



ECOLE NATIONALE SUPÉRIEURE D'INFORMATIQUE ET D'ANALYSE DES
SYSTÈMES - RABAT

Rapport de Projet de programmation : Jeux de mémoire : Animaux pour enfant

Réalisé par :

Amine TITROFINE
Aymane SAID

Encadré par :

Pr. Sanaa EL FKIHI



Remerciements :

Au terme de ce travail, nous profitons de l'occasion pour remercier du fond du cœur tous ceux qui ont participé de près ou de loin à la réalisation de ce projet, nous adressons donc, en particulier nos sincères remerciements à notre encadrant Mme EL FKIHI Sanaa pour sa disponibilité et ses précieux conseils, et n'oublions pas notre cher professeur Mr EL FAKER Abdellatif pour les précieux informations et outils acquis durant les séances du cours de « Structures de données ».

En espérant que ce travail soit à la hauteur de vos attentes. Cordialement.



Résumé

Le travail présenté dans ce projet porte sur la réalisation d'un jeu de mémoire en utilisant le langage c. C'est un jeu de société qui consiste à étaler aléatoirement des cartes, et ensuite les retournées face cachée, et au joueur de trouver les paires identiques.

Abstract

The goal of this project is to create a Memory Matching game using the C programming language . This game is a board game that requires players to match similar elements.

Table des matières

Introduction	2
1 Présentation du projet	3
1.1 Sujet	3
1.2 Principe du jeu	3
2 Analyse et conception	4
2.1 Cahier de charge	4
2.1.1 Conception du jeu	4
2.1.2 But de jeu	4
2.1.3 Interface graphique	4
2.1.4 Sauvegarde de la partie	4
2.1.5 Gestion du temps	4
2.2 Méthode de résolution	5
2.2.1 Conception de la grille	5
2.2.2 Condition d'arrêt	5
2.2.3 Sauvegarde de la partie	6
3 Réalisation du projet	7
3.1 Outils utilisés	7
3.1.1 Code : :Blocks	7
3.1.2 Interface graphique SDL	7
3.1.3 Interface graphique GTK+	8
3.1.4 Adobe Illustrator	8
3.1.5 Rédaction du rapport	9
3.2 Fonctions du code source	9
3.2.1 Les directives des préprocesseurs	9
3.2.2 Définition des constantes	9
3.2.3 Les structures	9
3.2.3.1 Structure <i>JOUEUR</i>	9
3.2.3.2 Structure <i>UTILISATEUR</i>	10
3.2.4 Fonction <i>MODEJEU</i>	10
3.2.5 Fonction <i>MENUDEJEU</i>	10
3.2.6 Fonction <i>MEMORISER</i>	10
3.2.7 Fonction <i>AFFICHEGRILLE</i>	10
3.2.8 Fonction <i>NOMBREDEJOUEUR</i>	11
3.2.9 Fonction <i>CREERJOUEUR</i>	11
3.2.10 Fonction <i>FOUND</i>	11
3.2.11 Fonction <i>CHECKADMIN</i>	11
3.2.12 Fonction <i>EXCLUREJOUEUR</i>	11
3.2.13 Fonction <i>MODIFIERSCORE</i>	12
3.2.14 Fonction <i>CODEJEU</i>	12
3.2.15 Fonction <i>INITMAT</i>	12
3.2.16 Fonction <i>REEMPLIRMAT</i>	12
3.2.17 Fonction <i>FINDEJEU</i>	12
3.2.18 Fonction <i>TIMER</i>	12
3.2.19 Fonction <i>RECOMMENCER</i>	13
3.2.20 Fonction <i>QUIT</i>	13
3.2.21 Fonction <i>main</i>	13

4 Interface graphique	14
4.1 Icône du jeu	14
4.2 Premier Affichage	14
4.3 Menu principale	15
4.3.1 Bouton Se connecter	15
4.3.2 Bouton S'inscrire	16
4.4 A propos	17
4.4.1 Bouton <i>ADMIN</i>	17
4.4.2 Bouton <i>INSTRUCTIONS</i>	18
4.4.3 Bouton <i>Quittter?</i>	18
4.5 Mode de jeu	19
4.5.1 Facile	19
4.5.2 Moyen	20
4.5.3 Difficile	20
4.6 Gagnant	21
Conclusion et perspectives	22

