On commence par importer deux «modules» à l'aide des instructions suivantes :

```
import numpy as np
import matplotlib.pyplot as plt
```

Le module `numpy` (appelé ici avec l'alias `np`) permet d'utiliser de nombreuses fonctions et constantes mathématiques :

```
>>> np.sqrt(3)
1.7320508075688772
>>> np.log(2)  # ln(2)
0.6931471805599453
>>> np.exp(1.5)
4.4816890703380645
>>> np.pi
3.141592653589793
>>> np.e
2.718281828459045
>>> np.cos(np.pi / 2)
6.123233995736766e-17
```

Pour le moment, il convient de connaître de ce module l'instruction :

`np.linspace(start, stop, num)`

qui revoie un tableau de `num` valeurs (50 par défaut) réparties régulièrement de `start` à `stop` (compris) :

```
>>> np.linspace(0, 3, 5)
array([0.  , 0.75, 1.5 , 2.25, 3.  ])
>>> np.linspace(0, 1, 11)
array([0. , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1. ])
```

L'application d'une fonction `f` à un tel tableau `T` produit un tableau `f(T)` constitué des images de chacun des éléments du tableau `T` par la fonction `f` :

```
>>> T = np.linspace(0, 3, 4)
>>> T
array([0., 1., 2., 3.])
>>> np.exp(T)
array([ 1.        ,  2.71828183,  7.3890561 , 20.08553692])
>>> np.floor(np.exp(T))
array([ 1.,  2.,  7., 20.])
```

Exercice 1

Soit la fonction f qui à $x > 0$ associe $\lfloor \ln(x) \rfloor$.
Définir une fonction python `images(n)` donnant le tableau des valeurs des $f(k)$ pour $k \in [\![1, n]\!]$.
Définir une fonction python `imagesbis(n)` donnant le tableau des valeurs des $\frac{f(k)}{k}$ pour $k \in [\![1, n]\!]$.

Pour tracer des courbes, on utilise le module `matplotlib.pyplot` importé ici avec l'alias `plt`. Par défaut, au moment de l'importation, Python crée une «feuille de figures» numérotée 1 dans laquelle on peut insérer une ou plusieurs figures. On peut créer d'autres pages par la commande `plt.figure(n)` où `n` est entier et on navigue entre les pages par la même commande. Par défaut, on est dans la feuille 1 et les tracés sont toujours faits dans la figure courante. On détruit la feuille courante par la commande `plt.close()`. On affiche les différentes pages de figures dans des fenêtres interactives extérieures par `plt.show()`.

Exercice 2

Expérimenter ces instruction avec le code suivant :

```
plt.title("la fonction cos")
X = np.linspace(0, 2*np.pi)
plt.plot(X,np.cos(X),'r:s')
plt.show()
```

La commande standard pour insérer une courbe dans la feuille courante est :

```
plt.plot ( tabx, taby, ch_tracé , label = "nom_de_la_courbe" )
```

où `tabx, taby` constituent les deux tableaux donnant les coordonnées des points à relier, le paramètre optionnel `label` permet de donner un nom à la courbe pour la légende et `ch_tracé` est une chaîne de caractères précisant le type du tracé *via* trois informations :

- le premier caractère de `ch` indique la couleur via la première lettre du nom de la couleur en anglais b(lue),r(ed),g(reen),c(yan),m(agenta),y(ellow),k(black),w(hite).
- les caractères suivant indiquent la façon de relier les points ('-' pour une ligne, '- -' pour des pointillés tirets, ' :' pour des pointillés points).
- le dernier caractère indique comment sont signalés sur le graphique les points de la courbe donnés explicitement *via* les listes `listex` et `listey` ('o' pour des petits cercles, 's' pour des petits carrés, 'v' pour des triangles, des pixels par défaut).

Par défaut, le tracé se fait dans la couleur courante (qui change à chaque plot) et en trait plein. Les noms des courbes apparaissent dans la légende générée par la commande `plt.legend()`. On peut forcer le tracé d'une grille en filigrane dans la figure par `plt.grid(True)`.

Exercice 3

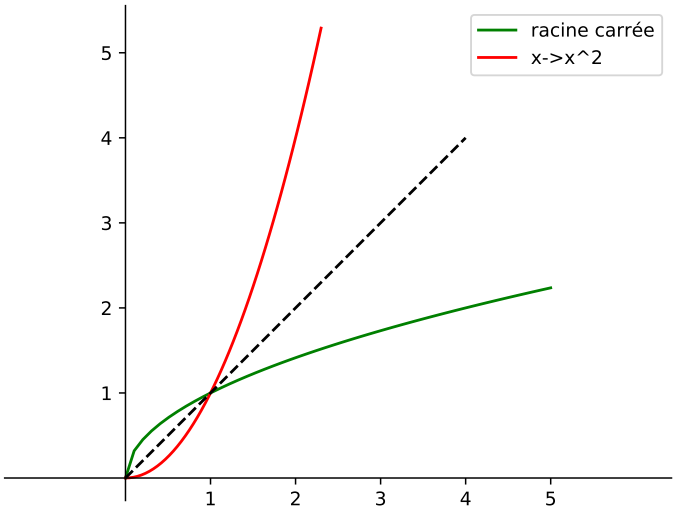
Tracer les représentations graphiques des fonctions exp, ln, racine carrée et de la fonction $x \mapsto x^3 + x - 2$.

