Projet: Qix.

Le but de ce projet est de réaliser le jeu classique Qix[1]. Ce jeu se joue sur un plateau rectangulaire, sur lequel circulent le "Qix" (sur l'aire de jeu) et deux "Sparx" (sur les bords du plateau). Le but du joueur est de recouvrir au moins 75 % de l'aire du plateau en traçant des formes selon des contraintes mentionnées ci-dessous. Le joueur perd une vie s'il se fait directement toucher par un Sparx ou si le Qix touche un chemin qu'il est en train de tracer.

Actions du joueur. Le joueur contrôle le clavier pour dessiner des chemins polygonaux. Lorsqu'un chemin se ferme, il divise la zone de jeu en deux parties : celle qui ne contient pas le Qix se retrouve coloriée. Au fur et à mesure que le jeu avance, le joueur peut se déplacer sur les bords de la zone coloriée et les utiliser comme points de départ ou d'arrivée de ses chemins. Les chemins doivent être simples, autrement dit, ils ne peuvent pas s'auto-intersecter.

Déplacement des ennemis. Le Qix se déplace de manière aléatoire en continu, sans chercher à toucher le joueur ou ses chemins. Ses déplacements sont confinés à l'espace non recouvert par le joueur, et il est donc possible d'"enfermer" le Qix. Chaque Sparx se déplace toujours dans le même sens et suit le contour de l'aire de jeu non recouverte par le joueur : au début du jeu, les Sparx se déplacent donc sur le bord du plateau, mais au fur et à mesure que le joueur recouvre le plateau, les Sparx finiront par suivre le contour externe des polygones dessinés — c'est-à-dire les côtés qui séparent une partie coloriée d'une partie non coloriée. Ils ne peuvent donc pas grimper le long d'un chemin que le joueur est en train de tracer.

Fin de la partie. La partie se termine quand le joueur n'a plus de vies, ou quand il a réussi à colorier au moins 75 % de l'aire de jeu.

Le projet est à réaliser en plusieurs phases. Lisez bien le sujet dans son intégralité avant de commencer le travail, car les choix que vous effectuerez au début auront un impact sur la suite.

1 Le programme à réaliser : le jeu de base

Dans sa version de base, l'aire de jeu est initialement vide. L'avatar du joueur est un cercle placé centré au milieu du côté inférieur de l'aire de jeu. Vous devrez implémenter le jeu avec les règles suivantes :

1. Le joueur se déplace sur les bords de l'aire de jeu à l'aide des flèches du clavier. Quand il souhaite dessiner un chemin, il doit presser "Enter" tout en se déplaçant perpendiculairement à un côté de l'aire de jeu ou de la zone coloriée. Il peut ensuite se déplacer dans n'importe quelle direction (à 0°, 90°, 180° ou 270°). Le déplacement est contrôlé par le clavier ; si le joueur intersecte le chemin en cours, il perd une vie.
2. Les Sparx sortent du centre du côté supérieur de l'aire de jeu et longent son bord dans des sens opposés.
3. Vous êtes libres de décider de la forme du Qix. Cependant, des points supplémentaires seront accordés aux formes les plus intéressantes.
4. Le joueur a trois vies. Il passe au niveau suivant quand il a conquis au moins 75 % de l'aire de jeu. Dans le niveau suivant, le Qix et les Sparx bougent plus rapidement.

Pour que le jeu fonctionne, il vous faut entre autres :

1. Consultez `https://fr.wikipedia.org/wiki/Qix` pour plus d'informations, et les vidéos suivantes : `https://www.youtube.com/watch?v=2x7GiHrJHwQ` et `https://www.youtube.com/watch?v=Ga-aMRZi6fs`, pour voir des exemples de parties.

— une fonction qui détecte quand les Sparx ou le Qix tuent le joueur ;
— une fonction qui dessine et calcule l'aire d'une région définie par un chemin.

Il vous appartient de choisir les paramètres (vitesse, taille du Qix, etc.). Le jeu doit être amusant.

