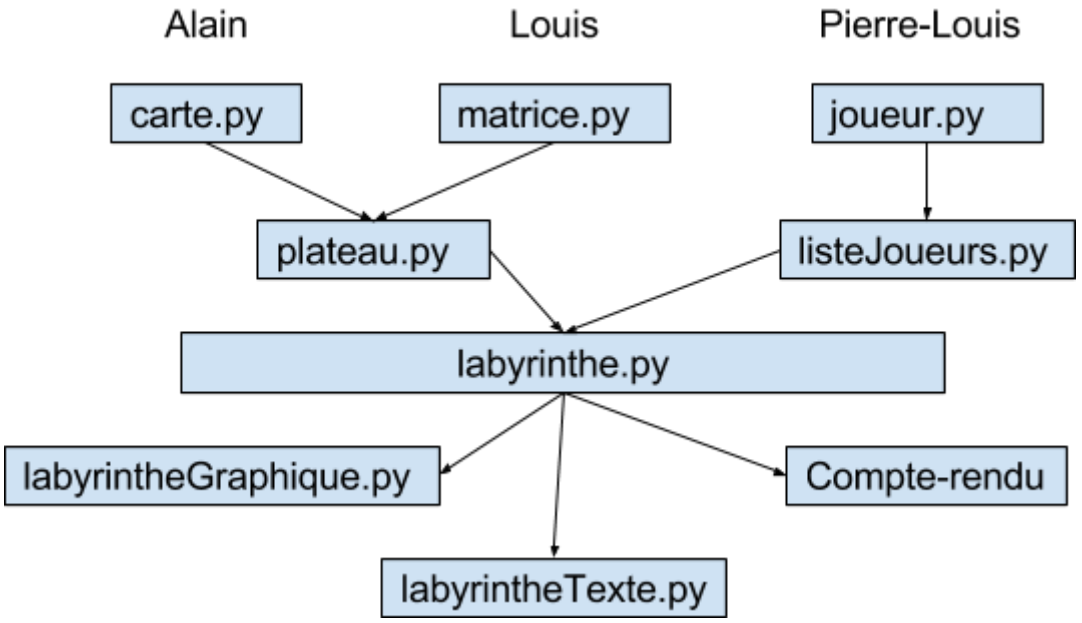
Dossier de programmation Jeu Labyrinthe en Python

Groupe 2A
Bara Alain
Bertrand Pierre-Louis
Bonduel Louis

1A21
Année scolaire 2017-2018
Projet de fin de premier semestre

Table des matières :

Répartition du travail



Alain	Louis	Pierre-Louis
carte.py	matrice.py	joueur.py
plateau.py		listeJoueurs.py
labyrinthe.py		
labyrintheGraphique.py	labyrintheTexte.py	Compte-rendu

Méthodes de tests adoptées

Pour tester nos fonctions, nous avons utilisé deux techniques différentes.
Pour les fonctions qui ne retournent des variables, nous avons utilisé des asserts. Mais pour celles qui n'en retournent pas, nous avons exécuté le fichier dans un terminal avec la commande "`python3 -i fichier.py`". Avec cette commande, on peut donc appeler les fonctions pour tester leur bon fonctionnement.

Choix de structures de données utilisées

Pour stocker les données de la partie, nous avons utilisé un dictionnaire de données contenant 4 clés :

- `'Joueurs'`
- `'Phase'`
- `'CoupPrecedent'`
- `'Plateau'`

`['Joueurs']` :
Cette partie contient une liste. Dans cette liste, on retrouve à l'indice 0 le numéro du joueur courant. Ensuite, on retrouve une liste par joueur, qui contient le nom du joueur sous la forme d'une chaîne de caractères, puis une liste contenant les différents trésors qu'il lui reste à trouver.

`['Phase']` :
Cette partie contient un entier, qui correspond à la phase actuelle du tour. Cet entier est égal soit à 1, soit à 2.
La première phase du tour pour un joueur est la phase où le joueur fait (ou non) tourner la pièce, avant de l'insérer dans le plateau de jeu.
La seconde phase du tour pour un joueur est la phase où le joueur déplace son pion sur les cases du plateau afin de récupérer son prochain trésor.

