Exercice 1

1. Testez **list(str(140))** puis **''.join(['0', '4', '1'])**. Commentez puis écrivez une fonction **R** d'argument n renvoyant le nombre retourné.
2. Ecrivez une fonction F d'argument n renvoyant la distance entre n et **R(n)**. Tester avec $n = 1024$, $n = 4201$.

   Soit $u_0 = a$ et $u_{n+1} = F(u_n)$
3. Ecrivez une fonction **LU** d'arguments a et N renvoyant la liste des $N + 1$ premiers termes de la suite $(u_n)_{n\in\mathbb{N}}$.
4. On appelle "longueur" d'un nombre a (quand elle existe) l'indice du premier terme de (u_n) valant 0, sachant que $u_0 = a$. Trouver les nombres composés de 3 chiffres de longueur maximale.
5. Vérifier qu'il est impossible de traiter la question précédente pour les nombres à 4 chiffres.
6. Trouver tous les nombres à 4 chiffres tels que la suite (u_n) soit périodique.

Exercice 2

Un nombre n de p chiffres est dit narcissique ssi la somme de ses chiffres à la puissance p égale n.

1. On considère la fonction

```
def chiffres(n):
    L = []
    while n != 0:
        L.append(n % 10)
        n = n // 10
    return L[::-1]
```

   Quelle est la valeur de retour ? Expliquer.
2. Vérifier que 93084 est narcissique.
3. Ecrire une fonction **narcisse** d'argument n renvoyant un booléen indiquant si n est narcissique.
4. Efficher tous les nombres narcissiques compris entre 0 et 10 000
5. Ecrire la fonction **narcisse_suiv(n, N)** renvoyant le premier nombre narcissique supérieur à n et inférieur à N.
6. Déterminer les nombres entre 1 et 10 000 qui sont à la fois narcissiques et premiers.

Exercice 3

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 4

Données : un fichier csv contenant 365 lignes de la forme "temperature ;date" où la date est au format aaaa-mm-jj

1. Créer une liste contenant toutes les températures contenues dans le fichier.
2. Calculer le minimum, le maximum et la moyenne des températures sur l'année. Afficher la courbe des températures.
3. Donner le nombre de fois où l'étendue de température sur 7 jours consécutifs a dépassé 12K.
4. Donner la moyenne des écarts de température entre deux jours consécutifs sans écrire de boucle. On pourra utiliser numpy.

Exercice 5

On définit $E_n = \{0, 1, \ldots, n-1\}$ et A une partie de E_n. Pour $k \in A$, on note $\sum_{k\in A} \exp\left(\frac{2ik\pi}{n}\right)$ la somme des racines n-ièmes de l'unité..

On cherche à maximiser $|S_A|$.

1. Ecrire une fonction **racine(n, k)** qui renvoie la valeur de $\exp\left(\frac{2ik\pi}{n}\right)$.
2. Combien de sous-ensembles A deux à deux distincts peut-on tirer dans E_n ? Tester pour $n = 20,\ n = 50,\ n = 100$.

3. Ecrire une fonction **tirer(n, k)** qui renvoie une liste de k entiers aléatoires compris entre 0 et $n-1$ et distincts 2 à 2. On pourra utiliser le module **randint**.
4. Ecrire une fonction **somme(n)** qui renvoie le maximum de $|S_A|$ et la valeur de k pour laquelle ce maximum est atteint.

Exercice 6

On veut créer un programme donnant le nombre de façon de payer n (ie. donner le montant exact sans retour de monnaie) avec des jetons distincts. Par exemple il y a 4 façons de payer 5 avec des jetons de $1, 2, 5$:

— $1 \times 5 + 0 \times 2 + 0 \times 1$
— $0 \times 5 + 1 \times 2 + 3 \times 1$
— $0 \times 5 + 2 \times 2 + 1 \times 1$
— $0 \times 5 + 0 \times 2 + 5 \times 1$

1. Ecrire une fonction **P5(n)** qui renvoie le nombre de façon de payer n avec uniquement des jetons 5.
2. Ecrire une fonction **P25(n)** qui renvoie le nombre de façon de payer n avec des jetons 2 et 5. $P25$ utilisera une boucle où k représentera le montant payé avec des jetons de 2 et **P5(n - k)** pour le montant restant.
3. Ecrire **P125**
4. Ecrire une fonction **P(V, n)** qui renvoie le nombre de façons de payer n avec les jetons dont la valeur est contenue dans $V = [a, b, c, \dots]$. **P(V, n)** utilisera la fonction **P([b, c, ...], n - k)**.

