Dessiner avec Python : le module matplotlib.pyplot

Petit aide-mémoire (suffisant pour réaliser les exercices proposés)

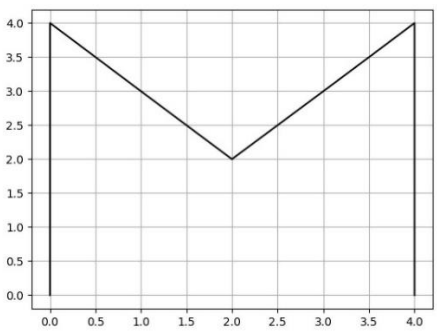
Import du module matplotlib.pyplot	import matplotlib.pyplot as plt
Dessine un point de coordonnées (X ; Y) sous forme de point (.), de couleur rouge (r) et de taille 3 (ms = markersize). Les formes possibles sont . (point), x (croix), o (disque), <, >, v, ^, *, s (carré), p (pentagone), h (hexagone), d (diamant), +… Les couleurs possibles sont b (bleu, couleur par défaut), g (vert), r (rouge), c (cyan), m (mauve), y (jaune), k (noir), w (blanc), ou triplet RGB (p. ex. (0.5, 0.1, 0.3))	plt.plot (X, Y, "r.", ms = 3)
Dessine en rouge (r) une séquence de points (.) de taille 3, reliés entre eux par une ligne continue (-) d'épaisseur 2 (lw = linewidth). Les abscisses des points sont dans listeX, leurs ordonnées dans listeY. Les styles de lignes possibles sont – (ligne continue, style par défaut), : (ligne en pointillés), -- (ligne en tirets), -. (alternée point / tiret).	plt.plot (listeX, listeY, "r.-", ms = 3, lw = 2)
Redéfinit les limites des axes du graphique. Par défaut, ces limites sont celles du graphique dessiné.	plt.axis (xmin = 0, xmax = 5, ymin = -3, ymax = 2) ou : plt.axis (L), avec L = [0, 5, -3, 2]
Pour avoir la même échelle sur les deux axes	plt.axis ("equal")
Définit les graduations (ticks) du graphique : GradX et GradY sont des listes de valeurs numériques.	plt.xticks (GradX) plt.yticks (GradY)
Affichage du graphique construit. Cette instruction suspend l'exécution du script python, jusqu'à la fermeture de la fenêtre graphique.	plt.show()
Enregistrement du graphique au format pdf, sous le nom monGraphique.pdf (dans le dossier courant). Les formats possibles sont png, jpg, tiff, pdf, ps, eps… L'enregistrement du graphique est également possible depuis la fenêtre graphique.	plt.savefig ("monGraphique.pdf", format = "pdf")

Parcours 4. Exercices de base

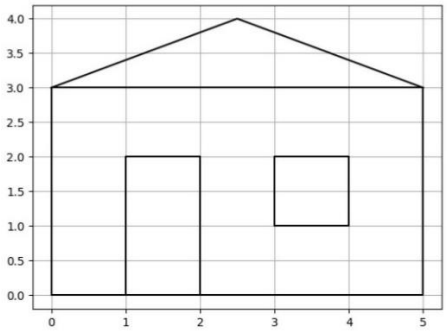
Exercice 34. (Inspiré d'un sujet zéro du CAPES de mathématiques.) La fonction suivante permet d'afficher le dessin d'une lettre « M » :

```
import matplotlib.pyplot as plt

def sujetZeroCAPES():
    X = [0, 0, 2, 4, 4]
    Y = [0, 4, 2, 4, 0]
    plt.plot(X, Y, "k-")
    plt.grid()
    plt.show()
```

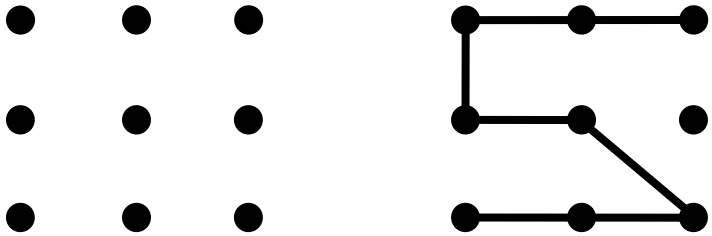


Écrire une fonction **dessineMaison** (sans arguments) qui dessine la « maison » représentée ci-dessous. Modifier cette fonction pour que la « porte » soit bleue et la « fenêtre » rouge.



Exercice 35. Écrire une fonction dessinePolygone qui prend comme argument une liste non vide de listes représentant les coordonnées des sommets d'un polygone, et dessine ce polygone. (Par exemple, la liste [[1, 3], [1, 5], [4, 5], [4, 3]] définit les sommets d'un rectangle.)

Exercice 36. (Schéma de déverrouillage d'un smartphone.) Les « neuf points » ci-dessous (à gauche) permettent de définir un « schéma de déverrouillage » de smartphone. Un « schéma » est alors une suite de points (définie en glissant le doigt) donnant lieu à un tracé tel que celui représenté ci-dessous.



En supposant que les neuf points sont désignés par leurs coordonnées entières (le point inférieur gauche étant de coordonnées (0 ; 0)), le tracé représenté ci-dessous est donné par la suite de points de coordonnées (2 ; 2), (0 ; 2), (0 ; 1), (1 ; 1), (2 ; 0), et (0 ; 0), représentable en Python par la liste de listes [[2, 2], [0, 2], [0, 1], [1, 1], [2, 0], [0,0]].

