alpha)`, où A est une matrice d'ordre n, $i, j \in [\![1,n]\!]$, et $\alpha \in \mathbb{R}$, associe le résultat du produit de la matrice de transvection $T(i,j,\alpha)$ d'ordre n, par A. Attention : par souci d'efficacité, il est préférable de voir cette opération comme une opération entre des lignes de A, plutôt que comme un produit matriciel. On s'arrangera autant que possible pour éviter les calculs inutiles sur les coefficients qu'on sait être nuls.
2. Écrire une fonction `trans_colonne` qui à un quadruplet `(A,i,j,alpha)`, où A est une matrice d'ordre n, $i, j \in [\![1,n]\!]$, et $\alpha \in \mathbb{R}$, associe le résultat du produit de A par l'inverse de la matrice de transvection $T(i,j,\alpha)$ d'ordre n, par A (la remarque de la question précédente est toujours d'actualité).
3. Écrire une fonction `LU` retournant les deux matrices L et U de la décomposition ci-dessus, pour une matrice A donnée.
4. Écrire une fonction `resolution_sup` de résolution d'un système triangulaire supérieur.
5. Écrire une fonction `resolution_inf` de résolution d'un système triangulaire inférieur.
6. Soient $A \in \mathcal{M}_{100}(\mathbb{R})$ et $b_0, \dots, b_{24} \in \mathcal{M}_{100,1}(\mathbb{R})$, choisies aléatoirement (les coefficients étant pris dans $[\![0,100]\!]$).
   Calculer :

(a) le temps de résolution de 25 systèmes associés à A, en calculant une fois pour toutes la décomposition LU de A, puis en résolvant 2 systèmes triangulaires pour chaque second membre B ;
(b) le temps de résolution de 25 systèmes associés à A, en reprenant un pivot complet pour chaque système ;
(c) le temps de résolution de 25 systèmes associés à A, en calculant A^{-1} par la méthode du pivot, une fois pour toutes puis en calculant $A^{-1}B$ pour chaque second membre.

7. Plus généralement, tracer, pour $k \in [\![1,25]\!]$, les 2 courbes correspondant au temps de réponse pour la résolution de k systèmes associés à $A \in \mathcal{M}_{100}(\mathbb{R})$, par décomposition LU, et par calcul de A^{-1}. Commenter.

SYS-001.tex

On considère une fonction réelle f de classe $\mathscr{C}^n$ au voisinage de 0, vérifiant $f(0)=0$ et $f'(0)\neq 0$. On cherche à calculer le développement limité à l'ordre n de f^{-1} au voisinage de 0, en fonction de celui de f.

On représentera un polynôme par la liste de ses coefficients, écrits par degrés croissants. Ainsi, le polynôme $1+2X^2$ pourra être représenté par les listes `[1.,0.,2.]` et `[1.,0.,2.,0.,0.]`.

Les vecteurs et matrices seront représentés en `Python` en utilisant le type `array` de la bibliothèque `numpy`.

Chaque fois que l'on demandera de calculer une complexité, on rédigera ceci *sur papier*, après avoir recopié *à la main* le code de la fonction et numéroté ses lignes, afin de pouvoir y faire référence clairement. Les fonctions ayant une complexité asymptotique clairement sous-optimale seront fortement pénalisées. On ne demande pas de justifier ce dernier point.

Enfin, on s'interdira les spécificités de Python permettant d'éviter d'écrire des boucles. Notamment : la fonction `sum`, la méthode `.dot`, l'écriture d'une liste en compréhension. Chaque boucle sera accompagnée d'un invariant justifiant sa correction.

QUESTION Écrire une fonction `produit(P,Q,n)` prenant en argument deux listes de flottants `P` et `Q` ainsi qu'un entier n et renvoyant la liste des coefficients du produit des polynômes représentés par `P` et `Q`, tronquée à l'ordre n.

Par exemple, avec `P = [1.,1.,1.]` et `Q = [3.,-1.,2.,4.]`, le produit des polynômes représentés par `P` et `Q` est

$$3+2X+4X^2+5X^3+6X^4+4X^5.$$

Ainsi, `produit(P,Q,3)` renverra `[3.0, 2.0, 4.0, 5.0]`, tandis que `produit(P,Q,7)` renverra `[3.0, 2.0, 4.0, 5.0, 6.0, 4.0, 0., 0.]`.

QUESTION *Question manuscrite.* Étudier la complexité asymptotique de `produit(P,Q,n)` en fonction de n.