Introduction générale

Les Memory et les jeux de cartes sont de bons moyens d'améliorer les compétences de la mémoire visuelle. Le traitement de l'information visuelle implique de regarder des images et de se souvenir de certains détails et aspects qui aident à les mémoriser. Les jeux qui encouragent la formation de notre mémoire visuelle aident également à la concentration, en accordant une attention aux détails et en classant l'information selon leurs attributs semblables ou différents. Le but de ce projet est de créer un tel jeu. Et pour se faire, il est nécessaire de positionner le problème et de se focaliser dans un premier temps sur la partie modélisation du projet. Il est indispensable dans ce contexte d'implémenter des algorithmes qui permettent de générer des chiffres aléatoirement et les associer aux images étaillées dans la grille de jeu . Le choix de l'interface a été un facteur important dans le projet ainsi l'utilisation des différentes structures étudiées sans oublier la notion de modularité. Tout ceci sera développé dans la première partie. La réalisation de ce projet demandera plusieurs outils. Ceci sera expliqué et illustré dans la deuxième partie ainsi que le fonctionnement de jeu.

Chapitre 1

Présentation du projet

PROJET C s'agit de produire un programme d'environ 500 lignes de code afin de valider les compétences des cours : « Algorithmique », « Technique de programmation » et « Structures de données ». Le programme correspond à 20 heures de travail effectives en langage C. Les étudiants travaillent en binôme et bénéficient des conseils d'un professeur encadrant¹

1.1 Sujet

Jeu de Memory est un jeu de société édité par Ravensburger pour la première fois en 1959 et décliné depuis en de multiples versions.



FIGURE 1.1 – Jeu de mémoire

1.2 Principe du jeu

Le jeu se compose de paires de cartes portant des illustrations identiques. L'ensemble des cartes est mélangé, puis étalé face contre table. À son tour, chaque joueur retourne deux cartes de son choix. S'il découvre deux cartes identiques, il les ramasse et les conserve, ce qui lui permet de rejouer. Si les cartes ne sont pas identiques, il les retourne faces cachées à leur emplacement de départ.

Le jeu se termine quand toutes les paires de cartes ont été découvertes et ramassées. Le gagnant est le joueur qui possède le plus de paires.

1. Notice du projet de programmation 2020/2021

Chapitre 2

Analyse et conception

Ce chapitre est consacré à la présentation de la partie d'analyse et conception en présentant premièrement le cahier de charge et le modèle de résolution de problème .

2.1 Cahier de charge

Le cahier de charges présente l'ensemble des instructions et contraintes qui cadrent la réalisation du jeu. Le cahier de charges disponible donne les instructions qu'on va élaborer dans cette partie.

2.1.1 Conception du jeu

Le jeu devra générer des parties entre un ou plusieurs joueurs humains(on ne demande pas au programme de pouvoir simuler un joueur). Le programme doit étaler les cartes d'une façon aléatoire et laisser du temps aux joueurs pour les mémoriser. Ensuite, les cartes sont tournées et aux joueurs d'identifier les paires identiques.

2.1.2 But de jeu

Le but du jeu est pour chaque joueur de pouvoir mémoriser l'emplacement de la majorité des cartes. Condition d'arrêt : La partie s'arrête si toutes les cartes étalées sont tournées. Le joueur ayant mémoriser plus de cartes gagne.

2.1.3 Interface graphique

Afin de fournir une interface convivable et simple pour tout type d'utilisateur nous avons décidé de travailler avec les deux bibliothèques SDL2 et GTK+. Le choix de GTK+ était pour l'interface de connexion et d'inscription puisque la bibliothèque permet d'implémenter des « textbox » d'une façon très simple au contraire de SDL2. Le reste des interfaces est programmé avec SDL2 qui offre un contrôle complet du jeu et des animations. En somme, le programme est basé sur SDL2 et GTK+ n'est qu'une extension pour gérer l'interface de connexion et d'inscription.