Écrire une fonction **schemaDeverrouillage** qui prend comme argument une liste de listes représentant un schéma de déverrouillage et produit un dessin tel que celui représenté ci-dessus (à droite).

Exercice 37. Que fait la fonction suivante ?

```
import math
import matplotlib.pyplot as plt

def queFaitElle(n):
    # à déterminer...

    plt.title("Mystère...")
    LX = [ -4 + x / n for x in range(8 * n + 1)]
    LY = [ math.sin(x) for x in LX]
    plt.plot(LX, LY, "r-", linewidth = 2)
    plt.show()
```

Exercice 38. Écrire une fonction **traceCourbeCosinus** qui prend comme arguments un entier n, deux entiers a et b avec a < b, et trace la courbe de la fonction cosinus sur l'intervalle [a ; b] à l'aide de n points. (On pourra s'inspirer de l'exercice précédent…)

Exercice 39. Écrire une fonction **traceCourbeFonction** qui prend comme arguments une fonction, un entier n et deux entiers a et b avec a < b, et trace la courbe de cette fonction sur l'intervalle [a ; b] à l'aide de n points. (On pourra s'inspirer de l'exercice précédent…)

Parcours 5. Pour aller plus loin…

Exercice 40. (Construction graphique des termes d'une suite récurrente.) Écrire une fonction **IterationSuiteRecurrente** qui prend comme arguments une fonction définissant les termes d'une suite récurrente, un nombre réel qui est le premier terme de la suite, le nombre d'itérations, et illustre la construction graphique des termes de cette suite.

Exercice 41. (Encadrement par la méthode des rectangles.) On souhaite déterminer un encadrement de $\int_2^3 \ln(x)dx$ par la méthode des rectangles et représenter graphiquement cet encadrement. Écrire une fonction **methodeRectangles** qui prend comme argument le nombre de rectangles souhaité, trace le graphique correspondant, et renvoie les valeurs encadrantes.

Rappel. Le module **random** (**import random**) permet notamment d'utiliser les fonctions **random.randint(a, b)**, qui renvoie un entier pris aléatoirement entre **a** et **b** (bornes comprises), **random.random()**, qui renvoie un réel compris entre **0** et **1** (bornes comprises), et **random.uniform(a, b)**, qui renvoie un réel pris aléatoirement entre **a** et **b** (bornes comprises).

Exercice 42. (Approximation d'une intégrale par la méthode de Monte-Carlo.) Notre objectif est de calculer une valeur approchée de $\int_0^2 e^{(-x^2)}dx$ et de visualiser la simulation correspondante. Une étude rapide de la fonction permet de montrer que la courbe est contenue dans le rectangle R défini par $0 \leq x \leq 2$ et $0 < y \leq 1$. La probabilité p qu'un point pris au hasard dans le rectangle R soit situé sous la courbe de la fonction est p = (aire sous la courbe) / (aire de R).
Écrire une fonction **montecarlo** qui prend comme argument le nombre de points que l'on souhaite simuler, affiche une visualisation graphique de la simulation, puis renvoie l'approximation de l'aire sous la courbe.

Exercice 43. (Alice et le Charpentier.) Alice et le Charpentier se sont donnés rendez-vous chez le Lapin pour prendre un thé entre 17h et 18h. Chacun d'eux a promis d'attendre l'autre un quart d'heure, pas plus, et en aucun cas d'attendre au-delà de 18h. On suppose que les heures d'arrivée d'Alice et du Charpentier suivent des lois uniformes sur l'intervalle [17;18]. On veut trouver une estimation de la probabilité qu'Alice et le Charpentier se retrouvent.

- Écrire une fonction **RendezVous** qui prend en argument un nombre N, simule aléatoirement N couples d'horaires d'arrivée et renvoie la fréquence des rendez-vous réussis.
- Rajouter à la fonction précédente la visualisation graphique de cette simulation : les rendez-vous manqués seront représentés pas un point noir et les rendez-vous réussis par un point rouge (coordonnées = (Alice ; Charpentier)).

Exercice 44. (Aiguille de Buffon.) Écrire une fonction **aiguilleDeBuffon** qui prend comme argument un entier N, simule le lancer de N aiguilles de longueur 1 sur un plancher dont les lames parallèles sont de largeur 1, affiche le résultat (dessin) obtenu, et renvoie la fréquence de l'événement « l'aiguille est à cheval sur deux lames ».
Pour chaque lancer, on tirera aléatoirement les coordonnées d'une extrémité de l'aiguille dans un rectangle de largeur 1 et de hauteur 3 (par exemple), ainsi qu'une orientation de l'aiguille (angle).

Exercice 45. Que fait la fonction suivante ?

```
import matplotlib.pyplot as plt
import random

def queFaitElleBIS():
    # à déterminer

    for i in range(100):
        n = 0
        for j in range(400):
            if random.random() < 0.7:
                n = n + 1
        fréquence = n / 400
        if (fréquence > 0.65) and (fréquence < 0.75):
            plt.plot(i, fréquence, "rx")
    plt.axis(xmin = 0, xmax = 100, ymin = 0.5, ymax = 0.9)
    plt.show()
```

Exercice 46. En 2017, 47,5 % des élèves qui ont passé le baccalauréat en France ont eu une mention. En vous inspirant de l'exercice précédent, écrire une fonction **mentionsBAC** permettant d'illustrer le fait que dans 95 % des échantillons, le pourcentage des élèves français ayant leur bac avec une mention reste compris entre deux valeurs proches de 47,5 %.