El Quoridor

Dans notre groupe de trois, nous avons choisi Python comme langage de programmation pour notre projet de création de jeux vidéo pour plusieurs raisons.

Tout d'abord, Python était le langage que nous connaissions le mieux dans le domaine de la création de jeux vidéo. Nous avions tous une certaine expérience de programmation en Python, et nous savions que Python est souvent utilisé dans le développement de jeux vidéo grâce à sa facilité d'utilisation et à ses nombreuses bibliothèques disponibles.

De plus, nous avons également considéré que Python était un langage de programmation relativement simple à apprendre et à utiliser, avec une syntaxe claire et concise. Nous n'avions pas beaucoup de temps pour développer notre jeu, donc le choix de Python nous a permis de nous concentrer sur l'aspect créatif plutôt que sur l'apprentissage d'une nouvelle syntaxe de programmation

Nous avons également étudié le langage C, qui est souvent utilisé dans le développement de jeux vidéo, mais nous avons décidé de ne pas l'utiliser pour notre projet. En effet, nous avons constaté que nous n'avions pas suffisamment de connaissances approfondies en C pour l'utiliser efficacement. Nous avons préféré nous concentrer sur notre confort et notre familiarité avec Python pour mener à bien notre projet.

En conclusion, notre choix de Python comme langage de programmation pour notre projet de création de jeux vidéo était basé sur notre expérience et notre familiarité avec le langage, sa facilité d'utilisation et les nombreuses bibliothèques disponibles. Nous avons également tenu compte de nos limites en matière de connaissances en C et avons donc choisi de nous en tenir à ce que nous connaissions le mieux.

PyPygame: Nous avons utilisé la bibliothèque Pygame pour créer notre jeu vidéo, car elle fournit une interface graphique pour l'affichage de l'animation, des sons et des événements de clavier et de souris. Pygame est également facile à utiliser et offre de nombreux exemples et tutoriels pour vous aider à apprendre à l'utiliser rapidement.

Sys: Nous avons utilisé la bibliothèque Sys pour accéder à des fonctions du système d'exploitation telles que l'écriture de messages d'erreur, la redirection de la sortie et la gestion des paramètres de ligne de commande. La bibliothèque Sys est facile à utiliser et offre des fonctions utiles pour la gestion des flux d'entrée et de sortie.

Subprocess: Nous avons utilisé cette bibliothèque Subprocess pour lancer des processus externes depuis notre programme Python. Nous avons utilisé Subprocess pour lancer les scripts et des programmes externes en arrière-plan tout en continuant à exécuter notre programme principal. Subprocess est une bibliothèque très utile et flexible pour la gestion des processus externes.

Socket: Nous avons utilisé cette bibliothèque Socket pour la communication entre notre jeu et d'autres ordinateurs ou appareils connectés à Internet. Nous avons utilisé la bibliothèque Socket pour créer des connexions réseau et envoyer des données entre les machines. La bibliothèque Socket est très utile pour la communication entre ordinateurs sur un réseau. \_

\_thread: Nous avons utilisé cette bibliothèque \_thread pour créer des threads en Python. Les threads sont des processus légers qui peuvent être exécutés simultanément dans un programme Python. Nous avons utilisé \_thread pour exécuter plusieurs tâches simultanément et améliorer les performances de notre jeu. \_thread est une bibliothèque simple mais très utile pour la gestion des threads en Python.