Projet pyChase

Modalité du projet Durée

• 5 semaines (jusqu'à la semaine 14 inclue)

Évaluation:

L'évaluation sera décomposée en deux parties : qualité et algorithmes. Voici les sous-parties et le poids de chacune :

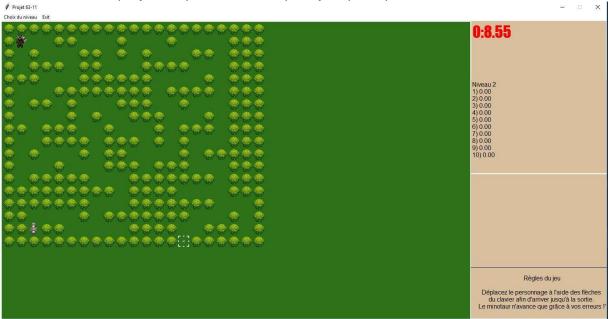
- Qualité du code (15%)
 - o Docstring (6%)
 - Typage (variables, paramètres et types de retour) (5%)
 - Documentation du code complexe (4%)
- Fonctionnalité (35%)
- Logique et algorithmique (25%)
- Qualité du programme (25%)
 - o Décomposition de fonctions et modules
 - Structures de données
 - o Programmation défensive (test de paramètres, valeur nulle, etc.)

Temps à disposition

• Heures de laboratoire

Description du projet

L'objectif de ce projet est de mettre à profit toutes les notions apprises lors des cours du module 63-11. Le projet en question est un petit jeu qui se présente sous la forme suivante :



L'objectif du jeu

Le but du jeu est d'atteindre la sortie adans le temps imparti sans se faire attraper par le minotaure. Le minotaure n'avance que lorsque le joueur commet une erreur. Par erreur nous entendons le fait que le joueur avance contre un mur avancer de 5 cases sur le chemin le plus court pour atteindre le joueur.

Quelques règles

- Lorsque le joueur atteint une sortie, il disparaît de l'écran et le jeu s'arrête.
- Lorsque le minotaure atteint un joueur, le joueur disparaît, le minotaure prend la position du joueur et le jeu s'arrête.

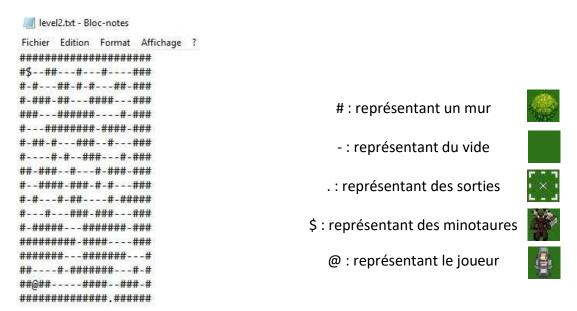
Instructions

Starkov Andrei

Semaine 9 : Lecture du projet et analyse (haut niveau) du code fourni

Semaine 10 : Charger la carte et Condition de jeu

Dans le dossier niveaux se trouve 3 fichiers textes. Chaque fichier texte contient la description d'un niveau détaillé comme ceci :



Pour charger la carte, il faudra donc lire le fichier sélectionné caractère par caractère et remplir les structures de données permettant ensuite l'affichage de la carte. Vous devez choisir quels types de structures de données utiliser et combien sont nécessaires selon vous.

La seule structure obligatoire est celle permettant de calculer le chemin le plus court entre un minotaure et le joueur. Pour cela vous devez construire une matrice (liste de liste) comportant toutes les <u>entités</u> selon leurs coordonnées sur le niveau. Par exemple :

Pour le niveau : ### la structure ressemblera à : [[mur, mur, mur], #\$# [mur,minotaure,mur], ### [mur,mur,mur]]

Le premier mur (haut à gauche) étant en x = 0 et y = 0.

Il est donc demandé dans cette période de développer les fonctions charger_niveau et jeu_en_cours

Semaine 11-12: Mouvement

Vous serez amenés à développer la fonction definir_mouvement et effectuer mouvement permettant de déplacer le personnage.

Vous devez prendre en compte les « Quelques règles » listées ci-dessus lors du développement des mouvements.

Quelques conseils:

- Essayez d'imaginer ce qu'implique un déplacement par rapport à une entité ainsi que sur vos structures de données.
- Essayez, à priori, de déterminer les 4 cas de déplacements possible (je ne parle pas de haut/bas/gauche/droit).

Semaine 13: Scores

Finalement, dans le répertoire scores, fichier scores.txt, se trouvent les scores pour chaque niveau. Ceux-ci sont stockés sous cette forme :

<numNiveau>;score1;score2;score3;score4;score5;score6;score7;score8;score9;score10;

Exemple pour le niveau 1:1;10.24;11.34;11.85;11.86;12.02;13.57;14.99;0.0;0.0;0.0;

- 4 fonctions seront à développer pour la gestion des scores :
 - chargement_score
 Fonction chargeant les scores depuis un fichier.txt et les stockent dans un dictionnaire.
 - 2. maj score

Fonction mettant à jour l'affichage des scores en stockant dans un str. C'est l'affichage visible sur la droite du jeu.

("Niveau x

1) 11.25

2) ... ").

3. sauvegarde score

Fonction enregistrant un nouveau score réalisé par le joueur. Le calcul de score est le suivant :

temps_initial - temps_restant

Ce score est arrondi à 2 chiffres après la virgule et stocké en tant que float.

4. update score file

Fonction qui va écrire les scores dans le fichier scores.txt. Cette fonction est appelée lors de la fermeture de l'application

Fonctions et outils fournis

Dans le script outils.py vous trouverez toutes les fonctions à votre disposition. Voici la liste des fonctions que vous trouverez :

63-11 – Fondement de la programmation Issom David-Zacharie, Perrotte Sonia, Humbert Jérôme, Starkov Andrei

- creer image()
- creer mur()
- creer minotaure()
- creer sortie()
- creer personnage()
- creer case vide()
- coordonnee x()
- coordonnee y()
- est egal a()

Pour comprendre ce qu'elles font, référez-vous à leur docstring. Pour les utiliser, vous pouvez directement les appeler dans votre code (en faisant attention de bien utiliser leurs paramètres dans le bon ordre).

Dans le fichier fourni\personnage.py vous trouverez des fonctions que l'on pourra utiliser sur l'entité Joueur.

- est mort()
- est_attraper()
- a fini()
- est sorti()

Pour comprendre ce qu'elles font, référez-vous à leur docstring. Pour les utiliser, vous pouvez directement les appeler dans votre code sur votre entité joueur. Par exemple joueur.est sorti().

liste_image: [] Vous l'utiliserez uniquement lorsque vous avez besoin de la fonction
creer_image(). Chacun de ses index fait référence à une image:

- [0]: image du mur
- [1]: image de la sortie
- [2]: image du minotaure
- [3]: image de la caisse sur une cible
- [4]: image du joueur
- [5]: image du joueur sur la cible
- [6]: image du sol

Donc si je souhaite remplacer l'image par l'image du joueur -> liste image[4]

dict_score qui est le dictionnaire stockant les scores. Celui-ci est structuré de cette manière : key = numéro du niveau, value = tableau de scores ordonnés (mais les 0 sont à la fin)

ainsi : { 1 : [10.24,11.34,11.85,11.86,12.02,13.57,14.99,0.0,0.0,0.0]}