Labyrinthe pygame.

Avant toute chose, afin d'écrire ce programme, il m'a fallu visionner et assimiler les vidéos suggérées par Openclassroom. La principale, a mon sens, est celle consacrée à la programmation orientée objet. Elle me permis de regrouper au même endroit les fonctions et les structures ou le comportement et les données. Un choix d'organisation qui me permis de créer des objets que je peux par la suite manipuler.
La première étape fut de créer le labyrinthe à partir d'un fichier .txt. et non comme je le pensais directement dans le programme. Ceci me permettra plus tard de modifier la forme de mon labyrinthe si je le souhaite. On commence donc par charger le labyrinthe à partir d'un fichier txt.
La composition du plateau de jeu est une liste de listes. la méthode .append permet de les ajouter. la boucle for permet d'itérer les listes.
Le plateau est composé d'abscisses et d'ordonnées, des X et Y qui servent de coordonnées, utiles pour nous déplacer.
La deuxième étape concerne la création de class : Dans notre cas, il y en aura deux.
La première class concerne le personnage (character) et elle est caractérisee par son nom et sa position.
La méthode constructeur est employée pour trouver l'instanciation des attributs nom, position..
Autre class à créer : le labyrinthe. Il est ici caractérisé par ses murs et ses vides, sa taille, son entrée, sa sortie et une personne dedans.
La encore, la méthode __ init__, l'appel à cet objet éxecute cette méthode sur la nouvelle instance pour effectuer les initialisations nécessaires spécifiques à cette instance.
Le but étant de créer les attributs de l'instance qui vient d'être créé.
Initier le labyrinthe et ses objets placés de façon aléatoire grâce à la fonction random(). random() qui renvoie un nombre aléatoire et choice qui choisit au hasard un nombre dans une liste donnée. la fonction join() est employée car elle accepte n'importe quel itérable.
Les passages libres sont représentés par un "0" , les murs par un "1" et les objets par un "o".
Une difficulté pour moi fut de comprendre le mouvement du personnage sur le plateau de jeu et de me représenter sa progression. De formuler enfin, qu'au fur et à mesure du parcours du personnage, les cases reprennent leur forme initiale. Par exemple, Le personnage passant sur une case vide, au moment ou le personnage(P) est dessus, la case prend sa valeur puis lorsqu'il avance, la case reprend sa valeur initiale (0). Le personnage acquiert une nouvelle position.

Qautre directions sont alors possibles haut, bas, gauche et droite..
Autre difficulté : le ramassage des objets. Ici, a chaque mouvement du personnage, on fait appel à l'attribut character.bag. On fait appel à l'attribut de l'objet.

La fonction play nécessite l'emploi de la boucle While "tant que": Tant que la position du personnage est différente de la case sortie et qu'il n'a pas ramassé les objets, il parcours le labyrinthe.

La fonction "affiche" affiche le labyrinthe puis on calcule la position en pixel des cases.
Enfin, on implémente la fonction "bouge" pour faire avancer le personnage en fonction de la direction donnée puis on vérifie si le personnage est sur une case qui contient un objet. On le récupère alors ou non.
Concernant le jeu et son afichage quelques constantes sont utiles à définir.
on instancie la fenêtre d'affichage, on la nomme et on rajoute une icône puis on instancie le labyrinthe et on lui ajoute un personnage.

On instancie du texte qu'on va afficher par la suite quand l'utilisateur aura trouvé la sortie

Limitation de vitesse de la boucle s’effectue avec 30 frames par secondes pour ne pas surcharger le processeur

Il est essentiel que la fenêtre se ferme quand on clique sur la croix dans le coin.
On implémente les mouvements du caractère en fonction des instructions de l'utilisateur

si le personnage est sur une case gardien on met perdu à True et donc on ne rentrera plus dans la boucle et le personnage est donc bloqué.

On implémente une condition permettant de restart le jeu quand l'utilisateur clique sur Entrée
On affiche le labyrinthe dans la fenêtre
Ici une condition qui permet d'afficher du texte dans la fenêtre quand la partie est gagnée