QUESTION *Question manuscrite.* On note le développement limité de f^{-1} au voisinage de 0 et à l'ordre n comme suit :

$$f^{-1}(t) \underset{t\to 0}{=} x_1 t + x_2 t^2 + \cdots + x_n t^n + o(t^n).$$

En écrivant, au voisinage de 0, $(f^{-1}\circ f)(t)=t$, écrire le système vérifié par le vecteur $(x_1,\dots,x_n)$ et justifier que ce système est triangulaire inférieur. On pourra introduire les coefficients des développements limités des f^j, pour $1\leq j\leq n$, au voisinage de 0 et à l'ordre n.

QUESTION Écrire une fonction `matrice(P)` renvoyant la matrice du système précédent, où `P` est la liste des coefficients de la partie principale du développement limité de f au voisinage de 0.

Par exemple, si $f(t) \underset{t\to 0}{=} 3t-5t^2+3t^4+o(t^4)$, alors on utilisera `P = [0.,3.,0.,-5.,3.]` et l'on remarquera que `P` est alors de longueur 5.

QUESTION *Question manuscrite.* Étudier la complexité asymptotique de `matrice(P)` en fonction de n, où $n+1$ est la longueur de `P`.

QUESTION Écrire une fonction `resoutTI(T,Y)` renvoyant la solution du système $TX=Y$, où `T` est une matrice triangulaire inférieure et `Y` un vecteur de même dimension que `T` n'a de lignes.

QUESTION *Question manuscrite.* Étudier la complexité asymptotique de `resoutTI(T,Y)` en fonction de n, où n est la dimension de `Y`.

QUESTION Écrire une fonction `DLreciproque(P)` renvoyant la liste des coefficients du développement limité en 0 de f^{-1}, de même ordre que celui de f, où `P` est la liste des coefficients de la partie principale du développement limité de f au voisinage de 0.

QUESTION *Question manuscrite.* Étudier la complexité asymptotique de `DLreciproque(P)` en fonction de n, où $n+1$ est la longueur de `P`.

QUESTION *Application.* Écrire une fonction `DLtan(n)` prenant en argument en entier naturel `n` et renvoyant la liste des coefficients de la partie principale de développement limité de la fonction tangente au voisinage de 0 et à l'ordre n.

Commenter le résultat obtenus pour $n=7$.

SYS-002.tex

On étudie dans ce problème le comportement d'un objet déformable (poutre 1D) fixé à une extrémité et soumis à l'action d'une force à son autre extrémité.

On modélise cela par le montage en série de n systèmes ressort-ammortisseur (voir figure **??**) présentant, pour tout $0\leq i<n$, les mêmes masses m, raideurs k et coefficients d'amortissement c. On note $f(t)$ l'intensité de la force appliquée au dernier élément du système.

Pour tout $0\leq i<n$ et tout temps t, on note $u_i(t)$ le déplacement de l'élément n° i par rapport à sa position d'équilibre et l'on introduit le vecteur

$$X(t)=\begin{pmatrix} u_0(t) \\ \vdots \\ u_{n-1}(t) \end{pmatrix}.$$

On peut montrer que X vérifie l'équation différentielle

$$MX'' + CX' + KX = F(t),$$

où

$$M = \begin{pmatrix} m & 0 & \cdots & \cdots & 0 \\ 0 & m & 0 & \cdots & \vdots \\ \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & m & 0 \\ 0 & 0 & \cdots & 0 & m \end{pmatrix}, \quad C = \begin{pmatrix} 2c & -c & 0 & \cdots & 0 \\ -c & 2c & -c & \ddots & \vdots \\ 0 & -c & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 2c & -c \\ 0 & \cdots & 0 & -c & 2c \end{pmatrix}$$

$$K = \begin{pmatrix} 2k & -k & 0 & \cdots & 0 \\ -k & 2k & -k & \ddots & \vdots \\ 0 & -k & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 2k & -k \\ 0 & \cdots & 0 & -k & 2k \end{pmatrix}, \quad F(t) = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ f(t) \end{pmatrix}$$

Les matrices carrées écrites ici sont de dimension $n \times n$ et le vecteur $F(t)$ est de dimension n.

On utilise la méthode d'Euler pour résoudre de manière approchée cette équation différentielle. On fixe donc un pas de temps `dt` > 0 et l'on calcule de proche en proche les

$$X_q = X(q \times \texttt{dt}).$$

Le système est au repos au départ, on suppose donc que $X_0 = X_{-1} = 0$.

