Exercice 1
Ecrire une fonction `racine(a,n,e)` prenant en paramètre un nombre `a` > 0 et un entier `n` et qui applique la méthode de Newton pour retourner une valeur approchée à e près de $\sqrt[n]{a}$ (on prendra pour point initial $u_0 = a$). Quel critère d'arrêt peut-on considérer ici ?.

Exercice 2
Épreuve type - Banque PT. On considère la fonction g définie sur $[0, 2[$ par

$$g(x) = \begin{cases} x & \text{pour } 0 \leqslant x < 1 \\ 1 & \text{pour } 1 \leqslant x < 2 \end{cases}$$

1. Définir la fonction `g`. Tracer sa courbe représentative sur $[0, 2[$, c'est-à-dire la ligne brisée reliant les points $(x, g(x))$ pour x variant de 0 à 1.99 avec un pas de 0.01.
2. Définir une fonction f donnée de manière récursive sur $[0, +\infty[$ par

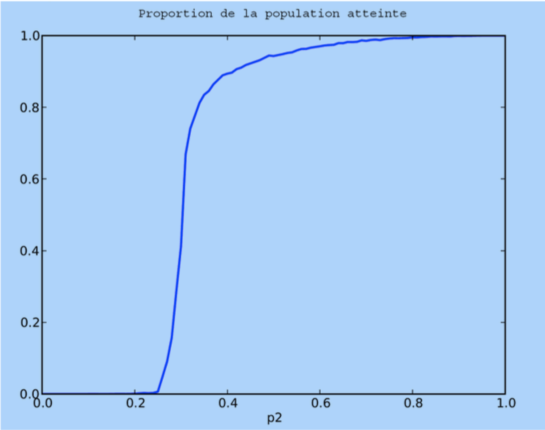
$$f(x) = \begin{cases} g(x) & \text{pour } 0 \leqslant x < 2 \\ \sqrt{x}\, f(x-2) & \text{pour } x \geqslant 2 \end{cases}$$

3. Tracer la courbe représentative de f sur $[0, 6]$.
4. Écrire les instructions permettant de calculer, à 10^{-2} près, la plus petite valeur $\alpha > 0$ telle que $f(\alpha) > 4$.

Exercice 3
D'après Écrit Mines-Ponts 2016.

Dans la simulation numérique de la propagation d'une pandémie on a tracé pour plusieurs valeurs de :

`p2`, probabilité qu'une personne saine en contact avec la maladie soit infectée,
La proportion `x_atteinte` de la population atteinte par l'épidémie au cours de la simulation.



On appelle seuil critique de pandémie la valeur de `p2` à partir de laquelle plus de la moitié de la population a été atteinte par la maladie durant la simulation. On suppose que les valeurs de `p2` et `x_atteinte` utilisées pour tracer la courbe de la figure 3 ont été stockées dans deux listes de même longueur `Lp2` et `Lxa`. Écrire en Python une fonction `seuil(Lp2, Lxa)` qui détermine par dichotomie un encadrement `[p2cmin, p2cmax]` du seuil critique de pandémie avec la plus grande précision possible. On supposera que la liste `Lp2` est croissante de 0 à 1 et que la liste `Lxa` des valeurs correspondantes est croissante.

Exercice 4
Tracé de la fractale de Newton.

La convergence d'une suite définie par la méthode Newton est un problème très difficile. L'ensemble des points u_0 pour laquelle la suite (u_n) converge est un ensemble très complexe.

On peut s'en convaincre en étendant la suite au plan complexe (c'est possible notamment lorsque f est un polynôme et f' son polynôme dérivée) :

$$(u_n) : \begin{cases} u_0 \in \mathbb{C} \\ u_{n+1} = u_n - \dfrac{f(u_n)}{f'(u_n)} \end{cases}$$

Illustrons le phénomène en prenant pour f le polynôme $z \mapsto z^3 - 1$ (et donc

$f' : z \mapsto 3z^2$) :

$$u_0 \in \mathbb{C} \qquad \forall n \in \mathbb{N} \quad u_{n+1} = u_n - \frac{u_n^3 - 1}{3u_n^2} = \boxed{\frac{2u_n^3 + 1}{3u_n^2} = u_{n+1}}$$

La fonction f a 3 racines dans $\mathbb{C}$: 1, $j = \exp(2i\pi/3)$, $\overline{j} = \exp(4i\pi/3)$. Nous allons représenter dans un domaine carré du plan complexe l'ensemble des points initiaux $u_0 \in \mathbb{C}$ pour lesquels cette suite converge :

$$\mathscr{D} = \{u_0 \in \mathbb{C} \mid (u_n) converge\}$$

La méthode de Newton se généralise très bien au plan complexe, et si (u_n) converge, alors nécessairement c'est vers l'une des 3 racines de f : $1, j, \overline{j}$.

Il faut choisir un critère approximatif de convergence. Prenons par exemple :
Donné $u_0 \in \mathbb{C}$, u_{30} est à distance $< tol = 0.01$ de l'une des 3 racines.
C'est à dire, en notant les racines $z_0 = 1$, $z_1 = j = \frac{-1+i.\sqrt{3}}{2}$, $z_2 = \overline{j}$:

$$\boxed{\min(|u_{30} - z_0|, |u_{30} - z_1|, |u_{30} - z_2|) < 0.01}$$

1. Importer les modules `numpy` et `matplotlib.pyplot`. Définir les constantes z_0, z_1, z_2 et la fonction $F(z) = \dfrac{2z^3 + 1}{3z^2}$.
2. Définir deux variables globales `N = 30` et `tol = 0.01`. Ecrire une fonction `g` prenant en paramètre un complexe z et qui retourne :
   - S'il existe, le plus petit rang $k \in [[1, N-1]]$ tel que, si (u_n) désigne la suite obtenue par la méthode de Newton avec $u_0 = z$:

     $$\min(|u_k - z_0|, |u_k - z_1|, |u_k - z_2|) < tol$$

   - Sinon, retourne `N`.
3. Définir une variable globale `K=500`. Créer un tableau `T` d'un maillage de $K \times K$ points du plan complexe régulièrement espacés dans le domaine carré $\{z \in \mathbb{C} \,||Re(z)| \leqslant 2, |Im(z)| \leqslant 2\}$.
4. Créer le tableau des images des éléments de `T` par la fonction g. Puis le représenter graphiquement à l'aide de la fonction `imshow` de `matplotlib.pyplot`. Interpréter le graphique obtenu.

Indications. Le nombre complexe i s'obtient par l'expression `1j`, le module par la fonction `abs`.

3) On conseille d'utiliser les fonctions `linspace` et `meshgrid` du module `numpy`. Exemple d'utilisation de `meshgrid` :

```
>>> x = [-1,-2] ; y = [1,2,3]
>>> X, Y = np.meshgrid(x,y)
>>> X, Y
array([[-1,-2],
       [-1,-2],
       [-1,-2]])
array([[1, 1],
       [2, 2],
       [3, 3]]
```

4) : on conseille de vectoriser la fonction `g` : `gv = np.vectorize(g)` pour l'appliquer à `T`.

```
# Création de la grille
x = np.linspace(-2,2,K)
y = np.linspace(2,-2,K)
X,Y = np.meshgrid(x,y)
T = X + Y * 1j
# Vectorisation de g pour l'appliquer à un tableau et tracé
:
gv = np.vectorize(g)
Tab = gv(T)
plt.imshow(Tab, extent=[-2,2,-2,2])
plt.show()
```