

Figure 1: ISIMA

Figure 2: Université de Clermont-Ferrand

Projet de Programmation avancée en C

Réalisation d'un jeu de Taquin



CLIQUEOT Théo et CHASSAGNOL Rémi
année 2020-2021

Professeurs: Vincent Limouzy
Référent TP: Nicolas Wagner

Travail à rendre pour le : 15/1/2021

Table des matières

| | | |
|----------|--|----------|
| 1 | Règle du jeux | 2 |
| 2 | Fonctionnement | 2 |
| 2.1 | Structure du taquin | 2 |
| 2.2 | Sauvegarde du taquin | 2 |
| 2.3 | Déplacement | 3 |
| 2.4 | création du taquin | 3 |
| 2.4.1 | Taquin aléatoire | 3 |
| 2.4.2 | Taquin charger | 3 |
| 2.5 | Taquin Trié | 3 |
| 3 | L’affichage SDL | 3 |
| 3.1 | L’affichage Principal | 3 |
| 3.2 | Les affichages secondaires | 3 |
| 3.2.1 | L’affichage de l’écran de départ | 3 |
| 3.2.2 | Le menu d’aide | 3 |
| 3.2.3 | La demande de sauvegarde | 4 |
| 3.2.4 | L’affichage de fin | 4 |
| 4 | Les contrôles claviers | 4 |
| 4.1 | Fonction principale | 4 |
| 4.2 | Fonction secondaire | 4 |
| 5 | Resolution | 4 |

Table des figures

| | | |
|---|--|---|
| 1 | ISIMA | 1 |
| 2 | Université de Clermont-Ferrand | 1 |

1 Règle du jeu

Ce qui va suivre va permettre à n'importe quelle personne de jouer au taquin. Cette section contient notamment tous les raccourcis clavier et toutes les fonctionnalités du jeu. Au lancement du jeu, si le joueur exécute le programme en donnant une grille (fichier texte) en paramètre, il sera affiché le menu de démarrage, il suffira alors d'appuyer sur une touche du clavier pour démarrer la partie. Si le joueur ne donne pas de grille, il lui sera d'abord demandé de saisir dans le terminal la taille de la grille à générer aléatoirement avant le lancement de l'affichage graphique.

- Déplacement : Flèches directionnelle ou les touches **h,j,k,l**.
- Espace entre les case ou non : touche **s**.
- Numéro sur les cases ou non : touche **n**.
- Charger un taquin : Pendant l'appel de l'exécutable donné en argument le nom du fichier.
- Taquin aléatoire : Lancé l'exécutable sans autre argument (la taille sera demandé via scanf).
- Tester si le taquin est correcte : touche **t**
- Solution au taquin : touche **r**
- echap pour l'aide.
- **q** pour quitter et sauvegarder si la grille n'est pas trillée.

Les mouvements sont définis par rapport à la case vide , c'est à dire que si on à :

| | | |
|---|--|----|
| 5 | | 10 |
|---|--|----|

 et qu'on appui sur **▷**, on obtiendra :

| | | |
|---|----|--|
| 5 | 10 | |
|---|----|--|

 Et inversement si on appui sur **◁**, on obtiendra :

| | | |
|--|---|----|
| | 5 | 10 |
|--|---|----|

2 Fonctionnement

2.1 Structure du taquin

On utilise la structure suivante (renommée en **taquin**). **tab** correspond à la position de nos pièces par rapport à la grille (un tableau 2D), l'entier **posCaseVide** correspond à la position de notre caseVide dans le taquin (par exemple si il est en bas à droite d'un taquin de dimension 5, il sera égale à 24), l'entier **taille** correspond à la dimension du taquin et enfin, on stocke le nombre de coup effectués par le joueur dans **nbrCoup**:

```

1  typedef struct taquin
2  {
3      int ** tab;
4      int posCaseVide;
5      int taille;
6      int nbrCoup;
7  }
8  taquin;
```

2.2 Sauvegarde du taquin

Cette fonction permet de sauvegarder la grille du joueur dans un fichier nommé **sauv_X.txt** où X est un nombre entier. Pour ce faire, la fonction ouvre puis ferme tous les fichier **sauv_i.txt** jusqu'à tombé sur un fichier inexistant (le nom n'a pas encore était attribué). Lorsque la fonction trouve le nom correcte, elle crée le fichier et écrit la grille dedans.

- Seul le tableau est sauvegardé puisqu'on peut retrouver les autres éléments du taquin à partir de celui-ci.
- Chaque case du tableau est séparée par un espace.
- Chaque ligne du tableau est séparée par un saut de ligne.

2.3 Déplacement

La fonction de déplacement prend en paramètre le taquin (un pointeur sur la structure) et un caractère constant qui correspond à la direction choisie. Elle permute la case vide (de valeur -1) avec la case qui correspond à la pièce que l'on déplace (qui se situe en haut, en bas, à gauche ou à droite). Si le mouvement n'existe pas (il n'y a pas de pièce à l'endroit sélectionné par le joueur), la fonction ne fait rien.

2.4 création du taquin

2.4.1 Taquin aléatoire

Pour créer un taquin aléatoire selon une taille donnée, il nous suffit de créer un taquin avec un tableau comme on le veut à l'arrivée et assigner à `posCaseVide` et `taille` la bonne valeur. Puis on applique un certain nombre de fois des déplacements aléatoires afin de mélanger le tableau. A noter que le fait de mélanger le tableau en faisant appel à la fonction de déplacement assure de la résolubilité de la grille.