Avec

$$H = \frac{1}{\texttt{dt}^2} M + \frac{1}{\texttt{dt}} C + K$$

et, pour tout $q \geq 1$,

$$G_q = \frac{1}{\texttt{dt}^2} M(2X_{q-1} - X_{q-2}) + \frac{1}{\texttt{dt}} C X_{q-1} + F(q \times \texttt{dt}),$$

on montre que, pour tout $q \geq 1$,

$$H \times X_q = G_q.$$

On remarquera que, comme les matrices C et K, H est tridiagonale.

Pour mettre en œuvre la méthode d'Euler, on résout donc plusieurs systèmes de même membre de gauche H. Nous allons utiliser ici la méthode de Jacobi pour résoudre ce système.

QUESTION Écrire une fonction `matrice_H(m,c,k,dt,n)` prenant en argument quatre flottants strictement positifs `m`, `c`, `k` et `dt` ainsi qu'un entier strictement positif `n` et renvoyant la matrice H de dimension `n` $\times$ `n` définie ci-dessus.

On donne la fonction suivante, permettant de calculer le vecteur G_q.

```
import numpy as np

def vecteur_G(m,c,dt,q,X1,X2,f):
    """Renvoie le vecteur Gq de dimension n
    Préconditions : m,c,dt flottants strictements pos
                    q entier positif
                    X1,X2 vecteurs nx1 (X1 : X_q-1 e
                    f fonction réelle, appel en 0(1)"
    n,_ = X1.shape
    G = np.zeros((n,1))
    for i in range(n):
        G[i] = m*(2X1[i]-X2[i])/(dt**2)
    G[0] = G[0] +  (2c * X1[0] - c*X1[1])/dt
    G[-1] = G[-1] +  (2c * X1[-2] - c*X1[-2])/dt + f
    for i in range(1,n-1) :
        G[i] = G[i] + (2c * X1[i] - c*X1[i-1] - c*X1
    return G
```

QUESTION Étudier la complexité temporelle d'un appel de la fonction `vecteur_G(m,c,dt,q,X1,X2,f)`, en fonction d'une grandeur que l'on explicitera.

La méthode du pivot de Gauss étant numériquement peu stable, on lui préfère souvent des méthodes itératives. On expose ici celle de Jacobi.

On résout le système $n \times n$ écrit matriciellement : $H \times Y = G$, à partir d'un vecteur initial Y^0. Pour tout $q \in \mathbb{N}$, on note

$$Y^q = \begin{pmatrix} y_0^q \\ \vdots \\ y_{n-1}^q \end{pmatrix}.$$

Si le vecteur Y^q est construit, la méthode de Jacobi consiste à définir le vecteur Y^{q+1} par, pour tout $0 \leq i < n$,

$$y_i^{q+1} = \frac{1}{h_{i,i}} \left(g_i - \sum_{j \neq i} h_{i,j} y_j^q \right).$$

QUESTION Proposer une simplification de la somme écrite ci-dessus utilisant le fait que la matrice H utilisée ici est tridiagonale (on distinguera les cas $i = 0$ et $i = n-1$).

QUESTION Écrire une fonction `iteration_Jacobi(H,G,Yq)` prenant en argument une matrice H tridiagonale ainsi que deux vecteurs `G` et `Yq`, renvoie le vecteur Y^{q+1} décrit ci-dessus.

On s'interdira ici d'utiliser la multiplication matricielle fournie par la bibliothèque `numpy`.

On considère la norme euclidienne d'un vecteur :

$$\left\| \begin{pmatrix} x_0 \\ \vdots \\ x_{n-1} \end{pmatrix} \right\| = \sqrt{x_0^2 + \cdots + x_{n-1}^2} = \sqrt{\sum_{k=0}^{n-1} x_k^2}.$$

QUESTION Écrire une fonction `carre_norme(X)` prenant en argument un vecteur `X` et renvoyant le carré de sa norme, *i.e.* $\|X\|^2$.

On utilise le critère d'arrêt suivant pour la méthode de Jacobi : on fixe une précision $\varepsilon > 0$, on s'arrête dès que $\|HY_q - G\| \leq \varepsilon \|G\|$ et l'on prend Y_q comme approximation de la solution de ce système.