2.1.4 Sauvegarde de la partie

On devra pouvoir mémoriser et sauvegarder l'ensemble des informations sur les joueurs. ainsi qu'enregistrer les scores .

2.1.5 Gestion du temps

A fin d'établir un équilibre entre les tours . Chaque tour doit être géré par un chronomètre.

2.2 Méthode de résolution

Dans cette partie, nous allons discuter quelques méthodes de résolution pour quelques instructions proposé dans le cahier de charges.

2.2.1 Conception de la grille

La grille du jeu peut être considérer comme un tableau de 2 dimensions ($a*b$) ;les dimensions varient selon le niveau de difficulté. Ce tableau est remplis par des nombres aléatoire (allant de 1 jusqu'à $a*b/2$).

Rand1	Rand2	Rand3	Rand4	Rand5	Rand6
Rand7	Rand8	Rand9	Rand10	Rand11	Rand12
Rand13	Rand14	Rand15	Rand16	Rand17	Rand18
Rand19	Rand20	Rand21	Rand22	Rand23	Rand24
Rand25	Rand26	Rand27	Rand28	Rand29	Rand30

FIGURE 2.1 – Exemple proposé de la grille de jeu

2.2.2 Condition d'arrêt

Condition d'arrêt : La partie s'arrête également si toutes les cartes sont tournées .
Pour vérifier cette condition, il faut établir un programme similaire au algorithme¹ suivant :

Algorithm 1 Etablir la fin de la partie

```
gagne ← 0;  
while gagne ≠ 1 do  
    retour ← 0;  
end while  
for i ← 0 to i ← a do  
    for j ← 0 to j ← b do  
        if retournmat[i][j] = 2 then  
            retour ← retour + 1;  
        end if  
    end for  
end for  
if retour = a * b then  
    gagne ← 1;  
end if
```

2.2.3 Sauvegarde de la partie

Afin de pouvoir sauvegarder les identifiants des joueurs et leurs scores , il faut faire usage au méthodes de stockages des fichiers. En effet, on propose la sauvegarde de ces informations dans un fichier texte . On propose également donner l'accès aux administrateurs s'ils veulent supprimer un joueur .

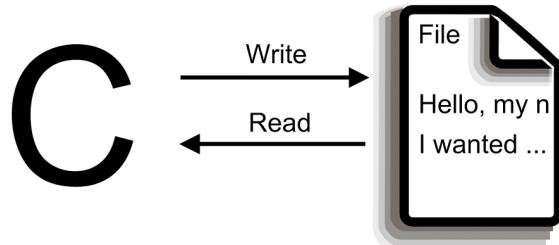


FIGURE 2.2 – Lecture et Ecriture dans les fichiers en C

Chapitre 3

Réalisation du projet

Dans ce chapitre nous allons citer les outils utilisés dans la réalisation du projet. Ainsi , nous allons expliquer le rôle des fonctions figurant dans le code source.

3.1 Outils utilisés

Cette partie détaille la démarche de réalisation du projet et les problèmes qui l'accompagnaient.

3.1.1 Code : :Blocks

Pour la réalisation du travail, nous avons utilisé le langage C dans l'IDE Code : :Blocks,Code : :Blocks est un environnement de développement intégré libre et multiplateforme. Il est écrit en C++ et utilise la bibliothèque wxWidgets. Code : :Blocks est orienté C et C++, mais il supporte d'autres langages.



FIGURE 3.1 – Logo de l'IDE Code : :Blocks

3.1.2 Interface graphique SDL

Simple DirectMedia Layer est une bibliothèque logicielle libre. Son API est utilisée pour créer des applications multimédias en deux dimensions pouvant comprendre du son comme les jeux vidéo, les démos graphiques, les émulateurs, etc.



FIGURE 3.2 – Logo de la librairie SDL

3.1.3 Interface graphique GTK+

GTK+ est un ensemble de bibliothèques logicielles, c'est-à-dire un ensemble de fonctions permettant de réaliser des interfaces graphiques. Cette bibliothèque a été développée originellement pour les besoins du logiciel de traitement d'images GIMP.