2 Variantes

Dans un deuxième temps, vous devrez fournir une version du jeu avec un menu initial qui permet d'utiliser une combinaison des alternatives suivantes. Les choix d'implémentation sont laissés à votre appréciation.

1. **Scores** : le nombre de vies restantes et l'aire gagnée par le joueur sont constamment affichés, de même que les points obtenus jusqu'ici. Pour le calcul des points : chaque zone coloriée donne un nombre de points proportionnel à son aire.
2. **Vitesse** : un bouton doit permettre de changer la vitesse du joueur au moment de dessiner un chemin ; une fois que le chemin est en cours de traçage, la vitesse ne change plus jusqu'à sa terminaison. Une aire coloriée à l'aide d'un chemin à plus grande vitesse donne un score plus faible : on attribue $A \times c$ points à une zone coloriée d'aire A, où c dépend de la vitesse de coloriage (par exemple : $c = 500$ en mode lent, $c = 250$ en mode rapide).
3. **Obstacles** : rajoutez des obstacles, que le joueur ne peut pas traverser (mais une zone coloriée peut contenir un obstacle).
4. **Bonus** : l'aire de jeu contient initialement un certain nombre de pommes ; lorsque le joueur en touche une, elle disparaît et il devient invincible pendant 3 secondes (le contact du joueur avec un Sparx ou de son chemin avec le Qix n'a donc aucun effet durant cette période).
5. **Deux joueurs** : une version avec deux joueurs, chacun ayant sa couleur, qui démarrent chacun de leur propre coin inférieur de l'aire de jeu. Le gagnant du niveau est celui qui aura recouvert le plus grand pourcentage de l'aire de jeu (le niveau change quand les joueurs ont colorié à eux deux au moins 75 % de l'aire de jeu). Si le joueur A trace un chemin qui croise le chemin que le joueur B est en train de tracer, le joueur A perd une vie. Il vous appartient de décider comment résoudre le cas où A se retrouve piégé dans un coloriage de B pendant le traçage de son chemin.
6. **Sparx "internes"** : les Sparx peuvent à présent circuler sur les segments internes des zones coloriées.
7. **Niveaux** : les niveaux plus difficiles rajouteront des Sparx et des Qix. Il vous appartient de décider comment résoudre décider de la zone à colorier en présence de plusieurs Qix.

3 Les bonus et améliorations

Finalement, vous devrez implémenter au moins deux des améliorations suivantes :

1. **IA** : ajoutez un mode où l'ordinateur joue tout seul, que l'on peut lancer avec une option spéciale de la ligne de commande. Il peut y avoir un ou deux joueurs virtuels (donc soit on joue contre l'ordinateur, soit l'ordinateur joue contre lui-même). Dans chaque cas, réfléchissez à des stratégies et motivez vos choix.
2. **Classement** : si vous implémentez l'IA, rajoutez de quoi mémoriser / afficher le classement des meilleurs scores.

3. **Mode circulaire** : dans cette variante, le tableau est circulaire et il tourne continuellement à une vitesse constante.
4. **Création des obstacles, bonus et malus** : rajoutez la possibilité de charger une aire de jeu agrémentée d'obstacles à partir d'un fichier texte contenant leur description et leurs emplacements. Vous choisirez un encodage pour les obstacles et leurs propriétés.
5. **Pause et sauvegarde** : rajoutez la possibilité de mettre en pause le jeu et de reprendre la partie, et de sauvegarder l'état du jeu à ce moment-là, ainsi que la possibilité de charger une partie sauvegardée en début de jeu.
6. **Sauvegarde des paramètres** : rajoutez un fichier de configuration pour le jeu afin de fixer les valeurs des différents paramètres (taille de la fenêtre, de l'aire de jeu, vitesse, ...).

