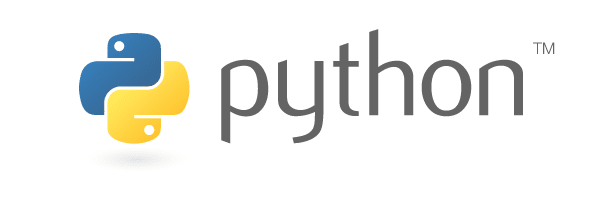


**Rapport de Projet**



**Simulation Python Matplotlib**

Intitulé du sujet :

* Construire une map avec les obstacles en utilisant matplotlib
* Robot doit traverser la map en diagonal en évitant les obstacles

Dans un premier temps, vu que nous allons utiliser une bibliothèque du nom de matplotlib, je fais des recherches et imprime la fiche des fonctions de ce module.

Je fais ensuite de nombreux exercices afin de comprendre à maitriser cet outil.

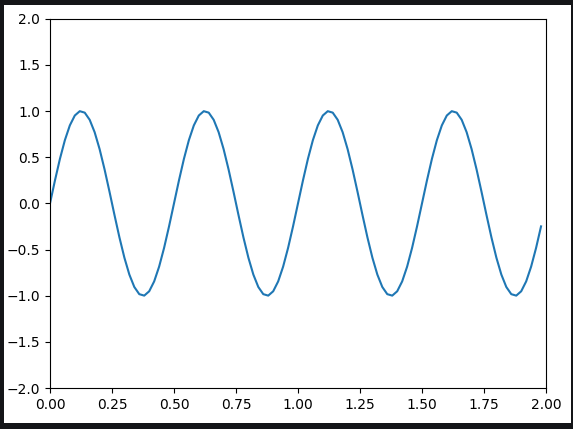
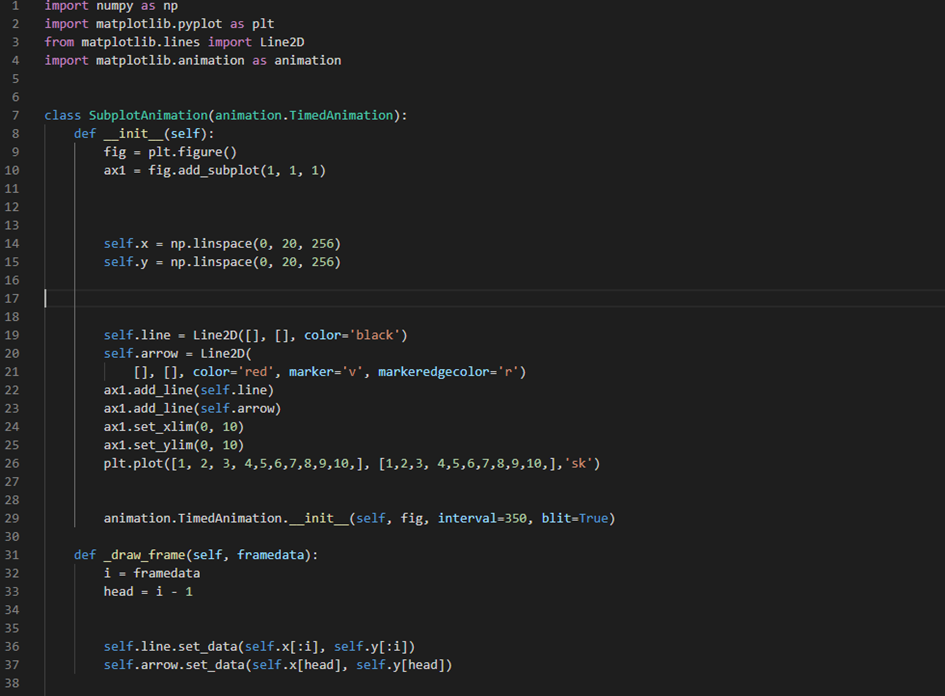


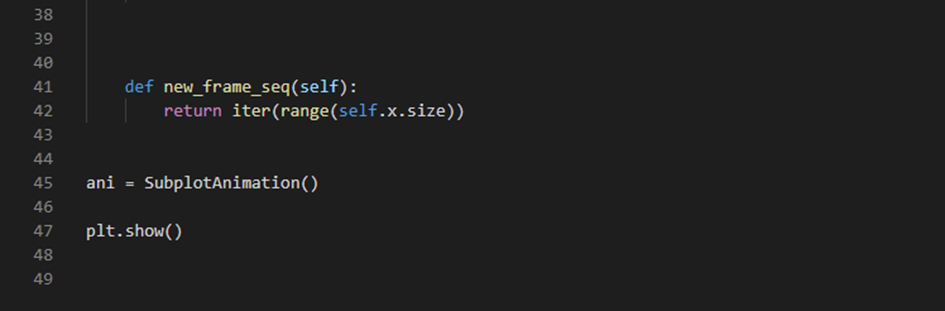
Fig 1 : Courbe sinusoïdale sur maplotlib

Une fois les fonctions de base acquise je me permet de me pencher plus sérieusement sur le projet et son intitulé.

