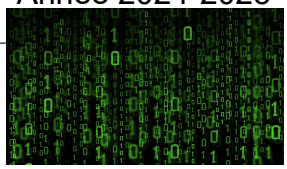
IUT d'Orléans
BUT 1 Informatique

Année 2024-2025



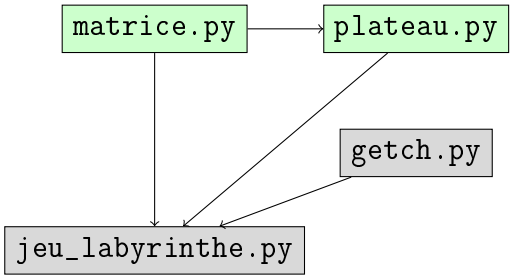
R1.01 Initiation au développement
Feuille de TP n°12
Algorithme du plus court chemin

Objectifs de la feuille

- Utiliser une des API construites les semaines précédentes pour coder une application qui manipule des matrices
- Implémenter l'algorithme du plus court chemin vu en TD la semaine dernière

Exercice 1 *Le jeu du labyrinthe : les modules matrice et plateau*

Anakin a terminé le travail d'analyse de son jeu. Ses conclusions sont les suivantes : le jeu a besoin de quatre modules dont il donne un graphe indiquant leurs dépendances.

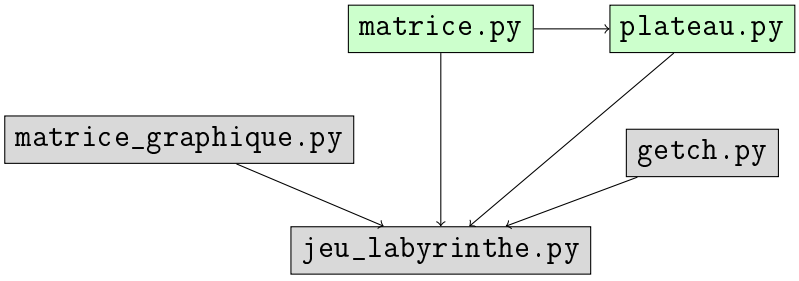


- Le module `matrice` est une API qui permet de manipuler des tableaux 2D.
- Le module `getch` est un module qui permet de demander à l'utilisateur un seul caractère sur l'entrée standard. Ce module est fourni.
- Le module `plateau` est une API qui permet de manipuler un plateau de jeu avec des couloirs, des murs, un personnage et un fantôme.
- Le module `jeu_labyrinthe` est le programme principal : il permet de lancer le jeu et gère les interactions avec l'utilisateur. Anakin a déjà codé ce module.

1.1 Complète le code du module `matrice` . Pour vous aider, le fichier `test_matrice.py` contient des tests pour ce module.

1.2 Complète le code du module `plateau` . Pour vous aider, le fichier `test_plateau.py` contient des tests pour ce module.

Anakin ajoute le module `matrice_graphique` pour pouvoir jouer en mode graphique.



1.3 Dans la fonction `joue` du module `jeu_labyrinthe` remplace la ligne `affichage_graphique = None` par la ligne `affichage_graphique = matrice_graphique.MatriceGraphique(mon_plateau)` puis relance le jeu.

Remarque : il sera peut-être nécessaire d'installer pygame. Pour cela, ouvre un terminal et tape la commande :

```
python3 -m pip install -U pygame --user
```

Exercice 2 *Le jeu du labyrinthe : amélioration du programme principal*

Voici le scénario imaginé par Anakin qui lui a permis de coder la fonction `joue` dans le module `jeu_labyrinthe` :

1. L'utilisateur lance l'application
   - A. → Le programme met en place le plateau de jeu et l'affiche

     Tant que le jeu n'est pas terminé :
   - B. → Le programme demande à l'utilisateur de choisir une direction.
     On utilisera par exemple les lettres s, w, q, z
2. L'utilisateur choisit une direction
   - C. → Le programme vérifie si le déplacement du personnage est valide (le personnage ne peut pas sortir du plateau de jeu ni passer à travers les murs). Si le déplacement est valide, le programme déplace le personnage dans la direction choisie, sinon, le personnage reste sur place
   - D. → Le programme déplace le fantôme *de façon intelligente*
   - E. → Le programme affiche le plateau de jeu `affiche(jeu)`
   - F. → si le personnage se trouve sur la même case que le fantôme, la partie est terminée : le joueur a perdu
   - G. → si le personnage se trouve sur la case sortie (tout en bas à droite) la partie est terminée : le joueur a gagné
   - H. → Sinon, on recommence avec l'étape B

2.1 Modifie le code du module `jeu_labyrinthe` de façon à ce que l'utilisateur puisse choisir un autre labyrinthe que le labyrinthe par défaut.

2.2 Que se passe-t-il si l'utilisateur choisit un labyrinthe duquel le personnage ne peut pas sortir ? Modifie le code de façon à résoudre le problème.

Exercice 3 *Le jeu du labyrinthe : ajout de fonctionnalités*

3.1 Modifie le programme de façon à proposer un mode "FLASH" dans lequel le fantôme peut se déplacer de deux cases.

3.2 Modifie le programme de façon à proposer un mode "EXPERT" dans lequel il y a plusieurs fantômes dans le labyrinthe.