QUESTION Écrire une fonction `Jacobi(H,G,Y0,eps)` prenant en argument une matrice tridiagonale `H`, deux

vecteurs `G` et `Y0` ainsi qu'un flottant strictement positif `eps` et renvoyant la valeur approchée du système $H \times X = G$ avec la condition initiale `Y0` et le critère d'arrêt donné par $\varepsilon =$ `eps`.

La méthode de Jacobi converge si la matrice H est à diagonale strictement dominante, *i.e.* si pour tout $0 \leq i < n$,

$$|h_{i,i}| > \sum_{j \neq i} |h_{i,j}|.$$

QUESTION Justifier que la méthode de Jacobi est ici convergente.

N'attaquez la dernière question que si vous avez traité toutes les autres questions correctement.

QUESTION On effectue ce calcul sur p valeurs de temps distinctes. On rappelle que l'on résout donc p systèmes de la forme $H \times X_q = G_q$. Toujours en utilisant la méthode de Jacobi, peut-il être préférable de mettre en œuvre une autre stratégie de résolution, du point de vue de la complexité temporelle ?

On pourra supposer que tous les appels de la fonction `Jacobi` *ont la même complexité temporelle.*

tableaux

TAB-000.tex

Écrire une fonction `supprime` prenant deux arguments, un tableau `t` d'entiers et un entier `n`, et renvoyant un tableau dont les éléments sont ceux de `t`, privé des occurrences de `n`.
Par exemple, si `t=[2,1,6,2,8,6,2,1]` et `n=2`, alors le tableau renvoyé sera `[1,6,8,6,1]`.

TAB-001.tex

Étant donnés deux vecteurs `u` et `v` de même taille, de coordonnées $(u_0, u_1, \cdots, u_{n-1})$ et $(v_0, v_1, \cdots, v_{n-1})$, on définit la somme `u+v` de coordonnées $(u_0 + v_0, u_1 + v_1, \cdots, u_{n-1} + v_{n-1})$, et le produit scalaire `u.v`, qui est le réel $\sum_{k=0}^{n-1} u_k v_k$.
On choisit de représenter tout vecteur $(x_0, x_1, \cdots, x_{n-1})$ par le tableau $[x_0, x_1, \cdots, x_{n-1}]$. Écrire alors deux fonctions calculant la somme et le produit scalaire de deux vecteurs.

TAB-002.tex

On représente une matrice par un tableau l'élément i est un tableau contenant les coefficients de la $(i+1)$-ème ligne de la matrice.
Par exemple, si $M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$, M est représentée par le tableau $[[1,2,3],[4,5,6]]$.
Écrire une fonction prenant deux matrices comme argument et retournant leur produit matriciel s'il existe, et un message d'erreur sinon.

TAB-003.tex

QUESTION Écrire une fonction `pascal(n)` ayant comme argument un entier naturel `n` et retournant la `n`$^{\text{e}}$ ligne du triangle de Pascal, sous forme de tableau. Ainsi, pour $n = 2$, cette fonction doit retourner $[1,2,1]$.
Attention : seul l'usage de la formule de Pascal est autorisé ; en particulier, il est interdit d'utiliser la relation $\binom{n}{k} = \frac{n!}{k!(n-k)!}$.

TAB-004.tex

Écrire une fonction qui, étant donné un tableau, retourne ce tableau, dans lequel les occurrences du minimum et celles du maximum ont été échangées.

TAB-005.tex

Écrire une fonction qui, étant donné un tableau d'entiers trié (dans l'ordre croissant), retourne l'indice du premier élément du plus grand sous-tableau constant.

TAB-006.tex

Étant donné un tableau $t = [t_0, \ldots, t_{n-1}]$ d'entiers de longueur n, on appelle sous-tableau croissant de t un tableau $[t_{i_0}, \ldots, t_{i_{k-1}}]$ (donc, de longueur k), avec

- $0 \leq i_0 < i_1 < \cdots < i_{k-1} \leq n-1$;
- $t_{i_0} \leq t_{i_1} \leq \cdots \leq t_{i_{k-1}}$.

On considèrera qu'un tableau vide, donc de longueur 0, est croissant.

On s'intéresse au problème de la recherche du plus long sous-tableau croissant dans un tableau d'entiers.

QUESTION Quelle est la longueur minimale d'un tel plus long sous-tableau croissant ? Quand est-elle atteinte ?

QUESTION Que dire si l'on trouve un plus long sous-tableau croissant de longueur n ?

QUESTION Écrire une fonction `est_croissant(t)` renvoyant un booléen indiquant si le tableau d'entiers `t` est croissant.