Exercice 7

On considère l'ellipse paramétrée par $\begin{cases} x(t) = a\cos(t) \\ y(t) = b\sin(t) \end{cases}$ avec a, b non nuls.

1. Ecrire les fonctions **X** et **Y** prenant en argument (t, r) et renvoyant respectivement $r\cos(t)$ et $r\sin(t)$.
2. Tracer l'ellipse pour $a = 2,\ b = 1$. On prendra $t \in [0, 2\pi]$ et on affichera 200 points.
3. Ecrire la fonction **BLint** prenant en argument $(a, b, T = [t_0, \dots t_{n-1}])$ et renvoyant la liste des coordonnées $(x(t_i), y(t_i))$ pour $i \in [\![0, n]\!]$ (on prendra $t_0 = t_n$).
   Tester pour $a = 2,\ b = 1,\ T = [0.5, 1.5, 3, 4, 5, 6]$.
4. Pour tous paramètre t, s avec $|t - s| < \pi$, les coordonnées du point d'intersection des tangentes respectives en $M(t)$ et $M(s)$ sont données par $\frac{1}{\cos\frac{s-t}{2}}\left(x\left(\frac{s+t}{2}\right), y\left(\frac{s+t}{2}\right)\right)$.
   Ecrire une fonction **Bext** prenant les mêmes arguments que **BLint** et renvoyant la liste des coordonnées des points d'intersections des tangents aux points de paramètres t_i et $t_i i + 1$ (on prendra $t_n = t_0$).
   Tester avec les mêmes valeurs des paramètres qu'à la question précédente.

Exercice 8

Pour $l = (a_0, \dots a_n)$ on note $S(l) = \frac{1}{a_0}\left(1 + \frac{1}{a_1}\left(\dots(1 + \frac{1}{a_n})\right)\right)$.

Par exemple $S((1,3,2)) = \frac{1}{1}\left(1 + \frac{1}{3}\left(1 + \frac{1}{2}\right)\right) = \frac{3}{2}$.

1. Implémenter une fonction **S** qui prend comme argument une liste l et renvoie $S(l)$. Tester avec $[1, 3, 2]$.
2. Définir une fonction **f** qui prend comme argument un réel x et renvoie $\begin{cases} 0 \text{ si } x \leqslant 0 \\ 1 + \lfloor \frac{1}{x} \rfloor \text{ si } x > 0 \end{cases}$.
3. Soit x un réel strictement positif. On pose $a_0 = f(x)$ et $\forall n \in \mathbb{N} a_{n+1} = f(a_0 a_1 \dots a_n (x - S(($
   Définir une fonction **Srec** qui prend comme argument x et n et qui renvoie la liste des $n+1$ premiers termes de (a_n).
   Tester avec $Srec(1, 5)$ puis $S(Srec(1, 5))$. De même avec $x = \frac{5}{7}$ et $n = 4$. Qu'observe-t-on ?

Exercice 9

1. Soit L une liste de longueur multiple de 3. On écrit $L = L1 + L2 + L3$ avec L_1, L_2, L_3 de même longueur. Ecrire une fonction **alterner(L)** qui, retourne la liste M construite successivement en prenant alternativement des éléments de L_3, L_2, L_1.
   Vérifier que **alterner([1, 2, 3, 4, 5, 6])** envoie bien $[5, 3, 1, 6, 4, 2]$.
2. Combien de fois faut-il appliquer **alterner** à L de longueur 78 pour que le premier élément se retrouve à nouveau en première position ?
   Répondre à la même question pour chaque autre élément.
3. Tracer l'évolution des positions des 3 premiers éléments de la liste au fur et à mesure que l'on mélange la liste.

Exercice 10

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 $\{1, 2, 3\}$, on associe **[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 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 $\{x_1, \ldots, x_n\}$ sous la forme $\sum_{k=1}^{n} 2^{x_i}$

   Ecrire une fonction pour coder un ensemble et une fonction pour le décoder.