FIGURE 3.3 – Logo de la librairie GTK+

3.1.4 Adobe Illustrator

Adobe Illustrator est un logiciel de création graphique vectorielle. Il fait partie de la gamme Adobe, peut être utilisé indépendamment ou en complément de Photoshop, et offre des outils de dessin vectoriel puissants. Les images vectorielles sont constituées de courbes générées par des formules mathématiques.



FIGURE 3.4 – Logo de Adobe Illustrator

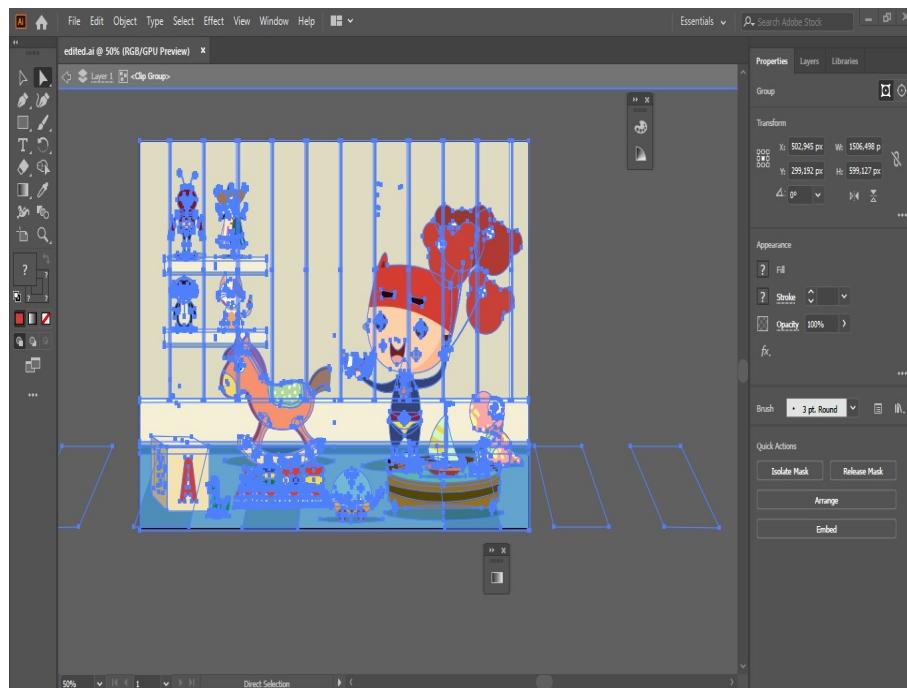


FIGURE 3.5 – capture de la réalisation de l'arrière-plan

3.1.5 Rédaction du rapport

La documentation professionnelle nécessite la manipulation du logiciel de traitement de texte LaTex. Travailler avec ce dernier est inévitable tôt ou tard, donc nous avons voulu exploiter cette opportunité et explorer LaTex.



FIGURE 3.6 – Logo du logiciel LaTex

3.2 Fonctions du code source

Comme vu précédemment dans le premier chapitre, chaque instruction nécessite une ou plusieurs fonctions. Dans cette partie nous allons élaborer chacune des fonctions utilisés dans le code source.

3.2.1 Les directives des préprocesseurs

```
1 #define LIBRARIES_H_INCLUDED
2 #include<stdio.h>
3 #include<stdlib.h>
4 #include<gtk/gtk.h>
5 #include<SDL2/SDL.h>
6 #include<time.h>
7 #include<string.h>
8 #include<windows.h>
9 #include "header.h"
10 #include "affichage.h"
11 #include "players.h"
12 #include "apropos.h"
13 #include "instruction.h"
14 #include "admin.h"
15 #include "menudejeu.h"
16 #include "modedejeu.h"
17 #include "codedejeu.h"
18 #include "findejeu.h"
19 #include "finedejeu.h"
```

Ces directives ont comme rôle l'importation les bibliothèques et les fichiers *header* contenant les fonctions à utiliser dans notre application.