Recherche exhaustive.

On cherche d'abord à mettre en œuvre une stratégie naïve : on explore tous les sous-tableaux de t (là, vous devriez vous dire que ce n'est pas une bonne idée) !

Pour faire cela, rappelons nous que prendre un sous-tableau de t correspond à prendre une partie de $[\![0, n[\![$. Or, l'ensemble des parties de $[\![0, n[\![$ est en bijection avec $\{0,1\}^{[\![0,n[\![}$. Un sous-tableau v de t est donc caractérisé par une famille $(e_0, \ldots, e_{n-1}) \in \{0,1\}^{[\![0,n[\![}$, avec, si $0 \leq i < n$, $e_i = 1$ si et seulement si $t[i]$ est présent dans v. Enfin, on peut voir que pour obtenir toutes les familles de $\{0,1\}^{[\![0,n[\![}$, il suffit d'écrire la décomposition binaire sur n bits de tous les entiers entre 0 et $2^n - 1$.

▪ **Exemple** Avec `t = [1,5,3,2]`, de longueur 4, le sous-tableau de `t` décrit par `[0,0,0,0]` est `[]`, celui décrit par `[1,0,0,1]` est `[1,2]` et celui décrit par `[0,1,1,1]` est `[5,3,2]`. ▪

On peut alors penser à implémenter l'idée suivante : on parcourt séquentiellement les entiers entre 0 et $2^n - 1$,

on calcule l'écriture binaire de chaque entier, cela définit un sous-tableau de t. Il ne reste plus qu'à savoir si ce sous-tableau est croissant et à calculer sa longueur.

Commencez par recopier le code suivant dans votre script.

```
def binaire(k,n):
    """Renvoie le tableau de n bits écrivant k en binaire
    Précondition : 0 <= k <= 2**n -1 """
    L = [0]*n
    p = k
    for i in range(n):
        L[n-1-i] = p % 2
        p = p // 2
    return L
```

Soit `k` un entier écrit en binaire avec `n` chiffres : `k` = $\underline{a_{n-1}\dots a_1 a_0}_2$, c'est-à-dire que

$$k = \sum_{j=0}^{n-1} a_j 2^j.$$

QUESTION Écrire un invariant portant sur `L` et `p` dans la boucle `for` de la fonction `binaire(k,n)` et justifier que cette fonction renvoie bien le résultat demandé.

QUESTION Écrire une fonction `longueur(e)` qui prend en argument un tableau `e` contenant des 0 et des 1 et qui renvoie le nombre de 1 dans `e`.

QUESTION Écrire une fonction `extrait(t,e)` qui prend en argument un tableau `t` d'entiers et un tableau `e` de 0 et de 1 et qui renvoie le sous-tableau de `t` désigné par `e`.

QUESTION Écrire une fonction `stc_exhaustif(t)` donnant la longueur du plus grand sous-tableau croissant de `t`.

5.6 Facultatif : programmation dynamique.

Le programme écrit dans la partie précédente n'est pas très efficace (essayez par exemple de le tester sur un tableau de longueur 100). Nous allons adopter une autre statégie.

Toujours avec $t = [t_0, \dots, t_{n-1}]$ un tableau d'entiers de longueur n, on note $m = [m_0, \dots, m_{n-1}]$ le tableau de longueur n tel que, si $0 \leq i < n$, m_i est la taille du plus grand sous-tableau croissant de t dont le dernier élément est t_i.

QUESTION Que vaut m_0 ?

QUESTION Soit $0 \leq i < n-1$, supposons que l'on connaisse t et $[m_0, \dots, m_i]$. Comment calculer m_{i+1} ?

QUESTION Si l'on connaît m, comment obtenir la longueur de la plus grande sous-suite croissante de t ?

QUESTION Écrire une fonction `stc_dynamique(t)` donnant la longueur du plus grand sous-tableau croissant de `t`.

QUESTION Quelle est la différence entre les fonctions écrites dans chaque partie, notamment à l'utilisation ?

TAB-007.tex

Écrire une fonction qui, étant donné un tableau d'entiers de longueur au moins égale à 3, retourne l'indice du premier élément d'un sous-tableau de longueur 3 dont la somme des éléments est maximale.

