



Interrogation 5

Dessins

Correction

Exercice 1 :

Donner les syntaxes précises suivantes avec rédaction éventuelle :

1. Nom du module permettant de faire des graphes de fonctions.

Le module permettant de pouvoir faire des graphes de fonctions est le module `matplotlib.pyplot`. Il est généralement renommé `plt`.

2. Définition de la syntaxe `plt.plot(X,Y)`.

La syntaxe `plt.plot(X,Y)` permet de créer un calque de la ligne brisée des points de coordonnées $(X[i], Y[i])$. Pour cela, il faut que les deux listes `X` et `Y` ait la même taille et ne soit composées que de flottants.

3. Expliquer comment modifier le code précédent pour changer le style du tracé en donnant un exemple.

Pour modifier le style d'un tracer, il faut rajouter une option dans l'utilisation de la fonction `plt.plot` après les deux listes de coordonnées. Cette nouvelle option doit être une chaîne de caractères dont chaque caractère est choisie dans une liste prédéfinie et code pour un style particulier. Ces caractères sont regroupés en trois catégories de choix esthétiques : la couleur, le style de points et le style de trait. Par exemple `plt.plot(X,Y,"yo-")` correspond à des points ronds reliés avec un trait continu jaunes.

4. Donner les deux commandes pour donner des noms aux axes sur un graphique (en précisant la position de ces commandes).

Pour donner des noms aux axes d'un graphique, on utilise `plt.xlabel("nom aex des abscisses")` et `plt.ylabel("nom aexe des ordonnées")`.

5. Donner la commande de modification de l'épaisseur du tracé d'une fonction (et l'endroit où la mettre).

L'épaisseur d'un trait d'un graphique est une option facultative dans la fonction `plt.plot` (en plus de la chaîne de caractères pour l'effet de style) et s'utilise de la manière suivante : `lw=nb` ou `linewidth=nb`, où `nb` est un nombre correspondant à l'épaisseur choisie du trait. Par exemple, `plt.plot(X,Y,"yo-",lw=3)` reprend l'exemple précédent avec une épaisseur de trait de 3.

6. Syntaxe d'une fonction `lambda`

Pour créer une fonction `lambda` (en général à l'intérieure d'une fonction), on utilise la syntaxe : `f=lambda x1,...,xn : f(x1,...,x_n)`.

7. Commande et position pour pouvoir mettre une légende sur un graphe.

Pour mettre une légende sur un graphe, on commence par définir des étiquettes pour chaque calque créée en ajoutant l'option facultative `label="nom de calque"` dans la fonction `plt.plot` correspondante (en plus du style et de l'épaisseur éventuelle). Puis on ajoute `plt.legend()` avant l'utilisation du `plt.show()` pour faire apparaître la légende.

Exercice 2 :

Écrire une fonction `composition(f:"function")` -> `None` qui permette de dessiner l'image du cercle trigonométrique par la fonction f , i.e. qui dessine $f \circ \sin$ en fonction de $f \circ \cos$ sur $[0, 2\pi]$ dans un repère orthonormé (avec importations éventuelles).

```

1 import matplotlib.pyplot as plt
2 import math as mt
3
4 def composition(f:"function") -> None :
5     plt.figure()
6     T = [2*mt.pi/1000*k for k in range(0,1001)]      # Listes des abscisses
7     fsin = [f(mt.sin(t)) for t in T]
8     fcos = [f(mt.cos(t)) for t in T]
9     plt.plot(fcos,fsin)
10    plt.xlabel("$f\circ\cos$")    # nom (avec  $\LaTeX$ ) des abscisses
11    plt.ylabel("$f\circ\sin$")    # nom (avec  $\LaTeX$ ) des ordonnées
12    plt.show()

```