3.2.2 Définition des constantes

```
1 #define WINDOW_WIDTH 800
2 #define WINDOW_HEIGHT 640
```

Ces constantes définissent les dimensions de l'interface du jeu.

3.2.3 Les structures

3.2.3.1 Structure *JOUEUR*

Pour bien gérer les informations sur les joueurs on a défini la structure suivante.

```
1 typedef struct __player{
2     char name[20] ;
3     char username[20] ;
4     char password[20] ;
5     int score ;
6 }player;
7 typedef struct __player* Player;
```

3.2.3.2 Structure UTILISATEUR

```
1 typedef struct _utilisateur {
2     int score=0;
3     char *nom;
4 } utilisateur;
```

Cette structure sert à stocker les informations importées à partir du fichier.

3.2.4 Fonction MODEJEU

```
1 void modejeu(SDL_Window *ecran ,SDL_Surface *surface ,int joueurs ,char* names []);
```

Cette fonction permet de déterminer le niveau de difficulté du jeu .

Input : nombre qui représente le niveau de difficulté(1 , 2 ou 3)

Output : (void)

3.2.5 Fonction MENUDEJEU

```
1 void menudejeu(SDL_Window *ecran ,SDL_Surface *surface ,int joueurs);
```

Cette fonction a comme rôle l'affichage du menu de jeu.

3.2.6 Fonction MEMORISER

```
1 void Memoriser (SDL_Window*ecran ,SDL_Surface *image ,int **retournemat , int **mat ,int a,int b)
;
```

Cette fonction affiche selon le niveau de difficulté les cartes tournées face pour les mémoriser.

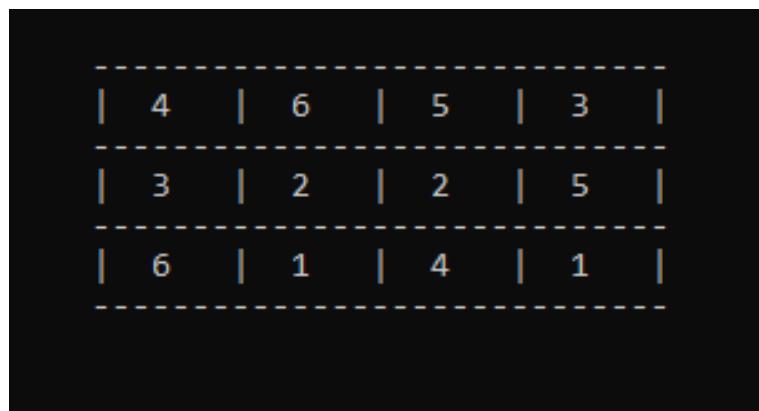


FIGURE 3.7 – Output de la fonction MEMORISER dans la console

3.2.7 Fonction AFFICHEGRILLE

```
1 void affichegrille (SDL_Window*ecran ,SDL_Surface *image ,int **retournemat , int **mat ,int a ,
int b);
```

Cette fonction permet d'afficher la grille du jeu.

```

| ? | ? | ? | ? | ? | ? |
| ? | ? | ? | ? | ? | ? |
| ? | ? | ? | ? | ? | ? |
-----
| ? | ? | ? | ? | ? | ? |
-----
| ? | ? | ? | ? | ? | ? |

score de player1 est : 0
score de player2 est : 0

Choisissez la premiere carte svp.
Numero de la ligne (de 1 a 5) :

```

FIGURE 3.8 – Output de la fonction *AFFICHEGRILLE* dans la console

3.2.8 Fonction *NOMBREDEJOUEUR*

```
1 void nombreDeJoueur(SDL_Window *ecran ,SDL_Surface *surface);
```

Cette fonction permet de choisir le nombre de joueur.

3.2.9 Fonction *CREERJOUEUR*

```
1 void creerJoueur(GtkWidget *widget ,gpointer data);
```

Cette fonction permet de créer un joueur et stocke ses informations dans un fichier texte suivant à son inscription.

