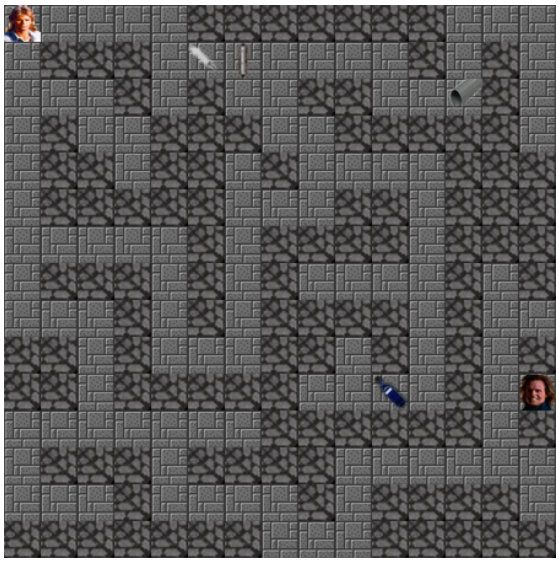
PARCOURS OPENCLASSROOMS
DEVELOPPEUR D'APPLICATIONS PYTHON

Projet 3 : Aidez MacGyver à s'échapper



https://github.com/AxlMi/OC_Project3

Réalisé durant le mois de mars

MICHEL Axel

Introduction :

Durant ma formation Openclassroom , il m'a été demandé de réaliser un labyrinthe doté de plusieurs fonctionnalités et de contraintes. Pour se faire j'ai commencé en respectant l'ordre de fabrication du labyrinthe donné sur le projet 3. J'ai commencé par créer le cadre de départ, d'animer le personnage, de récupérer les objets et ensuite de changer la fin du jeu en fonction des contraintes et conditions indiquées.

Apprentissage :

Pour réaliser ce projet je me suis aidé des cours de départ écrit par Céline Martinet Sanchez , du cours de Python écrit par Vincent Le Goff , du livre Apprendre à programmer avec Python 3 de Gerard Swinnen mais aussi principalement du cours « interface graphique Pygame pour Python » sur lequel je me suis inspiré pour le lancement du projet, sans oublier bien sur les explications et conseils de mon mentor.

Architecture du projet :

Module main : Elle commence par initialiser le jeu avec l'affichage des différentes informations de départ. Elle se compose aussi de la fonction launch_game(), qui permet de lancer le jeu. Elle est composée de deux boucles avec une condition de booléen permettant l'affichage du menu et du labyrinthe, ainsi que diverses conditions permettant le déplacement.
Modules constantes : Ce module se compose des différentes constantes du jeu.
Module classes : Dans ce module nous retrouvons principalement deux classes, la première permettant la construction, l'affichage du labyrinthe ainsi que des objets aléatoirement placés via des méthodes, la seconde classe concerne le personnage et ses méthodes pour se déplacer et de prendre des objets.

Algorithme :

Ramasser les items :

Pour commencer j'avais tout simplement indiqué un compteur dans la classe du personnage, indiquant qu'à chaque fois qu'il passerait sur une case d'objet « case i », le compteur s'incrémenterait de 1 et je modifiais la case en « o » pour ne pas ajouter un objet à chaque fois qu'on passerait sur la case. Pour gagner la partie, mon compteur devait être égal au nombre d'objet placés.

Je trouvais cette méthode fonctionnelle, cependant je ne pouvais pas savoir quel objet était ramassé car i ne correspondait pas à l'objet mais a une case contenant un objet.

J'ai donc modifié ce dernier avec une condition indiquant que si je me trouvais sur une case « i » j'effectuais :

- une boucle for dans le dictionnaire « pos_obj » et de comparer l'objet (qui a pour valeur un tuple avec les coordonnées (puisqu'elle celle-ci ne bougerait plus une fois l'objet placé) avec les coordonnées du personnage , si cela correspond, cela veut dire qu'on se situe sur la case de cet « objet » .
- Je décide donc d'afficher un message indiquant que nous avons bien ramassé cet « objet ».
- Cette boucle ne s'effectue que lorsque le personnage se trouve sur une lettre « i » pour n'effectuait qu'une fois la démarche, une fois le personnage passé , cette lettre redevient un « o ».
- J'ajoute ensuite l'objet dans l'inventaire du personnage.
- Pour gagner, le joueur devra avoir le même nombre d'objet dans son inventaire que dans la liste d'objet placé aléatoirement.

Difficultés :

Trouver une manière d'afficher le nom de l'objet ramassé et de pas réitérer cette méthode a chaque fois que l'on passe sur cette case.

Position aléatoire des items :

Pour afficher les objets aléatoirement dans le labyrinthe j'ai effectué une boucle for avec les objets à placer dans le labyrinthe.

- Ensuite dans une boucle while, j'ai créé deux variable « random_x » et « random_y », chacune d'elle affichera un nombre aléatoire compris entre 0 et le nombre de sprite (-1 car on commence à 0).
- J'affiche une condition permettant d'afficher la map du labyrinthe avec les coordonnes x et y (données précédemment aléatoirement), si elles correspondent a la lettre « O » qui correspond au carrelage, dans ce cas on effectue la condition.
- La condition enregistre la position de l'objet dans un dictionnaire avec un tuple concernant x et y. Ensuite elle multiplie x et y par la taille du sprite et on crée l'affichage de l'objet en fonction du positionnement aléatoire.
- Pour terminer la condition je change la lettre « o » par la lettre « i » qui permettra plus tard d'effectuer une condition si on passe sur « i » et dans cette condition je remettrai la lettre « i » en « o » pour éviter qu'elle ne se répète s'il y a plusieurs passages sur la case. Je termine l'instruction de la boucle while.

Difficultés :

Les difficultés rencontrées étaient de trouver une manière d'afficher le nom des objets positionnés aléatoirement dans le labyrinthe, j'ai donc décidé de les enregistrer dans une constante nommé « pos_obj » avec un tuple pour les positions x et y qui ne seront pas modifié par la suite.

Conclusion :

Pour conclure, étant novice dans le monde du développement, je suis satisfait de ce que j'ai pu produire, même si je suis persuadé qu'il existe un moyen plus rapide et plus simple d'accéder à ce résultat. Ce projet m'a permis de découvrir et de mettre en pratique le fonctionnement de Pygame, la structuration d'un projet, l'utilisation de GitHub et d'utiliser un environnement virtuel.