2.4.2 Taquin charger

Il nous suffit de lire un fichier avec la même construction que notre fichier de sauvegarde et récupérer les valeurs de `tableau`, `taille` et `posCaseVide`

2.5 Taquin Trié

Il suffit de parcourir le tableau est de regarder si dans la case 0 on à 1, dans la case 1 on à 2 ... et pour la dernière -1. Pour cela, la fonction utilise une boucle **for** allant de 0 à la taille au carrée - 1 et test si la case de coordonnées (`i/taille`, `i%taille`) contient la valeur `i + 1` puis test si la dernière case du tableau contient -1. La fonction retourne 1 si la grille est valide, 0 sinon.

3 L'affichage SDL

Il y a en tout cinq fonctions d'affichage pour cinq écran différents. La première fonction va afficher le message de début de jeu (et le logo), la seconde va afficher un message qui demande à l'utilisateur si il souhaite sauvegarder sa progression dans la partie. Les trois autres vont afficher les écrans de jeu de fin et le menu d'aide.

3.1 L'affichage Principal

Cette fonction est la fonction d'affichage principale, elle va afficher les différentes parties de l'image en fonction de la grille du taquin. Chaque case du tableau correspond à une certaine partie de l'image en fonction de la valeur qu'elle contient. Par exemple, une case qui contient 1 correspond au coin en haut à gauche de l'image. La taille de la surface de l'image qui correspond à une case de tableau dépend de la taille de la grille car la taille de l'image est fixe.

Cette fonction utilise une seule surface (sans compter l'écran) qui contient l'image et deux **SDL_Rect**; une qui sert à découper l'image en plusieurs partie et l'autre qui contient la position de la partie de l'image coller sur l'écran.

3.2 Les affichages secondaires

3.2.1 L'affichage de l'écran de départ

Cette fonction permet d'afficher le petit texte qui apparaît lorsqu'on lance le programme. Elle affiche aussi le logo de DOOM en utilisant la fonction **SDL_SerColorKey** et l'option **SDL_SRCCOLORKEY** pour ne pas afficher le fond blanc du logo. Elle fait appel à la fonction **attend_touche** pour attendre que l'utilisateur appui sur une touche pour démarrer le jeu et donc changer l'affichage. Cette fonction retourne un entier pour que l'on puisse détecter quand l'utilisateur quitte la fenêtre et donc ne pas démarrer le jeu dans la fonction main.

3.2.2 Le menu d'aide

Cette fonction est appelée quand l'utilisateur appuis sur la touche **escape** pendant le jeu et affiche un texte qui donne les différents contrôles clavier au joueur.

3.2.3 La demande de sauvegarde

Cette fonction est appelée quand l'utilisateur quitte la fenêtre (en appuyant sur **q**) sans avoir fini le jeu. Elle affiche un texte expliquant que l'utilisateur peut appuyer sur **y** pour sauvegarder sa partie, toute autre touche fermera le programme sans sauvegarde. La boucle attendant la réponse de l'utilisateur se trouve dans la fonction.

3.2.4 L'affichage de fin

Cette fonction est appelée quand l'utilisateur a complété le défi et a appuyé sur **t** ou **q**. Elle affiche "victoire" et aussi le nombre de coup que le joueur a mis pour terminer le jeu à l'aide de **SDL_TTF**. Elle fait appel à la fonction **attend_touche** qui permet d'attendre que le joueur appuie sur une touche pour fermer le programme.

4 Les contrôles claviers

Il y a deux fonctions qui permettent de détecter quand l'utilisateur appuie sur des touches du clavier. La fonction **attend_touche** qui est située dans `affichage.c` et la fonction **recup_touche** située dans le `main.c`.

4.1 Fonction principale

Cette fonction permet de récupérer toutes les entrées clavier qui correspondent à celles affichées par le menu d'aide pendant la partie (déplacements, ...). Elle utilise la fonction **Wait_Event** de la library `sdl` qui attend une action de l'utilisateur puis qui enregistre cette action dans la variable **evenement**. Ensuite, on utilise **switch** imbriqués, le premier effectue des actions en fonction du type d'événement (ici seul **SDL_QUIT** et **SDL_KEYDOWN** sont utilisés). Le second **switch** intervient lorsque l'événement est de type **touche pressée** et va donc effectuer les actions des différents contrôles clavier quand la touche pressée correspond. Cette fonction retourne 0 quand l'utilisateur fait l'action de fermer la fenêtre (ou appuie sur **q**) pour pouvoir stopper le programme dans le `main`.

4.2 Fonction secondaire

Cette fonction attend que l'utilisateur ferme la fenêtre (et retourne 0 lorsque c'est le cas) ou appuie sur une touche du clavier, elle permet de maintenir un affichage tant que l'utilisateur ne fait rien. Elle utilise une boucle **while** pour éviter d'être stoppée si **Wait_Event** enregistre un événement de type différent de **SDL_QUIT** ou **SDL_KEYDOWN**.

5 Résolution

La résolution consiste à réarranger la première ligne, puis la seconde ... jusqu'à ce qu'il nous reste les 2 dernières lignes. Ensuite on va réarranger les colonnes de taille 2 jusqu'à ce qu'il nous reste qu'un carré de quatre cases. Ensuite il nous reste plus qu'à le tourner jusqu'à obtenir la bonne rotation.

Pour la première étape. On va à chaque fois positionner les cases à leur place respective sauf pour les cases des deux dernières colonnes. ou on va utiliser une méthode pour mettre les 2 en même temps sans bloquer la case vide après.

Pour la seconde étape même principe on va utiliser une méthode pour remplir les 2 cases d'une même colonne

Le principe consiste à mettre la case de l'avant dernière colonne (ligne) à la place de la case de la dernière ligne (colonne). Puis de mettre celle de la dernière colonne (ligne) en bas (à gauche) de sa position. puis on fait une boucle.