TAB-008.tex

Soit n un entier naturel. On dit que c'est un *2-palindrome* si son écriture en base 2 est la même qu'elle soit écrite de gauche à droite ou de droite à gauche. Plus précisément, si $n = \sum_{i=0}^{l} a_i 2^i$, avec les a_i dans $\{0,1\}$, et $a_k = 1$, n est un 2-palindrome si pour tout i dans $\{0, \cdots, k\}$, on a $a_i = a_{k-i}$. Par exemple les entiers 3 (11), 5 (101), 7 (111), 9 (1001) et 15 (1111) sont des 2-palindromes. Écrire un programme qui calcule les 2-palindromes n tels que $n \leq 511$.

TAB-009.tex

Le but de cet exercice est de construire la suite ordonnée des entiers de la forme $2^p 3^q 5^r$, où p, q et r sont des entiers naturels.
Cette suite commence par :

valeurs	**1**	**2**	**3**	**4**	**5**	**6**	**8**	**9**	**10**	**12**	**15**	**16**
indice	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
				↑ i_5		↑ i_3		↑ i_2				

Principe :

Le tableau `t` étant en cours de construction, l'élément suivant du tableau sera à choisir parmi les nombres suivants : `2*t[i_2]`, `3*t[i_3]` et `5*t[i_5]`, où i_2 (respectivement i_3, i_5) est l'indice du plus petit élément du tableau n'ayant pas son double (respectivement triple, quintuple) présent dans le tableau.
Dans l'exemple, $i_2 = 8$, $i_3 = 6$ et $i_5 = 4$, et l'élément suivant est donc à choisir parmi 2*9=18, 3*6=18 et 5*4=20.
Le plus petit élément étant 18, c'est l'élément à rajouter au tableau. Les indices concernés par le choix (ici i_2 et i_3) sont à augmenter de 1, ce qui donne le tableau suivant :

valeurs	**1**	**2**	**3**	**4**	**5**	**6**	**8**	**9**	**10**	**12**	**15**	**16**
indice	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
				↑ i_5			↑ i_3		↑ i_2			

1. Écrire, selon ce principe, la fonction qui construit le tableau des n premiers nombres de la forme $2^p 3^q 5^r$.
2. Donner l'invariant de boucle et la démonstration de ce théorème.

TAB-010.tex

Le but de cet exercice est de trier un tableau `t` dont les éléments ne peuvent prendre qu'un "petit" nombre de valeurs (ici, trois valeurs : 0, 1 ou 2).
Par exemple, à partir du tableau initial :

valeurs	2	0	1	0	2	1	1	0	1	2	2	1	0
indice	1	2	3	4	5	6	7	8	9	10	11	12	13

il faut obtenir le tableau final :

valeurs	0	0	0	0	1	1	1	1	1	2	2	2	2
indice	1	2	3	4	5	6	7	8	9	10	11	12	13

Le programme sera itératif (i.e. on utilisera une boucle `for`). Pour trouver comment écrire le corps de la boucle, on suppose que le tableau `t` (de taille n) a été traité jusqu'au rang i, et que sont connus j et k vérifiant :

- tous les éléments du tableau d'indice inférieur ou égal à j sont égaux à 0 ;
- tous les éléments du tableau d'indice supérieur strictement à j et inférieur ou égal à j sont égaux à 1 ;
- tous les éléments du tableau d'indice supérieur strictement à k sont égaux à 2.

0 ⋯ 0	1 ⋯ 1	2 ⋯ 2	x	
1 $\quad$ j	$j+1$ $\quad$ k	$k+1$ $\quad$ i	$i+1$	n

Le prochain élément du tableau à traiter est `t[i+1]`, noté `x`.

• En supposant $1 \leq j < k < i < n$ (c'est-à-dire qu'il y a au moins un 0, un 1 et un 2 placés), quelles sont les modifications à faire sur `t`, i et j si $x = 2$? si $x = 1$? si $x = 0$?

Il se peut qu'il n'y ait pas encore de 0, 1 ou 2 dans la partie de tableau déjà triée. Il faut en tenir compte dans les modifications de `t`, i et j selon les valeurs de x.

• Écrire le traitement complet du tri.

TAB-011.tex

On définit la fonction de Kaprekar comme suit (on la note K). Si $n \in \mathbb{N}$, on écrit n en base 10 (sans zéros inutiles), on prend c le nombre obtenu en écrivant les chiffres de n dans l'ordre croissant et d celui obtenu en écrivant les chiffres de n dans l'ordre décroissant. On pose alors $K(n) = d - c$.

