Projet 3 : The hero of MacGyver !!!

Introduction du projet :

Dans le parcours développeur Python sur OpenClassroom, il nous est demandé dans le Projet 3 de développer un jeu de labyrinthe sur interface mettant en scène MacGyver perdu dans de labyrinthe. Il doit s'en échapper et pour ce faire MacGyver doit récupérer trois objets disposés de manière aléatoire dans le jeu pour fabriquer une seringue en vue d'endormir le garde et de pouvoir s'évader.

L'algorithme du labyrinthe :

a . Pour la création du code de mon labyrinthe, j'ai commencé par coder un labyrinthe en mode console. La question qui vient est pourquoi avoir choisir cette façon de coder ? La réponse est que : Le labyrinthe en mode console va me permettre en premier lieu de faire toutes les classes nécessaires au fonctionnement du labyrinthe (je détaillerais les difficultés que j'ai rencontrées plus en bas)

b . J'ai répartie le fonctionnement de mon labyrinthe en six fichiers :
- labyrinthe.py = Ce fichier contient toutes les fonctions principale de la class maze.
- MacGyver.py = La class Macgyver contient ses coordonnées(x,y).
- Guardien.py = Cette class est celle du guardien.
- Objet.py = Cette class est celle des objets.
- Constantes.py = Ce fichier contient toutes les images et la taille de la fenêtre au fonctionnement de l'interface du jeu.
- Interface_laby.py = Ce fichier gère l'affichage et les déplacements du labyrinthe en mode pygame.

Les différentes problématiques rencontrées dans ce projet de façon détaillées et structurées :

Les différentes fonctions de ma class maze (labyrinthe.py) :

a . def parse_file = Cette fonction que j'ai crée est une grille fictive avec les coordonnées x et y. Cette fonction me permet de gérer les différents coordonnées de Macgyver, du guardien et des objets.

B . def get_keys = Elle me permet de récupérer les coordonnées de tous les chemins contenus dans le dictionnaire.

c . def random_coordinates = La fonction random_coordinates sert à choisir aléatoirements 3 coordonnées de chemins qui nous permettrons plus-tard de déplacer les objets aléatoirement dans le labyrinthe.

d . def get_items = Cette fonction récupère la fonction random_coordinates, pour renomer les 3 coordonnées des chemins en objet, c'est-à-dire : que les clés des trois coordonnées sont nommer avec le caractère c, mais dans la fonction get_items ses 3 coordonnées s'appelleront t, s et e.

e . def bottom, def top, def left, def right = Ses différentes fonctions me permettent de gérer les déplacements de Macgyver.

f . def check_move = Cette fonction est comme un feu tricolore, elle nous indique si ont peut avancer, s'arreter, ramasser les objets et faire attention aux collisions.

g . def parse_laby = Grâce à la fonction d'origine qui est self.labyrinthe qui à été initialisé dans la méthode def __init__ , la fonction parse_laby dessine un labyrinthe en mode console.

h . def picture_maze = Elle dessine également le labyrinthe en mode pygame, graçe aux coordonées de la variable taille_sprite = 30, taille_sprite est la taille des caractères de mon labyrinthe. Ce qui nous permettra de voir la taille des caractères comme nous le souhaitons.

i . def move = Dans cette fonction, il y a quatre direction de déplacements, qui nous servent à déplacer Macgyver, si on regarde bien la structure de la fonction, on a fait un réfactorielle des directions, ce qui nous permettra en premier lieu de faire une vérification de chaque direction, pour vérifier si on est autoriser à avancer ou non, ensuite ont remplace les coordonnées de Macgyver à un chemin, en fonction des directions ont incrémente pour pouvoir avancer ou ont décrémente ses coordonnées. Et pour finir ont redéfinir Macgyver à son caractère initiale (d).

Le script interface_laby :

a . Bien évidemment, avant de vouloir coder je dois importer les différents fichier et module nécessaire au fonctionnement du labyrinthe.

b . J'ouvre une fenêtre avec comme variable nombre_de_sprite = 15 et la taille_sprite = 30, ceci me permet d'avoir une taille adéquate pour avoir une interface conforme au labyrinthe.

c . Je créer un titre pour le labyrinthe

d . Je rafraichis la fenêtre

e . Je déclare une variable de boucle infinis pour le jeu

f . Dans ma boucle j'intègre différentes fonctions que vous verrez directement dans mon code avec les commentaires.

Le fichier constantes_2.py :

a . Dans ce fichier pour commencer, j'ai tout simplement calculer le nombre_de_sprite * taille_sprite, pour ensuite mettre ses calculs dans une variable que je vais nommer cote_fenetre.

b . Pour finir J'ai donner à chaque image un nom_de_variable qui lui est propre, ce qui me permettra de les mettre directement dans ma fonction picture_maze.

Conclusion de ses 6 mois de travaille pour le projet 3 :

Pour être honnête, j'ai du m'exercer pendant 3 mois avant de me lancer dans ce projet. Certes après m'être exercer je me suis lancer dans le projet 3, ce qui m'a pris 6 mois car j'avais besoin de comprendre et de me familiariser avec le language python.