Par défaut, python adapte les axes aux valeurs des abscisses fournies pour la première courbe. On peut modifier les limites du tracé par `plt.xlim(val_min,val_max)` et `plt.ylim(val_min,val_max)` ou par `plt.axis([valx_min,valx_max,valy_min,valy_max])`.
On donne un nom aux axes par `plt.xlabel('nom_abscisse')` et `plt.xlabel('nom_ordonnées')`. On peut aussi modifier les graduations placées automatiquement sur l'axe avec `plt.xticks` et `plt.yticks`.
Par défaut, le tracé s'effectue dans un cadre formé de quatre bords `('right','top','bottom','left')`; les bords bas et gauche indiquent respectivement les abscisses et les ordonnées. Cette configuration de base n'est pas adaptée pour le tracé traditionnel des courbes en mathématiques où l'on trace des courbes dans un repère orthonormé avec des axes passant par le point (0,0). Pour cela if faut modifier l'objet "cadre", d'abord en le récupérant dans une variable puis en effaçant les bords inutiles (haut et droite par exemple), en limitant les graduations sur les deux bords qu'on conserve (gauche et bas) et enfin en déplaçant ces bords sur l'abscisse ou l'ordonnée 0. Voici les commandes adaptées :

```
ax = plt.gca()
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.yaxis.set_ticks_position('left')
ax.spines['left'].set_position(('data',0))
ax.spines['bottom'].set_position(('data',0))
```

Exercice 4

Reproduire la figure suivante :



On peut tracer sur une même feuille plusieurs figures différentes (on parle de sous-figures). Il faut pour cela les organiser en lignes et colonnes (par exemple, trois lignes de deux figures). Les différentes figures sont alors numérotées dans l'ordre en commençant par la première ligne. Par exemple, pour six figures organisées en trois lignes et deux colonnes,

$$\begin{array}{cc} F_1 & F_2 \\ F_3 & F_4 \\ F_5 & F_6 \end{array}$$

Pour passer une sous-figure en figure courante, on utilise la commande

```
plt.subplot( nbre_lignes , nbre_colonnes , numéro )
```

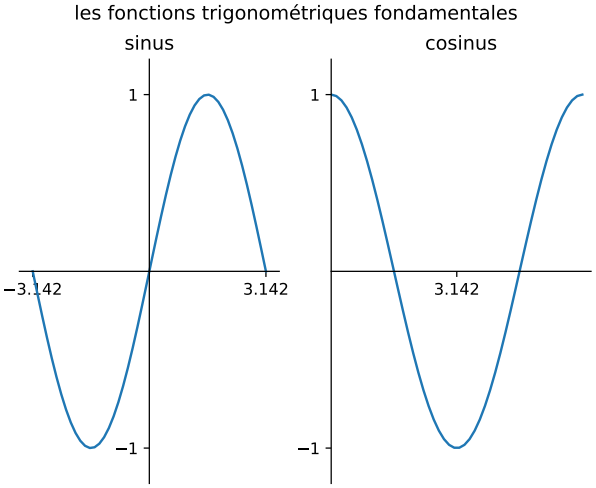
Le tracé se fait alors comme d'habitude avec `plt.plot`. La commande `plt.suptitle(nom)\verb`, où `nom` est une chaîne de caractères, permet de donner un nom global à la page de figure courante tandis que la commande `plt.title(nom)` donne un nom à la figure (ou sous-figure) en cours. Par exemple, le code suivant génère la page de figures qui suit :

```
X = np.linspace(-np.pi, np.pi)
XX = np.linspace(0, 2*np.pi)
plt.suptitle("les fonctions trigonométriques fondamentales")

plt.subplot(1, 2, 1)
ax = plt.gca()
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.yaxis.set_ticks_position('left')
ax.spines['left'].set_position(('data', 0))
ax.spines['bottom'].set_position(('data', 0))
plt.xticks([-np.pi, np.pi])
plt.yticks([-1, 1])
plt.axis([-3.5, 3.5, -1.2, 1.2])
plt.plot(X, np.sin(X))
plt.title("sinus")
```

```
plt.subplot(1, 2, 2)
plt.plot(XX, np.cos(XX))
ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['right'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.yaxis.set_ticks_position('left')
ax.spines['bottom'].set_position(('data', 0))
plt.xticks([-np.pi, np.pi])
plt.yticks([-1, 1])
plt.axis([0, 6.5, -1.2, 1.2])
plt.title("cosinus")

plt.show()
```



Exercice 5

Reproduire la figure suivante :