▪ **Exemple** $K(6384) = 8643 - 3468 = 5175$, $K(5175) = 5994$, $K(5355) = 5994$, $K(5994) = 5355$, $K(5355) = 1998$, $K(1998) = 8082$, $K(8082) = 8532$, $K(8532) = 6174$ et $K(6174) = 6174$. ▪

Toutes les suites récurrentes ainsi construites bouclent. Ici, nous avons un cas particulier : la suite est stationnaire.

QUESTION Construire une fonction `Kaprekar(n)` prenant en argument un entier `n` et donnant en sortie les valeurs de la suite décrite plus haut, jusqu'à ce qu'elle ne boucle.

▪ **Exemple** L'appel de `Kaprekar(6384)` devra donner comme résultat
`[6384, 5175, 5994, 5355, 1998, 8082, 8532, 6174]`
▪

Indice : On pourra écrire une fonction convertissant un entier en la liste de ses chiffres en base 10, une fonction convertissant une liste de chiffres en entier, une fonction calculant K et utiliser la méthode `sort`.

TAB-012.tex

Dans un tableau t contenant $2N$ ou $2N+1$ nombres, une médiane est un réel m tel que, si l'on classe t par ordre croissant,

1. m est supérieur ou égal aux N premiers nombres de t ;
2. m est inférieur ou égal aux N derniers nombres de t.

▪ **Exemple**
- La seule médiane de $[1,3,2]$ est 2.
- La seule médiane de $[5,1,2,5,5]$ est 5.
- La seule médiane de $[0,1,0,0]$ est 0.
- L'ensemble des médianes de $[-3,4,1,-1]$ est l'intervalle $]-1,1[$.

▪

QUESTION Écrire une fonction `mediane(t)` qui, à un tableau de nombres `t`, renvoie une médiane de `t`.

On pourra s'inspirer de l'une des deux idées suivantes.
- *Copier* `t`*, lui ôter tant que possible son maximum et son minimum.*
- *Trier* `t` *par ordre croissant.*

TAB-013.tex

QUESTION Écrire une fonction `miroir(t)` qui, à un tableau `t` de nombres, renvoie le tableau contenant les même nombres, renversé.

▪ **Exemple** Avec `t = [5,2,6,3,7]`, `miroir(t)` renverra `[7,3,6,2,5]`. ▪

TAB-014.tex

QUESTION Écrire une fonction `doublon(t)` qui, à un tableau `t`, renvoie un booléen indiquant s'il existe un élément de `t` se trouvant au moins deux fois dans `t`.

TAB-015.tex

Dans un tableau $t = [t_0, \ldots, t_{n-1}]$, on dit que l'on a un record à une position $0 \leq k < n$ si $\forall i < k,\ t_i < t_k$. Par définition, on a toujours un record en position 0.

QUESTION Écrire une fonction `record(t,k)` qui, à un tableau de nombres `t` et un entier `k`, renvoie le booléen `True` si `t` admet un record en position `k`, `False` sinon.

QUESTION Écrire une fonction `nb_records(t)` qui, à un tableau de nombres `t`, renvoie le nombre de records dans `t`.

TAB-016.tex

On rappelle que, si un entier naturel $p \geq 2$ n'a aucun diviseur inférieur à $\sqrt{p}$ (sauf 1), alors p est premier.

QUESTION Écrire une fonction `crible(p)` qui, à un entier naturel `p`, renvoie le tableau des nombres premiers inférieurs ou égaux à p^2, triés par ordre croissant.

TAB-017.tex

QUESTION Écrire une fonction Python renvoyant un indice du maximum d'un tableau de nombres T.

QUESTION Écrire un invariant de boucle pour cet algorithme. Démontrer que l'algorithme précédent donne le bon résultat.

QUESTION Donner la complexité dans le pire des cas du calcul d'un indice du maximum de T par cette fonction, en fonction de la longueur de T, que l'on notera n.

TAB-018.tex

Soit `t` $= [t_0, \dots, t_{n-1}]$ un tableau de nombres de longueur $n \in \mathbb{N}^*$, soit $k \in \mathbb{N}^*$ et $i \in [\![0, n+2-2k[\![$.

