# Rapport de TP

Programmation C avancée

TP4 : Jeu de taquin

**UNIVERSITE PARIS-EST MARNE-LA-VALLEE**

16 novembre 2019

Créé par : SAVANE Kévin

# Table des matières

# I. Introduction : Jeu de taquin

Le but de ce tp est de mettre en place un jeu de taquin, aussi appelé 15-puzzle, avec une interface graphique en C.

L'image choisie au préalable est divisée en 16 pièces, puis est mélangée automatique. Le joueur n'a plus qu'à essayer de résoudre le puzzle pour gagner la partie.

Deux types structurés représentant le plateau de jeu et ses cases, ainsi qu'une fonction permettant d'initialiser ce plateau nous sont fournis dans le sujet du tp. Nous pouvons donc nous baser sur ces trois données pour commencer notre tp.

## II. Utilisation

Pour jouer au jeu de taquin, il faudra au préalable se placer dans le dossier *src* du tp, puis lancer la commande **make** dans le terminal afin de compiler les fichiers et créer un exécutable.

Utilisez les commandes suivantes pour :

**make clean** : Supprime tous les fichiers .o ;

**make mrproper** : Supprime tous les fichiers .o ainsi que le fichier exécutable.

Depuis le terminal, utilisez la commande **./Puzzle** pour lancer le programme.

Une fois le programme compilé puis lancé, vous pouvez choisir une image parmi 14 disponibles. L'image est ensuite mélangée, et vous pouvez tenter de le résoudre une fois le mélange terminé. L'image que vous avez choisie s'affiche en double : celle de gauche est le puzzle à résoudre, celle de droite est l'image référence.

Pour déplacer une pièce adjacente à la case noire, il suffit de cliquer dessus, et elle se déplacera automatiquement.

Lorsque vous aurez complété le puzzle, un message de félicitation s'affichera, et le programme s'arrêtera. Vous pourrez ensuite relancer le programme pour recommencer.

# III. Développement

## 1. Préconception

Avant de commencer à programmer, il faut au préalable définir les fonctionnalités et les objets à mettre en place. Après avoir passé cette étape, nous avons réalisé le schéma d'inclusion suivant :

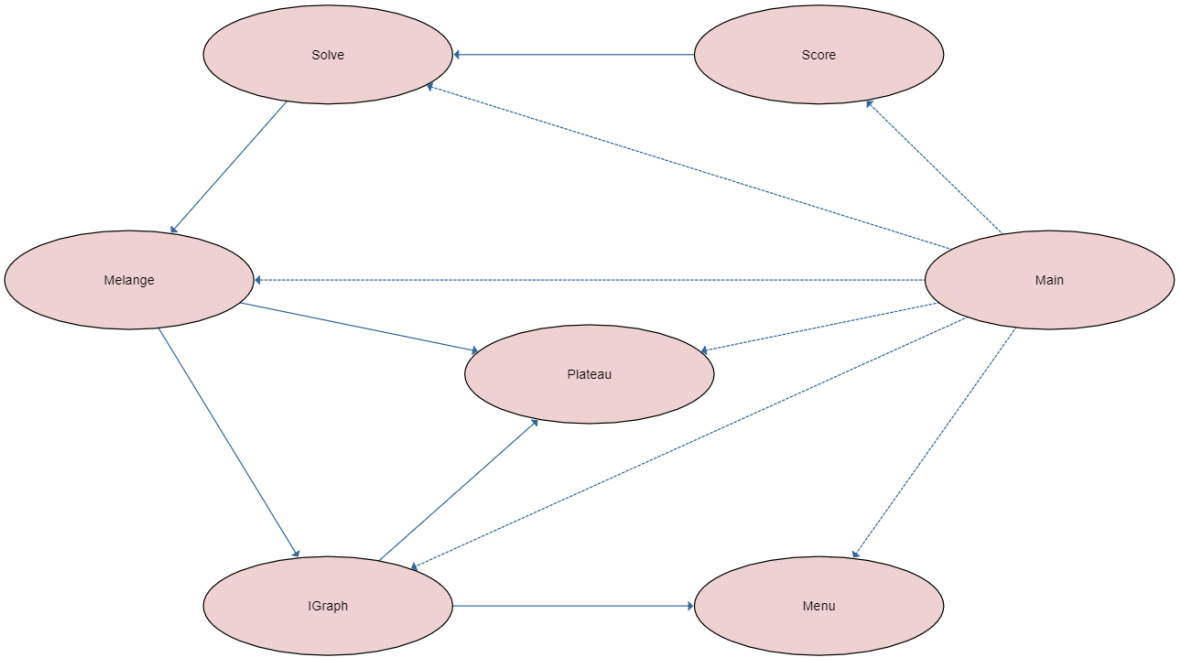


FIGURE 1 : SCHÉMA D'INCLUSION

## 2. Module Plateau

Ce module contient les types structurés et la fonction fournis dans le sujet. Il contient toutes les fonctions nécessaires à la mise en place du plateau de jeu, ainsi que les constantes définissant la taille du puzzle.

## 3. Module Melange

Le module Melange est le module regroupant toutes les fonctions utilisées pour mélanger automatiquement le puzzle. Ces fonctions permettent aux mouvements n et n+1 du mélange de ne pas s'annuler. Elles vérifient aussi la légalité d'un mouvement.

Nous pouvons noter l'absence de double boucle for dans les fonctions, nous évitant ainsi de devoir parcourir tout le plateau pour retrouver la pièce à déplacer. Etant donné que la case première case à déplacer pour le mélange est la case en position (3, 3), il suffit de récupérer ces valeurs, et d'enregistrer la nouvelle position de la case après déplacement, réduisant de cette manière la complexité de la fonction.