3.2.10 Fonction *FOUND*

```
1 player* found(char *nomdefichier , char * login , char * password)
```

Cette fonction permet de vérifier si le joueur existe déjà dans le fichier.

3.2.11 Fonction *CHECKADMIN*

```
1 void checkAdmin(GtkWidget *widget ,gpointer data);
```

Cette fonction vérifie si vraiment un admin qui essaie d'accéder aux paramètres de jeu.

3.2.12 Fonction *EXCLUREJOUEUR*

```
1 void excludeJoueur(GtkWidget *widget ,gpointer data);
```

Cette fonction donne le droit aux admins de supprimer un joueur.

3.2.13 Fonction *MODIFIERSCORE*

```
1 void modifierscore(player *ptr,int score);
```

Cette fonction permet de mettre à jour le score dans le fichier ,et stocker le meilleur score qu'un joueur a pu obtenir.

3.2.14 Fonction *CODEJEU*

```
1 void codejeu(SDL_Window*ecran,SDL_Surface *surface,SDL_Event event,int a,int b,int joueurs,  
char*names[]);
```

Cette fonction a comme rôle l'appel des principales fonctions dans notre code ,elle permet donc de générer des parties à partir des outputs des fonctions *MODEJEU* , *NOMBREDEJOUEUR* .

3.2.15 Fonction *INITMAT*

```
1 void initmat(int **mat,int **retournemat,int a,int b);
```

Cette fonction permet d'initialiser les valeurs d'une matrice de 2 dimensions à partir de (Output de la fonction *MODEJEU*).

3.2.16 Fonction *REEMPLIRMAT*

```
1 void remplirmat(int val[],int **mat,int a,int b);
```

Cette fonction permet de remplacer les valeurs initialisées par *INITMAT* par des valeurs aléatoires entre (1 et $(a*b)/2$) .

3.2.17 Fonction *FINDEJEU*

```
1 void findejeu(char *nom,int score);
```

Cette fonction permet d'afficher le joueur qui a gagné la partie, ainsi que son score , Et d'enregistrer dans le cas de la réalisation d'un meilleur score dans un fichier texte.

3.2.18 Fonction *TIMER*

```
1 void timer(SDL_Surface *surface,int variable);
```

Cette fonction permet de gérer les tours entre les joueurs par un chronomètre.

3.2.19 Fonction *RECOMMENCER*

```
1 void recommencer(GtkWidget *widget , gpointer data);
```

Cette fonction donne la possibilité de recommencer une autre partie .

3.2.20 Fonction *QUIT*

```
1 void quit(GtkWidget *widget , gpointer data);
```

Cette fonction nous permet de quitter le jeu au début ou à la fin d'une partie.

3.2.21 Fonction *main*

```
1 #include "header.h"
2 #include<SDL2/SDL_audio.h>
3
4 int main(int argc ,char* argv []){
5     gtk_init(&argc,&argv); // permet d'initialiser GTK
6
7     srand(time(NULL)); // nécessaire pour l'appel de la fonction rand()
8     SDL_Window *ecran=NULL; // déclaration d'une variable de type ecran
9     SDL_Surface *surface=NULL ; // déclaration d'une variable de type surface
10
11    putenv( "SDL_VIDEO_CENTERED=1" ); //ON CENTRE LA FENETRE
12
13    ecran = SDL_CreateWindow( "JeuDeMemory" ,SDL_WINDOWPOS_CENTERED,SDL_WINDOWPOS_CENTERED,
14                             LARGEUR_FENETRE,HAUTEUR_FENETRE,0 );
14    surface= SDL_GetWindowSurface( ecran );
15
16    nombreDeJoueur(ecran ,surface); // La première fonction nous permet de choisir le nombre de
17                                // joueurs
17
18    SDL_FreeSurface(surface);
19    SDL_DestroyWindow(ecran);
20    SDL_Quit();
21    return EXIT_SUCCESS;
22 }
```

Cette fonction est le main de notre programme.

Chapitre 4

Interface graphique

Dans ce chapitre, nous allons valider les attentes du projet à travers une interface graphique en utilisant les bibliothèque SDL2 et GTK+.

