Exercice 1
Ouvrir le fichier `ds-pt.csv` et observer le format de chaque ligne.

1. A partir de ce fichier, créer 4 listes : date, debut, fin, matiere.
2. Afficher toutes les matières de devoirs se déroulant au mois de novembre.
3. Sans faire référence aux matières, trouver tous les devoirs dont la durée n'est pas 4H.

Exercice 2
Un bac (de pièces industrielles) contient u pièces défectueuses et z pièces de qualité suffisantes. Dans la suite on retourne 1 dans le cas d'une pièce défectueuse et 0 sinon.

1. On considère le programme suivant :

```
import random

def tirer(u, z):
    x = random.randint(1, u + z)
    if x <= u:
        return 1
    else:
        return 0
```

   Que fait ce programme ?
2. Ecrire un programme qui prend comme argument u, z et n et qui retourne une liste donnant le résultat de n tirages de pièces avec remise. Le tester.
3. Ecrire un programme qui prend comme argument u, z et n et qui retourne une liste donnant le résultat de n tirages de pièces sans remise. Le tester.
4. Ecrire un programme qui prend comme argument u, z et N et qui retourne la moyenne du nombre de pièces défectueuses sur N tirages de 15 pièces.

   Question subsidiaire : comment calculer le nombre de 1 dans une liste composée de 0 et de 1 ?

Exercice 3
1. Tracer sur un même graphique les courbes représentatives des fonctions $f : x \mapsto \frac{1}{1+x^2}$ et $g : x \mapsto \arctan(x)$. On prendra l'intervalle $[-2, 2]$ et on effectuera un tracé comportant 250 points.
2. Dans la suite, pour approximer la valeur de la dérivée de h en x, on utilise l'approximation
$$h'(x) \approx \frac{h\left(x + \frac{\mathrm{d}x}{2}\right) - h\left(x - \frac{\mathrm{d}x}{2}\right)}{\mathrm{d}x}$$
   où $\mathrm{d}x$ est une quantité que l'on choisira petite en pratique.
   (a) Implémenter la fonction `d1(h, x, dx)` qui retourne, pour une fonction `h`, et deux nombres `x` et `dx`, l'approximation de $h'(x)$ donnée par la formule précédente.
   (b) On prend 10^{-2} comme valeur de `dx` dans cette question. Tracer la courbe représentative de la fonction $x \mapsto d1(g, x, dx)$ pour les mêmes valeurs de x qu'à la question 1. Comparer avec le tracé de f.
   (c) Définir `dn(h, x, dx, n)` de manière récursive, pour obtenir une approximation de $h^{(n)}(x)$.
   Indication : on pourra créer une fonction locale à utiliser pour l'appel récursif.
   (d) Bonus : trouver ainsi les coefficients du développement limité de g en 0 à l'ordre 9 et comparer au développement théorique.

Exercice 4
On note H l'ensemble des nombres de la forme $2^m \times 3^n \times 5^p$ où m, n, p sont des entiers naturels. On a donc
$$H = \{1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, \ldots\}$$
Le but de cet exercice est de calculer des éléments de H.

1. Donner une fonction récursive `HQ(n)` qui retourne un booléen indiquant si l'entier n est un élément de H.
   Indication : traduire la forme des éléments en terme de divisibilité.
2. Ecrire une fonction `premiers_H(N)` qui retourne la liste des N premiers éléments de H et la tester avec $N = 100$.
3. On va essayer de réduire la quantité de calcul.
   (a) Construire la liste des différences entre deux éléments consécutifs de H, pour les 100 premiers éléments. On observe que cette différence à tendance à devenir de plus en plus grande. La situation empire encore pour les 200 premiers éléments.
   (b) Si $n \in H$, que peut-on dire de $2n, 3n$ et $5n$?
   (c) La remarque précédente nous permet de proposer un nouvel algorithme de construction de H.
   — On part d'une liste contenant le seul élément 1.
   — A chaque étape, on recherche le prochain élément h de H sous la forme $2n, 3n, 5n$, où n est un élément déjà construit.
   h doit être strictement plus grand que tous les éléments trouvés jusqu'alors, et doit être le plus petit des éléments de H possédant cette propriété
   — On ajoute h à la liste en construction et on recommence (l'élément maximum vient de changer).
   Implémenter cet algorithme dans une fonction `construit_H(N)`

4. BONUS : Donner une version itérative de `HQ` (après test, cette version semble légèrement plus rapide)

Exercice 5

Pour une entier $n \in \mathbb{N}$, on note $prod(n)$ le produit des chiffres de son écriture en base 10.

1. Ecrire deux fonctions calculant $prod(n)$, l'une itérative, l'autre récursive.
2. Etant donné un entier, on définit la suite $u_0 = n$, $u_{k+1} = prod(u_k)$. On admet que la suite (u_k) finit toujours par prendre une valeur entre 0 et 9 (et donc stationne).
   On appelle persistance de n le premier rang k pour lequel $u_k \in [0, 9]$. Ecrire une fonction `persistance` d'argument n et retournant la persistance de l'entier n.
3. Trouver le plus petit entier de persistance 5.

