Bobo Jules
Lior Zana

Projet Tétriste

Notice développeur

Sommaire

1. Introduction

Nous allons d'abord énoncer les fonctionnalités implémentées dans le jeu, en les commentant rapidement sur leurs aspects tels que leur fonctionnement, la complexité et les difficultés rencontrées enfin nous verrons comment améliorer le programme.

Tout d'abord un rapide résumé des fonctionnalité :

Le jeu sera affiché dans le terminal, les inser4ons à droite et à gauche fonctionnent, de même pour les suppressions de trois pièces consécutives. Il y a possibilité de faire des décalages vers la gauche ou vers la droite. Les 5 pièces suivantes sont affichées et il y a aussi un système de combo. Le système de sauvegarde est fonctionnel ainsi qu'un système

de meilleurs scores. La partie s'arrête après qu'il y ait plus de 15 pièces présentes sur le plateau de jeu. Il y a possibilité de relancer une partie ou de stopper à tout moment.

2. Fonctionnalitées du programme

2.1.Les structures

Nous utilisons deux structures :

Une structure Maillon contenant les caractéristique de couleur de forme et formant des chaines circulaire doublement liées sur les autres maillons de même couleur et forme. **Attention**: Contrairement au sujet, on utilise aussi une chaîne doublement circulaire sur les maillons du jeu. Ce choix nous permet de supprimer à temps constant à droite de façon facile, sinon on serait obligé d'avoir une suppression à la fin de temps linéaire ou de conserver plusieurs pointeur sur les éléments de fin ce qui complexifierait le code.

Une structure liste contenant un pointeur un premier et dernier maillon, on utilise 9 listes au total, une pour le jeu et une pour chacune des couleurs et des formes.

2.2 Les fonctionnalité par rapport au sujet

- Affichage du jeu dans le terminal (ASCII art)
- Les insertions à gauche et à droite fonctionnent .
- La suppression de trois pièces consécutives ayant couleur ou forme commune fonctionne
- Allocation et libération de la mémoire partiel d'après les derniers test (suite à des soucis avec les ordinateur de Dauphine nous n'avons puis finir de tester valgrind)
- Les décalages à gauche fonctionnent via les listes de couleurs et formes
- Système de score affiché (on marque des points à chaque suppression de triplets, avec plus de points si obtenu via décalage car plus difficile à voir)
- La partie s'arrête après plus de 15 pièces présentes sur le plateau de jeu
- Sauvegarde/chargement d'une partie (on stocke dans un fichier texte l'état du jeu)
- Système de meilleurs scores : on sauvegarde dans un fichier texte les 10 meilleurs scores réalisés avec le nom de l'auteur, stockés dans l'ordre décroissant
- On affiche les cinq prochaines pièces à venir dans le jeu plutôt qu'une seule
- Calculer un score plus réaliste qui donne plus de points pour les disparitions en cascade de triplets (combo), ou bien la création de 4-uplet ou 5-uplet

2.3 fonctionnement du système de score et de combo

On attribue 5 points lors d'une insertion provoquant un triple, 15 points si c'est un triple de forme et de couleur(exemple: 3 carrés jaune de suite).
Lors d'un décalage un triple donne 15 points, 20 pour un quadruple, 25 pour un quintuple.

Le système de combo est multiplicateur si le compteur est supérieur ou égal à 3 et fonctionne de la manière suivant: le compteur est initialisé à 0 et augmente de 1 par suppression effectuer, un triple provoqué par l'insertion l'incrémente de 1, un décalage peut l'incrémenter de 1 ou plus en cas de cascade. Il est réinitialisé à 0 si une action (décalage ou insertion ne provoque aucune suppression). Un combo à 3 multipliera les points de 3 et ainsi de suite. La stratégie pour faire un haut score est donc d'accumuler les pièces sur le plateau et de faire des triples en cascade.

2.4 les fonctions majeurs

Les fonctions d'insertions début et fin:

De complexité T(1) elles sont le pilliers de notre code puisqu'elles sont réutiliser partiellement dans le chargement de la sauvegarde, et dans le décalage.
Elles sont segmentées en 3 parties : l'ajout dans le jeu, dans une liste couleur et dans une liste forme qui fait avec la fonction addListeSpec. Ces 2 dernières parties permettent de réparer les liens que l'on casse dans après un décalage. Ces segmentation en sous fonctions permettent d'éviter la redondance sans le code et de le raccourcir.

Ces fonctions nous ont donné du mal au début car suite à un mauvais choix de structure le code était très complexe et la complexité était au mieux T(n).

La fonction décalage:

De complexité T(n) l'algorithme et plus simple qu'en apparence, on fait un rotation vers la direction indiqué des caractéristique couleur et forme de la liste indiqué par le joueur en temp T(n).

Exemple : on choisit de faire un décalage vers la gauche pour les rouges. On parcourt donc la liste Tabliste[4]en modifiant pour chaque Maillon sa forme par la forme du Maillon suivant.

L'ordre des pièces étant changé, il fait réparer les liste couleurs et formes. On les réinitialise donc et on parcourt le plateau et ajoute chaque élément du plateau dans sa liste forme et couleur qui correspond donc n*T(1) car les insertions sont de temps constant. Grâce à notre choix d'avoir 9 listes et d'avoir des insertions très efficaces, le décalage est très rapide et ne nous a pas posé de soucis majeur durant l'implémentation.

Les fonctions de suppression

Nous effectuons 3 sortes de suppressions différentes, celles de début et de fin de temps T(1) qui suppriment si nécessaire les 3 éléments concernés. Celle de décalage qui compte le nombre d'éléments concernées (triple, quadruple …) et les supprime en retournant ce nombre pour calculer les points.
Elles appellent toutes une suppression individuelle qui supprime un élément du jeu, sa liste couleur et sa liste forme avant de libérer le maillon concerné.

Les fonctions de sauvegarde et de score T(n)

La sauvegarde écrit sur un document le score, le combo, les pièces du jeu ainsi que celle suivant suivant un code: 1ère lettre de la couleur + celle de la forme.
La charge de sauvegarde les lit puis les ajoute avec la fonction insertion fin dans les listes et le jeu.
L'algorithme de score était lui plus complexe au niveau du tri et de l'enregistrement des anciens scores. La gestion des buffers à au premier abord créé de nombreuses erreurs de segmentation.

3. Difficultés rencontrés

Une mauvaise gestion des types rendait le code complexe sans raison et un manque de liste pour chaque couleur et forme nous empêchait d'accéder instantanément aux premier éléments d'une couleur ou forme nécessaire pour les fonctions addition et décalage.

Nous avons donc changé et sommes partis sur
- Une structure Maillon qui contient directement toutes les information nécessaires. (Prochain Maillon de même couleur ...)
- Une structure Liste qui con4ent un Maillon premier et un maillon dernier comme expliqué précédemment.

Bobo Jules
Lior Zana

4. Améliorations Possibles

Les allocations mémoire sont imparfaites aux dernières nouvelles (nous n'avons pas eu de tester la dernière version suite aux soucis à Dauphine) et le fait que l'on ai 7 mallocs dans le type élément n'aide probablement pas même si on a 7 free dans la fonction libération.
Il manque l'interface graphique et l'affichage dans le terminal n'est pas le plus beau.
Il y a quelques bugs possibles si l'utilisateur entre par exemple un pseudo vide.
Enfin le système de combo peut être difficile à comprendre pour l'utilisateur.