4.1 Icône du jeu

Nous retrouvons ici l'icône symbolisant notre jeu.



FIGURE 4.1 – Icône du jeu

4.2 Premier Affichage

Nous retrouvons ici la capture du premier affichage qui permet aux joueurs de choisir le mode de jeu (individuel ou en groupe).



FIGURE 4.2 – Premier affichage lors de l'exécution du jeu

4.3 Menu principale

Après avoir choisir le nombre de joueur, maintenant c'est l'étape de s'identifier ou bien s'inscrire dans le jeu.



FIGURE 4.3 – Menu des choix

4.3.1 Bouton Se connecter



FIGURE 4.4 – Bouton Se connecter

Il déclenche une fenêtre où le joueur doit insérer son nom d'utilisateur et mot de passe ,ensuite il fait appel à la fonction found pour vérifier si le joueur existe ou pas.



FIGURE 4.5 – Fenêtre de connexion

4.3.2 Bouton S'inscrire



FIGURE 4.6 – Bouton S'inscrire

Ce bouton concerne les nouveaux joueurs pour faire l'inscription.

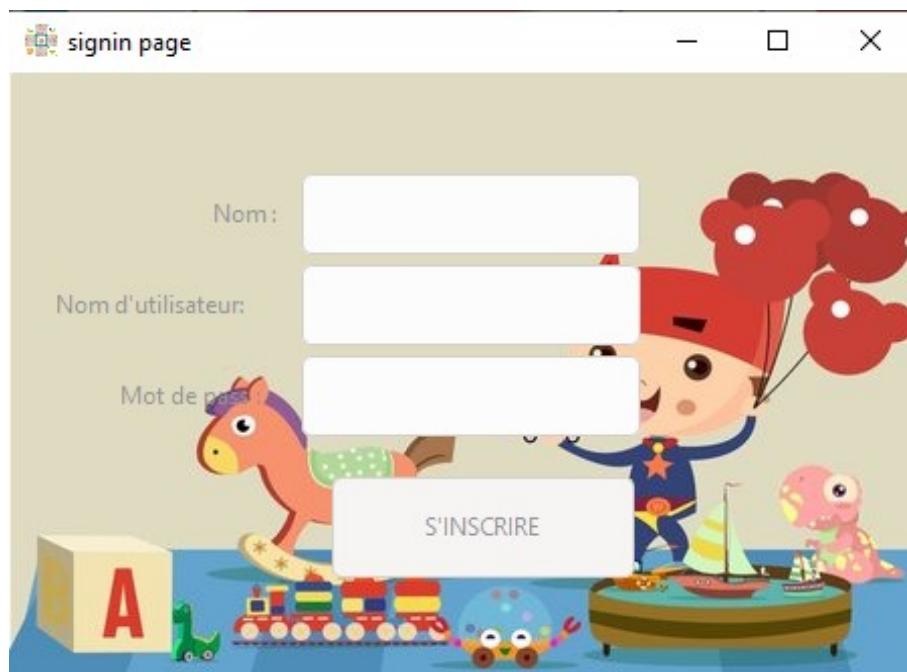


FIGURE 4.7 – Fenêtre de connexion

4.4 A propos



FIGURE 4.8 – fenêtre A propos

4.4.1 Bouton ADMIN

Ce bouton est consacré seulement à l'admin

A close-up view of a red rectangular button with the word 'Admin' written in white, bold, sans-serif font, centered within the button.

FIGURE 4.9 – Bouton Admin



FIGURE 4.10 – Fenêtre de connexion

4.4.2 Bouton *INSTRUCTIONS*

Instructions

FIGURE 4.11 – Bouton Instruction

ce bouton permet d'ouvrir le fenêtre qui contient les instructions du jeu.



FIGURE 4.12 – Interfaces d'instructions

Elle permet d'expliquer le déroulement du jeu

4.4.3 Bouton *Quittter?*

QUITTER

FIGURE 4.13 – Bouton quitter

4.5 Mode de jeu

Cette fenêtre nous permet de choisir le niveau de difficulté du jeu.