Ce module contient aussi la constante représentant le nombre de mouvement pour le mélange.

## 4. Module Solve

Les fonctions mettant en place la résolution du puzzle se situe dans le module Solve. Elles permettent de récupérer la position de la souris sur l'affichage graphique, puis de la convertir en position sur le plateau, mais aussi de vérifier l'état fini ou non du puzzle, ainsi que la vérification d'un coup légal.

La fonction de vérification de légalité d'un coup ressemble beaucoup à celle du module Melange. La différence majeure entre les deux est le fait que dans le module Melange, nous savons déjà que la pièce à déplacer est effectivement déplaçable. Elle aura toujours au moins deux possibilités de déplacement, tandis que dans le module Solve, comme le joueur peut cliquer sur n'importe quelle case, il n'est pas assuré que cette pièce puisse être déplacer. Une presque-duplication a donc été nécessaire.

## 5. Module Menu

Le menu du jeu contient uniquement la sélection d'image parmi une banque d'images prédéfinies.

Ce module utilise un type structuré afin de contenir les différentes images et leurs informations. Il met en place des fonctions simples de gestion de listes chaînées ainsi que des fonctions de choix d'image.

## 6. Module IGraph

Le module IGraph contient toutes les fonctions liées à la bibliothèque graphique MLV. Nous pouvons y retrouver les fonctions créant la fenêtre, affichant le puzzle, le message de résolution, et le menu.

Dans notre programme, nous avons décider d'afficher des bandes noires entre les pièces, complexifiant un peu le calcul des positions. En effet, il ne suffit plus de simplement récupérer la position sur une image unie, mais d'y ajouter la taille et le nombre de bordures prises en compte.

Pour le menu, nous n'affichons que de simples miniatures des images. Comme elles sont par défaut de taille 512x512, il faut les redimensionner pour pouvoir afficher les 14 images sur une seule fenêtre.

Nous avions pour but de mettre en évidence quelle image est survolée. Les fonctions associées à cette option sont toutes fonctionnelles, mais elles requièrent beaucoup trop de ressources. La surbrillance n'apparaît pas immédiatement, il y a un léger délai, suffisamment grand pour que nous décidions de retirer cette fonctionnalité.

Le module contient toutes les constantes liées à l'affichage graphique, comme la taille de la fenêtre, des miniatures d'images, du puzzle, mais aussi la position du coin nord-ouest de la première miniature, pour faciliter le placement des autres.

## IV. Améliorations envisageables

De nombreuses améliorations peuvent être mise en place. Nous avons déjà la possibilité de laisser l'utilisateur choisir son image parmi une **banque d'image**, et d'afficher une **image référence** pour faciliter la résolution du puzzle.

L'amélioration affichant des numéros plutôt qu'une image avait été mise en place pendant la conception du tp. Nous l'avions utilisée comme un moyen de vérifier si notre mélange était fonctionnel, et l'avions finalement retirée une fois cette fonctionnalité terminée. Etant donné le fait qu'il faille modifier le menu si nous souhaitons ajouter un puzzle numéroté, nous avons pris la décision de ne pas réintégrer cette amélioration. Celle-ci reste néanmoins relativement simple à mettre en place.

La seconde possible amélioration simple à mettre en place est la tenue d'un score. Il suffit de chronométrer le temps pris par l'utilisateur pour résoudre le puzzle, appliquer une formule mathématique à ce temps, et écrire le résultat dans un fichier.

Pour le choix de la complexité, comme indiqué dans le sujet, cette amélioration nécessite une grosse refonte du projet. Elle prendrait beaucoup de temps à mettre en place, mais resterait néanmoins très intéressante à réaliser.

Une amélioration supplémentaire consistant à permettre à l'utilisateur de choisir s'il souhaite quitter le jeu ou relancer une partie peut être intéressante.

La réalisation d'un solveur automatique de puzzle est un défi très intéressant à relever, qui demanderait une bonne quantité de réflexion sur la technique à utiliser, ainsi que la façon de le mettre en place, tout en essayant de le rendre le moins coûteux possible.

Enfin, nous pouvons essayer de trouver un moyen d'améliorer les fonctions permettant de mettre en surbrillance les miniatures d'images dans le menu, afin de pouvoir mettre en place cette fonctionnalité, tout en gardant le programme fluide.

## V. Difficultés rencontrées

Lors de la réalisation du puzzle, la principale difficulté rencontrée était la mise en place du mélange. Sans compter les améliorations, cette étape a été la plus longue à réaliser. Cette difficulté était principalement due à la confusion entre la correspondance entre les abscisses et les ordonnées du plateau et de l'affichage graphique.

Mis à part cette fonctionnalité, la conversion des coordonnées de la souris était fastidieuse, étant donné l'affichage graphique que nous avions décidé de prendre. Il était nécessaire de prendre en compte les bandes noires du puzzle pour correctement récupérer la position du clic par rapport au plateau.

En prenant l'intégralité du tp, l'affichage du menu a été la partie la plus longue à mettre en place. De manière similaire à la position de souris sur le puzzle, il fallait récupérer correctement les coordonnées de la souris sur les miniatures, mais aussi les afficher correctement. L'ajout de surbrillance sur le survol des miniatures a été simple à mettre en place, mais rend malheureusement le programme trop lent.

## VI. Conclusion

Ce tp très amusant à réaliser nous ouvre à de nombreuses améliorations possibles. Il nous permet aussi de consolider nos acquis sur la gestion d'un tableau de types structurés, et comment l'exploiter efficacement, tout en perfectionnant d'autres points moins importants mais tout autant utile.