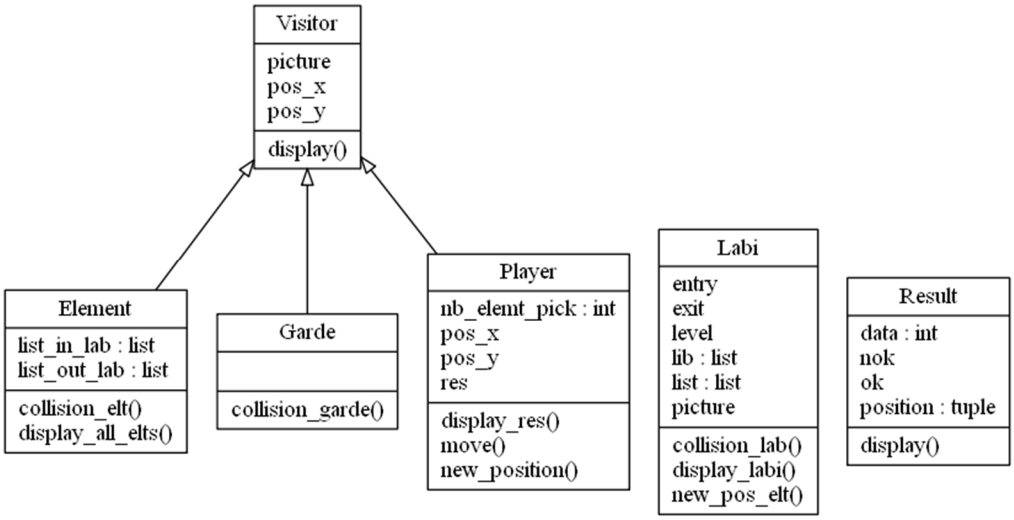
PARCOURS DEVELOPPEMENT D'APPLICATION PYTHON/P3 : Aidez MacGyver à s'échapper

ETUDIANT : Isabelle JOB

1. ORGANISATION DU CODE

1.1 Les Classes

L'ensemble du code est essentiellement organisé autour des classes suivantes :



La classe Labi, qui sert pour créer l'objet labyrinthe.
Ses attributs :

- entry pour indiquer l'entrée du labyrinthe
- exit pour indiquer la sortie du la labyrinthe
- level est la grille qui indique l'emplacement des briques du labyrinthe
- lib est la liste des positions libres du labyrinthe, celles que peut prendre MacGyver en mouvement
- list est la liste des positions occupées par les murs du labyrinthe
- picture est l'image d'une brique du labyrinthe

Ses méthodes :

- collision_lab est une fonction qui prend en entrée une position et indique en retour si cette position est occupée par une brique du labyrinthe ou pas
- display_labi est la fonction qui affiche toutes les briques du labyrinthe
- new_pos_elt qui fournit aléatoirement une position de la liste lib décrite plus haut

La classe result, qui sert à représenter l'objet résultat :
Ses attributs :

- data qui est le résultat : prend la valeur 2 quand la partie n'est pas terminée, puis à l'issue de la partie, 1 pour un résultat gagnant, 0 pour un résultat perdant
- Nok et ok sont des images représentant respectivement la défaite et la victoire

Sa méthode display affiche en fin de partie l'image (Nok ou ok) en fonction du résultat

La classe Visitor, pour représenter les occupants du labyrinthe

Ses attributs :

- pos_x et pos_y permet de définir la position de l'occupant dans le labyrinthe
- picture est l'image de l'occupant

Sa méthode display permet l'affichage de l'image de l'occupant.

Les occupants du labyrinthe sont MacGyver, le gardien et les objets à ramasser par MacGyver. Les classes suivantes héritent donc de la classe Visitor et vont servir à représenter les occupants du labyrinthe.

La classe Garde permet de représenter le gardien, sa méthode collision_garde prend une position en entrée et indique en sortie si cette position correspond à celle du gardien ou pas.

La classe Element pour représenter grâce aux attributs et méthodes de classes, l'ensemble des objets à ramasser par MacGyver.

Les attributs : list_out_lab represente la liste des éléments déjà ramassés, se trouvant donc hors du labyrinthe. list_in_lab, pour les éléments non ramassés par MacGyver et nb_elts pour compter le nombre d'objets à ramasser crées et les limiter.

Ses méthodes : display_all_elts affiche les images de tous les objets ramassés ou pas, collision_elt prend une position en entrée, indique en retour si c'est une position occupée par un objet à ramasser, si c'est le cas, met à jour list_in_lab et list_out_lab.

La classe Player permet de représenter l'objet MacGyver.
Ses attributs en plus des attributs de Visitor:

- nb_elemt_pick utilisé pour compter le nombre d'objets ramassés par MacGyver
- res de la classe Result pour représenter le résultat de MacGyver
- pos_x et pos_y sont initialisés à l'entrée du labyrinthe

Ses méthodes :

- new_position : calcule la position que l'utilisateur voudrait faire prendre à MacGyver à partir des évènements clavier
- move : qui gère la position du labyrinthe et les collisions avec les murs du labyrinthe et les autres occupants, elle fait appel à la méthode new_position et aux méthodes collision_garde, collision_lab et collision_elt des classes Garde, Labi et Element définies plus haut.
- display_res permet d'afficher le résultat de MacGyver en fin de partie

1.2 La mécanique du code

La mécanique du code est initiée par une boucle while qui récupère les évènements claviers tant que le résultat de MacGyver est indéfini et envoie l'information à la fonction routine.

La fonction routine :

- Appelle la méthode move de MacGyver, ce qui a pour effet de déplacer MacGyver, gérer les collisions avec les occupants du labyrinthe et mettre à jour le résultat de partie
- Se charge d'appeler toutes les fonctions d'affichage

2. DIFFICULTES RENCONTREES

La principale difficulté rencontrée sont les ressources, le cours qui n'est pas toujours à jour. Je n'ai quasiment jamais pu faire les installations et utiliser les commandes tel que décrit dans le cours. Avec de la patience, j'ai trouvé toutes les informations sur internet.

3. AXES D'AMELIORATION

L'ensemble du code ainsi qu'une version standalone du jeu existe sur Github :
https://github.com/IsaJcode/PDA_OC_P3
Les résultats de Pylint indiquent que des améliorations du code sont possibles.