FIGURE 4.14 – Modes de jeu

4.5.1 Facile



FIGURE 4.15 – Modes de jeu

4.5.2 Moyen



FIGURE 4.16 – Modes de jeu

4.5.3 Difficile



FIGURE 4.17 – Grille de jeu en mode difficile

4.6 Gagnant

A fin de chaque partie le gagnant est mentionné dans cette fenêtre.

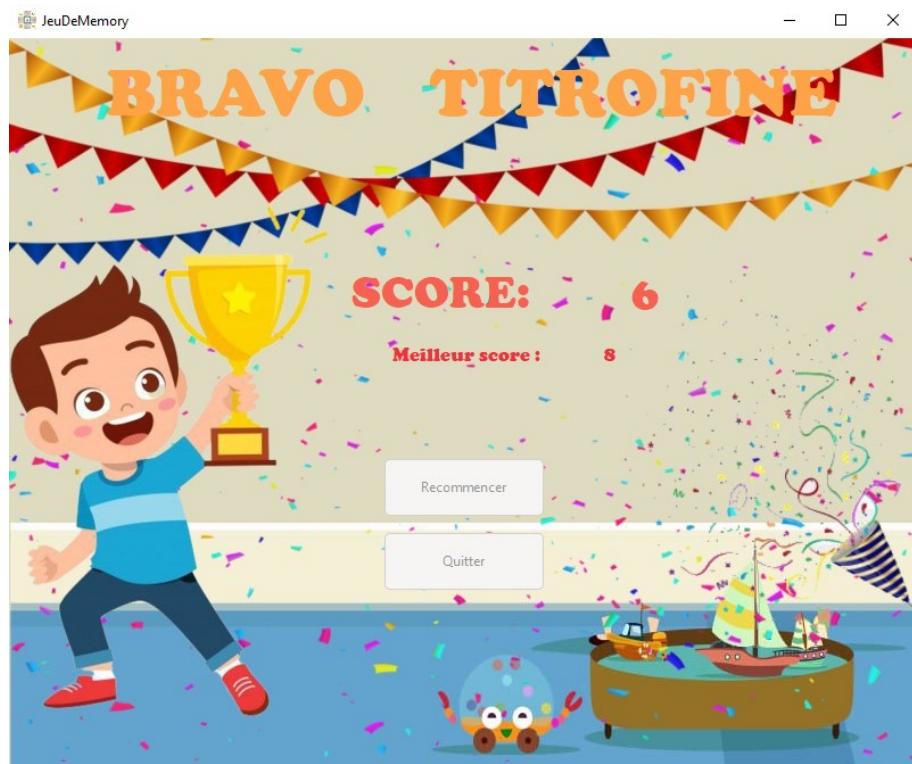


FIGURE 4.18 – Gagnant

Conclusion et perspectives

Le travail demandé a été de réaliser un jeu de mémoire . Nous avons respecté toutes les fonctionnalités exigées par le cahier de charge. Le menu de jeu offre plusieurs options à l'utilisateur à savoir se connecter, s'inscrire, lire les instructions ... Le jeu se joue entre un ou plusieurs joueurs , et chaque tour est réalisé par un chronomètre . A fin de chaque partie le programme actualise les scores. et pour organiser le jeu un admin a le pouvoir d'éliminer quelques joueurs.

Finalement, on peut ajouter un programme qui suggère le niveau en se basant sur le score marqué précédemment.Comme nous pouvons aussi chercher à améliorer nos algorithmes en diminuant leurs complexités.

Bibliographie

- [1] Cours de structures de données – Mr EL FAKER.
- [2] Cours de techniques de programmation – Mr GUERMAH.
- [3] <https://stackoverflow.com/>.
- [4] <https://wiki.libsdl.org/>, 2020. [Online ; accessed 10-December-2020].
- [5] <https://developer.gnome.org/gtk3/>, 2021. [Online ; accessed 07-January-2021].
- [6] <<https://www.overleaf.com/project>>, 2021. [Online ; accessed 26-January-2021].