Explication des consignes…

Capture du programme sur Python :





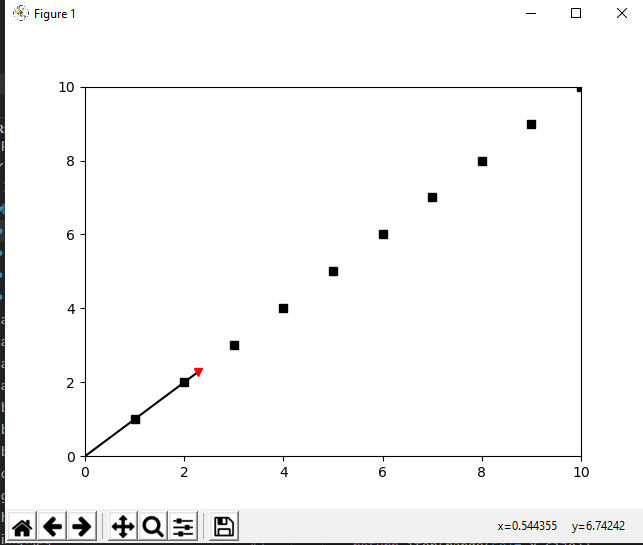
Fonctionnement et explications :

* On créer une classe qui regroupe les différentes fonctions que l’on va utiliser
* 1ère fonction : l’initialisation

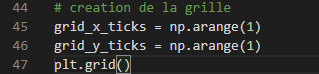
-On créer la figure

-On choisit l’échelle (ici on a mis 1)

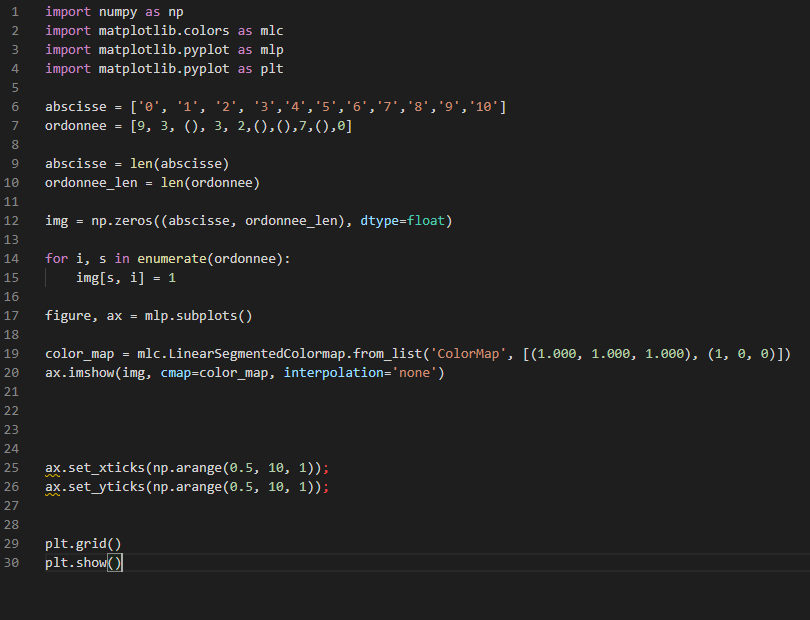
* On trace une courbe en x, y d’origine 0, vitesse 20 et précision 256.
* On modifie l’aspect de la courbe (couleur, forme…)
* On créer une forme que l’on va ajouter au bout de la courbe (ici une flèche pour simuler la positon à l’instant t du robot).
* On ajoute la courbe et la flèche sur la figure créer précédemment
* On définit les axes (ici les graduations représentent la taille de la map que l’on souhaite)
* On ajoute des points sur le graphique (qui vont simuler les objets que le robot devra éviter)
* Fonction animation qui lance le mouvement du robot
* Il faut que la flèche s’efface au fur et à mesure du mouvement car ce sont en faites une multitude de flèche qui s’ajoute au cours du temps. Pour cela on créer une nouvelle fonction avec une variablement i qui va s’incrémenter et grâce à cette variable que l’on soustrait on peut effacer les flèches du positionnement d’avant.