`['CoupPrecedent']` :
Cette partie contient les détails du coup précédent. Elle est utilisée pour vérifier si un coup n'est pas interdit. Le coup interdit est la coup qui consiste à annuler l'action du joueur précédent.
Elle contient la direction du déplacement (sous la forme d'un caractère 'N', 'O', 'E' ou 'S') ainsi que la colonne/ligne concernée par ce déplacement (sous la forme d'un entier 1, 3 ou 5).

`['Plateau']` :
Cette partie contient toutes les informations du plateau de jeu nécessaires au bon déroulement du programme, qui n'ont pas étés stockées précédemment.
Elle est composée d'un dictionnaire de données qui contient :

- `'nbJoueurs'` : le nombre de joueurs de la partie (sous la forme d'un entier)
- `'nbTresors'` : le nombre de trésors actuellement présents dans le plateau (sous la forme d'un entier. Attention ce nombre est différent d'un nombre de trésors que l'on entre lorsqu'on commence une nouvelle partie, car il change au cours des évènements de la partie.

- `'plateau'` : contient la matrice avec le nombre de lignes, le nombre de colonnes puis les valeurs qui contient les 49 cartes présentes sur la plateau ainsi que la 50ème carte non présente sur le plateau que le joueur insère dans le jeu.

Chaque carte (sur le plateau ou non) est stockée de la même forme : un dictionnaire avec 3 clés :

- `'murs'` : contient les directions de la case où se trouvent des murs (dans une liste contenant de 0 à 4 chaînes de caractères parmi `'nord'`, `'est'`, `'sud'`, `'ouest'`)
- `'tresor'` : si un trésor se trouve sur cette carte, ici sera stocké l'entier représentant le trésor en question (dans une liste).
- `'pions'` : si un ou plusieurs pions se trouve(ent) sur cette carte, l'entier représentant leur numéro de joueur sera stocké ici (dans une liste)

Principaux algorithmes implémentés

- Fichier matrice.py : les principaux algorithmes implémentés sont les décalages de colonnes/ligne, les getVal et setVal, et la création de la matrice.
- Fichier joueur.py et listeJoueurs.py : les principaux algorithmes implémentés sont la création de listes, les append, la recherche d'éléments dans une liste, la suppression d'éléments et la notion d'aléatoire (pour mélanger les trésors ou pour choisir au hasard le joueur qui commence la partie).
- Fichier carte.py : l'algorithme implémenté est principalement celui de la création d'une carte avec ses différentes composantes.
- Fichier plateau.py : c'est celui de MarquageDirect qui a été implémenté, qui permet de déterminer différents chemins entre les cases du labyrinthe. On y ajoute également un algorithme permettant de stocker ce chemin. A cela, on y ajoute celui de la création du plateau, avec ses différentes composantes.
- Fichier labyrinthe.py : les algorithmes les plus importants sont ceux qui permettent d'organiser un tour, d'en terminer, de gérer les décalages de colonnes, et bien sûr la création du labyrinthe.
- Fichier labyrintheTexte.py : la fonction principale est demarrer, elle utilise toutes les autres fonctions dans le bon ordre et permet le déroulement d'une partie.

Bugs connus de notre programme

- Impossible d’atteindre la 6ème ligne. (labyrintheGraphique.py)

Extensions que nous avons adopté

Nous n’avons pas inclus d'extensions, mais si nous avions eu le temps, nous aurions surement ajouté des effets sonores au jeu afin de le rendre plus vivant.

Annexes

Alain Bara

- Jour 1 : carte.py, plateau.py
- Jour 2 : plateau.py
- Jour 3 : plateau.py, labyrinthe.py
- Jour 4 : labyrinthe.py, labyrintheGraphique.py, aide Louis pour labyrintheTexte.py

Pierre-Louis Bertrand

- Jour 1 : joueur.py, listeJoueurs.py
- Jour 2 : listeJoueurs.py, joueur.py (versionObjet)
- Jour 3 : Compte rendu, labyrinthe.py, listeJoueurs.py (versionObjet)
- Jour 4 : labyrinthe.py, Compte rendu, diaporama

Louis Bonduel

- Jour 1 : matrice.py, plateau.py
- Jour 2 : plateau.py, matrice.py (versionObjet), labyrinthe.py
- Jour 3 : labyrinthe.py
- Jour 4 : labyrinthe.py, labyrintheTexte.py