"Aidez MacGyver à s'échapper !"
OpenClassrooms parcours DA Python - projet 3

Projet sur GitHub : https://github.com/beSyl/Projet3

Le programme
Le fichier 'game.py' correspond au programme dont les modules sont 'contants.py', 'labyrinth.py' (classe), 'hero.py' (classe) et "my_pygame.py" (classe).
Le fichier 'labyrinth.txt' représente la structure du jeu à l'aide de lignes de caractères.
Le répertoire 'images' contient les images utiles au jeu (fond, murs, personnages, etc.).
Une fichier README.txt détaille les règles du jeu.
Un environnement virtuel est créé spécifiquement pour répondre aux besoins du jeu (utilisation de Pygame).

Le présent document correspond au livrable « *Document texte expliquant votre démarche et comprenant le lien vers votre code source (sur Github)* », attendu dans le cadre du projet 3 OpenClassrooms.

Les règles du jeu
Le personnage MacGyver doit s'échapper d'un labyrinthe. Pour cela il doit retrouver le gardien afin de l'endormir avec une seringue : celle-ci doit être fabriquée par MacGyver à l'aide de 3 objets qu'il doit ramasser dans le labyrinthe (une aiguille, un tube et de l'éther).
Si MacGyver rejoint le gardien après avoir trouvé les 3 objets, alors il gagne le jeu. Sinon, il perd.

Les 3 objets sont placés dans le labyrinthe de manière aléatoire.

La mise en œuvre
Le fichier 'game.py' contient une fonction 'main()' qui regroupe le déroulement du jeu.
On y distingue 2 moments :

- on crée les instances : éléments de jeu Pygame, le labyrinthe (murs, objets aléatoires, etc.) et le personnage
- puis, on lance une boucle de jeu qui gère les événements du jeu (commandes de déplacement du personnage, ramassage d'objets, affichage à l'écran, etc.) jusqu'à ce que le personnage parvienne au gardien

À partir des classes 'MyPygame', 'Labyrinth' et 'Hero', on instancie :

- my_pygame = MyPygame() → éléments liés à Pygame (fenêtre de jeu, commandes au clavier, etc.)
- labyrinth = Labyrinth() → labyrinthe de jeu (structure, emplacement des objets, etc.)
- mac_gyver = Hero() → personnage du jeu (position, mouvements, etc.)

Le labyrinthe est vu comme un plateau de jeu de 15 lignes (numérotées de 0 à 14) et 15 colonnes (numérotées de 0 à 14). Pour le matérialiser, on rédige un fichier 'labyrinth.txt' de 15 lignes composées de 15 caractères chacune. Ces caractères symbolisent :

- une case 'chemin' via laquelle MacGyver peut se déplacer s'il s'agit d'un caractère espace « »
- une case 'mur' empêchant MacGyver de se déplacer s'il s'agit d'un caractère « X »
- le début du jeu, donc la position initiale de MacGyver, s'il s'agit d'un « s »
- la fin du jeu, donc le gardien, s'il s'agit d'un caractère « e »

La structure du labyrinthe est stockée dans la variable 'labyrinth.structure' comme suit :

- chaque ligne du fichier 'labyrinth.txt' est stockée dans une liste, composée de la liste des caractères qui la composent
- ces 15 « listes de ligne » sont insérées dans une liste globale qui correspond à la structure du labyrinthe

Une position se détermine selon les axes x et y (respectivement, abscisse et ordonnée). L'index du caractère dans la « liste de ligne » représente l'abscisse, l'index de la « liste de ligne » dans la liste globale de structure représente l'ordonnée.
Le personnage débute en haut à gauche du labyrinthe (x = 0, y = 0).
La fin du labyrinthe se situe en bas à droite du labyrinthe (x = 14, y = 14). Elle est matérialisée par la lettre « e » dans le fichier 'labyrinth.txt'.

On détermine 3 positions aléatoires à l'aide de la méthode 'random.randint()' en s'assurant qu'il s'agit de cases 'chemin' que MacGyver peut parcourir. Les objets à ramasser sont ainsi insérés dans la variable 'labyrinth.structure', et matérialisés par les lettres :
- « E » (ether)
- « N » (needle, l'aiguille)
- « T » (tube)

Ils ne peuvent être positionnés que sur des cases 'chemin', donc par conséquent ni sur un mur, ni sur le gardien, ni sur la case de départ de MacGyver.

Un objet est ramassé lorsque la position du personnage est identique à celle de l'objet. La lettre «E », « N » ou « T » est alors modifiée en un caractère espace « » et on incrémente le compteur d'objets ramassés : mac_gyver.cart . Ceci permet de vérifier de manière simple si tous les objets ont été ramassés en fin de jeu.

Les déplacements sont gérés en ajoutant / retranchant 1 à la position de MacGyver :
- en abscisse (x) s'il se déplace de manière horizontale
- en ordonnée (y) s'il se déplace de manière verticale

On vérifie qu'un déplacement est possible à l'aide de la méthode 'check_square()' de classe Hero pour s'assurer qu'il reste dans les limites du labyrinthe et qu'il ne conduit pas dans un mur.

Une fois le personnage arrivé à l'emplacement du gardien, on s'assure qu'il a ramassé 3 objets dans son panier à l'aide de la méthode 'check_victory()' de classe Hero, puis on affiche le résultat dans la fenêtre Pygame selon qu'il a gagné ou perdu.

Les difficultés rencontrées
Les difficultés suivantes sont les principales. Elles ont été traitées lors de sessions de mentorat.

Trouver une cohérence entre la structuration du labyrinthe et son affichage : le choix de listes dans une liste globale s'est révélé le plus adapté afin qu'abscisse et ordonnée, utiles à l'affichage, se distinguent facilement.

Simplifier les conditions permettant de se déplacer (1)dans les limites du plateau de jeu et (2)sans qu'il s'agisse d'un mur : les méthodes 'check_coord()' et 'check_square()' (classe Hero) ont permis de simplifier cela et de rendre le code plus lisible.

Utiliser un environnement virtuel : la démonstration par le mentor a permis de débloquer la difficulté de compréhension quant à l'utilisation de virtualenv.

Déployer sur GitHub via Git : la lecture a plusieurs reprises du tutoriel OpenClassrooms a été bénéfique, de même que les conseils du mentor notamment pour ignorer des éléments (ex : créer un fichier '.gitignore').