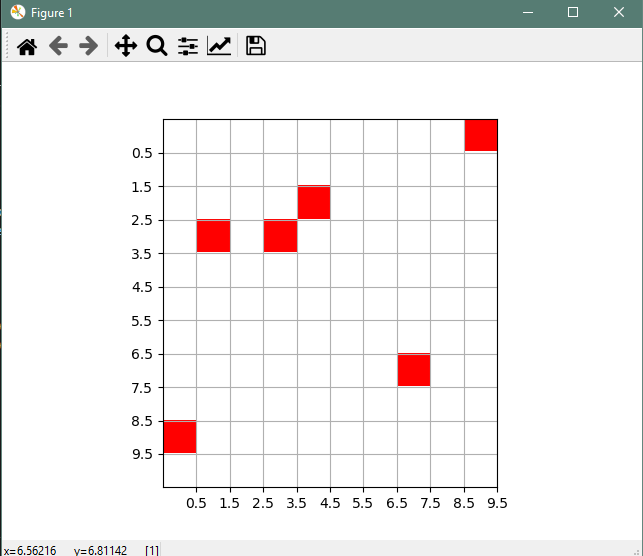


Prochain objectifs :

Amélioration des obstacles et création d’une grille sous python.

Création d’une grille :



Résultats sous Vscode :

Grâce à la grille on va pouvoir « colorier » des cases qui représenteront des obstacles et que le robot devra éviter par la suite. On souhaite au final une grille de 100x100, au niveau des dimensions.

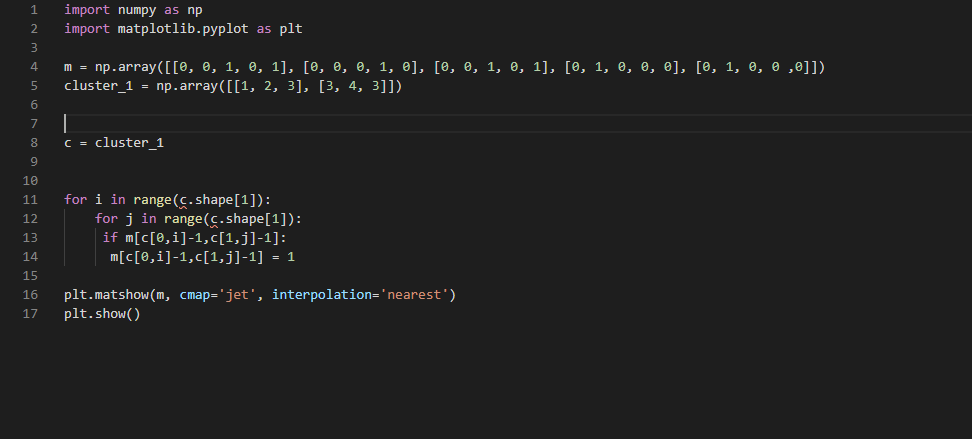
Dans le programme nous allons grâce à des tableaux remplir les coordonnées de points où seront les obstacles. Bien sûr il faut faire attention à la graduation des axes et qu’il soit logique avec les tableaux (ex : un obstacle à une coordonnée 100 :100 sur un graphique de 10x10).

Une fois cela réaliser, on va grâce à la fonction len (longueur) qui va nous permettre d’obtenir la longueur de la chaine du tableau et par la suite l’insérer sur notre figure (graphique) grâce à la fonction mlp.suplots().

Il nous reste plus que la mise en forme des obstacles à réaliser grâce à la fonction « colormap » qui grâce à des valeurs définies permet d’obtenir précisément une couleur (rouge ici).

Une fois ce programme réalisé, je me suis rendu compte que ce genre de programme aller me poser problème pour la suite avec la détection d’obstacle.

J’ai donc recherché une autre façon plus astucieuse de réaliser les obstacles. Le plus évident est de remplir une grille de 0 et de 1 (une matrice) et que chaque nombre représente un type d’élément (0= vide et 1= objets). On créer une matrice « a » avec la fonction « array » 



Maintenant que nous avons créés obstacle, le prochain objectif est de les générer aléatoirement et non déjà dans une matrice. Pour cela on créer une matrice aléatoire de 100x100.



Donc pour l’instant nous avons deux programmes séparés (une trajectoire robot+grille et une map d’obstacle), le but est de les assemblés pour qu’il fonctionne ensemble.

Pour faire ça il faut déjà revoir la structure des deux programmes, en effet le programme « trajectoire » est réalisé avec des classes alors que pas le deuxième cela peut créer des erreurs par la suite.

Il faut refaire les deux programmes de façon à que :

-le 1er contient la map obstacle, grille…

-le 2eme contient le robot et son déplacement.

-le 3 eme un programme final qui importerait les deux d’avant (on utilise la fonction « import »)

Autres test…