On dit que `t` possède une *pyramide* de *hauteur* k en position i si

$$t_i < t_{i+1} < \dots t_{i+k-1}$$

et si

$$t_{i+k-1} > t_{i+k} > \dots t_{i+2k-2}.$$

Par exemple, il y a une pyramide de longueur 1 en tout élément de `t`, et il y a une pyramide de longueur 2 en position i si

$$t_i < t_{i+1} \quad \text{et} \quad t_{i+1} > t_{i+2}.$$

Ainsi, avec

$$\texttt{t} = [-1, 0, 4, 2, -3, 0, 5, 1],$$

`t` a une pyramide de hauteur 3 en position 0, des pyramides de hauteur 2 en position 1 et 5 et des pyramides de hauteur 1 en toute position.

QUESTION Écrire une fonction `nb_pyramides_2(t)` renvoyant le nombre de pyramides de hauteur 2 présentes dans le tableau de nombres `t` passé en argument.

QUESTION Écrire une fonction `pyramide_a_partir(t,i)` prenant en argument un tableau de nombres `t` et un indice `i` et renvoyant la taille de la pyramide du tableau `t` débutant en position `i`.

QUESTION Écrire une fonction `plus_haute_pyramide(t)` renvoyant la taille de la plus haute pyramide présente dans le tableau de nombres `t` passé en argument. On pourra utiliser la fonction `pyramide_a_partir(t,i)` de la question précédente, même si cette question n'a pas été résolue.

TAB-019.tex

Soit `t` $= [t_0, \dots, t_{n-1}]$ un tableau de nombres de longueur $n \in \mathbb{N}^*$, soit $0 \leq i, j \leq n-1$.

On dit qu'il y a un *ascension* dans le tableau `t` aux indices i et j si

$$i < j \quad \text{et} \quad t_i < t_j.$$

La *hauteur* de cette ascension est alors $t_j - t_i$.

On propose la fonction suivante.

```
1  def plus_haute_ascension(t):
2      """Plus haute ascension de t, ou 0 s'il n'y en a
3      Précondition : t est un tableau de nombres"""
4      M = 0
5      n = len(t)
6      for i in range(n):
7          m = 0
8          for j in range(i+1,n):
9              if t[j] - t[i] > m :
10                 m = t[j] - t[i]
11         if m > M :
12             M = m
13     return M
```

QUESTION Montrer que « si `m` $>$ 0, alors `m` est la hauteur de la plus haute ascension de `t[i:j]`, sinon $m = 0$ » est un invariant de boucle pour la boucle `for` portant des lignes 8 à 10 de la fonction `plus_haute_ascension(t)`.

QUESTION Justifier qu'un appel de la fonction `plus_haute_ascension(t)` renvoie bien le résultat demandé, à l'aide notamment d'un invariant.

QUESTION Écrire une fonction `nb_ascensions(t)` renvoyant le nombre d'ascensions présentes dans le tableau de nombres `t` passé en argument.

TAB-020.tex

On considère la fonction suivante.

```
def suite(n):
    """Précondition : n entier naturel"""
    u = 2
    for i in range(n) :
        u = 4 * u - 3
    return u
```

QUESTION Montrer que « $u = 4^i + 1$ » est un invariant d'entrée de boucle, pour la boucle for de la fonction `suite(n)`. En déduire le résultat renvoyé par cette fonction.

TAB-021.tex

On considère la fonction suivante.

```
def mystere(L,M):
    """Préconditions : L tableau de nombres, M nombre
    S = 0
    n = len(L)
    for i in range(n) :
        if L[i] > M :
            S = S + L[i]
    return S
```

QUESTION Que renvoie un appel de la fonction `mystere(L,M)` ? On justifiera la réponse, à l'aide notamment d'un invariant.

TAB-022.tex

On considère la fonction suivante.

```
def mystere(n):
    """Préconditions : n entier positif"""
    L = []
    c = 0
    while c ** 2 <= n :
        L.append(c**2)
        c = c+1
    return L
```

QUESTION Montrer que, si `n` est un entier, un appel de la fonction `mystere(n)` renvoie un résultat, à l'aide d'un variant.

QUESTION Que renvoie un appel de la fonction `mystere(n)` ? On justifiera la réponse, à l'aide notamment d'un invariant.

TAB-023.tex

Dans un tableau de nombres $t = [t_0, \ldots, t_{n-1}]$ de longueur n, la position $1 \leq i < n-1$ est dit *sous l'eau* s'il existe $k \in [\![0, i[\![$ et $\ell \in [\![i+1, n[\![$ tels que

$$t_k > t_i \quad \text{et} \quad t_\ell > t_i.$$

QUESTION Écrire une fonction `nb_sousleau(t)` prenant en argument un tableau de nombres t et renvoyant le nombres d'indices sous l'eau de ce tableau.