4 Consignes de rendu

Le projet se déroulera en trois grandes phases : un premier rendu aura lieu fin octobre 2023, un second rendu fin novembre, et finalement un rendu final fin décembre 2023. Pour les rendus, il sera impératif de suivre les consignes précisées ci-dessous, sans quoi vous perdrez des points. En particulier :

— Ce projet est à faire en binôme (s'il est nécessaire de faire une exception, contactez les enseignants). Vous devrez sélectionner votre binôme sur e-learning dans la semaine suivant la publication du sujet. Pour des raisons d'organisation, les binômes doivent se constituer de membres du même groupe de TP.
— **Vous DEVEZ utiliser `fltk`**. L'utilisation de modules autres que les modules standards de Python est interdite ; en cas de doute, n'hésitez pas à poser la question.
— **Votre programme devra impérativement s'exécuter sans problème sur les machines de l'IUT.** Prévoyez donc bien de le tester sur ces machines en plus de la vôtre et d'adapter ce qui doit l'être le cas échéant (par exemple, les vitesses de déplacement). Il sera donc utile de permettre à l'utilisateur de préciser certaines options auxquelles vous attribuez des valeurs par défaut plutôt que de devoir modifier le code à chaque exécution.

4.1 Premier rendu

L'objectif général à atteindre pour le premier rendu est d'avoir un jeu fonctionnel et robuste avec :
— Le jeu de base (section 1) complet.
— Au moins une variante de la liste donnée en section 2.

Ne commencez pas les variantes de la section 2 avant d'avoir réalisé le jeu de base.

La date limite pour ce premier rendu est le **jeudi 26 octobre 2023 à 23h55**. Passé ce délai, la note attribuée à votre projet sera 0. Les séances de SAÉ de la semaine suivante seront consacrées à la vérification de votre rendu et à amorcer la seconde partie. Vous déposerez sur la plate-forme e-learning une archive contenant :

1. **les sources de votre projet**, commentées de manière pertinente (quand le code s'y prête, pensez à écrire les doctests). De plus :
   — les fonctions doivent avoir une longueur raisonnable ; n'hésitez pas à morceler votre code en plusieurs fonctions auxiliaires pour y parvenir ;
   — plus généralement, le code doit être facile à comprendre : utilisez des noms de variables parlants, limitez le nombre de tests, et veillez à simplifier vos conditions ;

   - quand cela se justifie, regroupez les fonctions par thème dans un même module ;
   - chaque fonction et chaque module doit posséder sa *docstring* associée, dans laquelle vous expliquerez leur fonctionnement, et pour les fonctions, ce que sont les paramètres ainsi que les hypothèses faites à leur sujet.
2. un **fichier texte** `README.txt` expliquant :
   - les bonus qui ont été implémentés (au moins un) ;
   - l'organisation du programme ;
   - les choix techniques ; et
   - les éventuels problèmes rencontrés.

   Vous pouvez y ajouter tous les commentaires ou remarques que vous jugez nécessaires à la compréhension de votre code.

Si le code ne s'exécute pas, la note de votre projet sera 0. Vous perdrez des points si les consignes ne sont pas respectées, et il va sans dire que si deux projets trop similaires sont rendus par 2 binômes différents, la note de ces projets sera 0.

4.2 Second rendu

L'objectif pour le second rendu est toujours d'avoir un jeu fonctionnel et robuste avec le jeu de base et toutes les variantes.

Les consignes données pour le premier rendu restent valables pour le second rendu. La date limite pour le second rendu est le **jeudi 30 novembre 2023 à 23h55**.

4.3 Troisième rendu

L'objectif est toujours d'avoir un jeu fonctionnel et robuste avec le jeu de base et toutes les variantes et **au moins deux bonus de la section 3.**

La date limite pour le second rendu est le **jeudi 21 décembre 2023 à 23h55**. Des mini-soutenances seront organisées la semaine suivante, au cours desquelles vous ferez une démonstration sur une machine des salles de TP et serez interrogés sur le projet (les choix techniques et algorithmiques que vous avez faits, les difficultés rencontrées, l'organisation de votre travail, ...). Les contributions de chaque participant seront évaluées, et il est donc possible que tous les participants d'un même groupe n'aient pas la même note.