Exercice 6

On souhaite représenter en python des ensembles finis d'entiers naturels de la forme $\{x_1, \dots, x_n\}$.

1. A un tel ensemble on associe une liste P de booléens : $P[i]$ vaut **True** si i est présent dans l'ensemble. Par exemple à $\{1, 3, 4\}$, on peut associer la liste [**False, True, False, True, True**]
   (a) Expliquer pourquoi une telle représentation n'est pas unique.
   (b) Ecrire une fonction **card(P)** où P est une liste de booléens associée à un ensemble et qui renvoie le cardinal de l'ensemble.
   (c) Ecrire une fonction **BtoE(P)** renvoyant la liste des entiers contenue dans l'ensemble associé à P.
   (d) Ecrire une fonction **EtoB(L)** renvoyant la liste (la plus courte possible) des booléens associée à l'ensemble des entiers contenus dans L.
   (e) Ecrire une fonction **Bunion** prenant deux listes de booléens et renvoyant la liste de booléens de la réunion.
   En déduire une fonction **union** renvoyant la liste réunion de deux listes d'entiers.
2. Une deuxième méthode est de coder $\{x_1, \dots, x_n\}$ sous la forme $\sum_{k=1}^{n} 2^{x_i}$
   Ecrire une fonction pour coder un ensemble et une fonction pour le décoder.

Exercice 7

On pose $M = 20$ et $m = 10$. A tout nombre c on associe la suite $(u_n)_n$ définie par

$$u_0 = 0 \text{ et } u_{n+1} = u_n^2 + c \text{ pour } n \geqslant 0$$

S'il existe, on note k le plus petit entier tel que on ait $0 \leqslant k \leqslant m$ et $|u_k| > M$.
On définit alors la fonction f par$f(c) = k$ s'il existe et $f(c) = m + 1$ sinon.

1. Donner un code python définissant f.
2. Tracer l'allure de la courbe représentative de la fonction f sur $[-2, 2]$, en créant une liste `LX` de 401 valeurs équiréparties entre -2 et 2 inclus.
3. Construire la matrice des valeurs de $f(x+iy)$ où x prend 101 valeurs entre -2 et 0.5 et y prend 101 valeurs entre -1.1 et 1.1.
   Rappel, le complexe $x + iy$ s'écrit en python `x + y * 1j` ou `complex(x, y)`.
   Tracer l'image que code ce tableau. On pourra utiliser les images rendues pas `matshow` de la bibliothèque `matplotlib.pyplot`

Exercice 8

On définit la suite $(t_n)_{n\in\mathbb{N}}$ par $t_0 = 0$ et pour $n \in \mathbb{N}$ $t_{2n} = t_n$, $t_{2n+1} = 1 - t_n$.

1. Expliquer rapidement pourquoi la suite (t_n) est à valeurs dans $\{0, 1\}$.
2. Ecrire une fonction récursive `t(n)` qui renvoie la valeur de t_n.
3. Ecrire une fonction itérative `liste_t(N)` qui renvoie la liste $[t_0, \dots, t_N]$ sans faire appel à la fonction précédente.
4. Ecrire une fonction qui converti retourne la chaîne dont les caractères sont $t_0, t_1, \dots, t_N$.
5. La question suivante est : pour un N fixé, quels sont les motifs qui se répètent le plus souvent dans la chaîne précédente.
   Ecrire une fonction `occurrences(txt, motif)` qui compte le nombre de fois où la chaîne `motif` apparaît dans `txt`. Deux occurrences ne sont pas forcément disjointes.
6. Trouver le(s) motif(s) de moins de 6 bits qui apparaît le plus souvent parmi les 10^5 premiers termes de (t_n). Quels sont ceux qui sont absents ?
7. Proposer une construction de suite binaire telle qu'elle contienne tout motif fini.

Exercice 9

Une variable aléatoire V_n prend les valeurs $[0, 1], [-1, 0], [1, 0], [0, -1]$ avec une loi uniforme. On définit la suite (X_n) par $X_0 = [0, 0]$ et $X_{n+1} = X_n + V_n$.

1. Définir une fonction **tirage** représentant la variable V_n. On pensera à utiliser numpy pour la valeur de retour, en vue de la suite.
2. Tracer la ligne reliant les points de coordonnées $X_0, \dots, X_n$ pour $n = 10, 100$ et 1000.
3. En moyenne, combien de fois le point X_k est-il à l'origine quand $k \in [\![1, 1000]\!]$?
4. On appelle rayon d'un marche le longueur N (la ligne brisée reliant les N premiers points visités) le rayon du plus petit disque contenant cette marche. On le note $R(N)$ Déterminer les rayons moyens des marches de longueur $10, 100, 1000$.
   Proposer un ordre de grandeur de $R(N)$ en fonction de N