

REPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITÉ DES SCIENCES ET DE LA TECHNOLOGIE HOUARI BOUMEDIEN



FACULTÉ D'INFORMATIQUE

SPÉCIALITÉ : L2 ACAD C

RÉALISÉ PAR :

NOM :	PRÉNOM :	MATRICULE :
BOULOUH	ABDERRAHMENE	222231649908
BOURABIA	CHEMS EDDINE	222231622113
HAMIDAT	MOHAMED ABDALLAH	212132025529
MERROUKI	MOHAMED EL MAHDI YACINE	222232281314
MERAKCHI	RAHMA	222231657517
OUABEL	LYNA	222231660007
IGUER	LINA	222231354418
GHEBRIOU	KHEIR EDDINE	212131046908

MINDMAZE

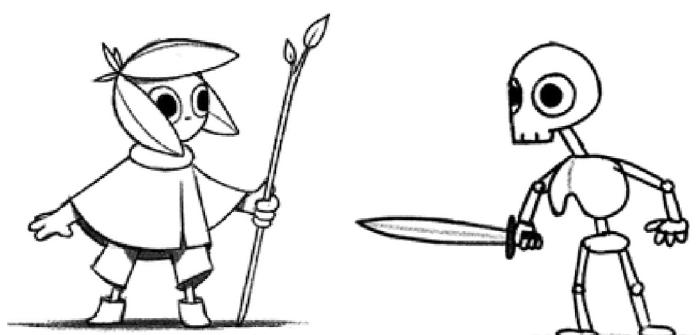
ALEX VS NEXUS

REPORT



SOMMAIRE

• <u>INTRODUCTION/GAME STORY</u>	1
• <u>TECHNOLOGIES UTILISÉES/IMPLICATION</u>	2
• <u>CONCEPTION DU JEU /GRAPHISME</u>	3
• <u>MODE PLAYER</u>	4
• <u>EVALUATION DU NIVEAU/SCORE</u>	5
• <u>MODE MACHINE</u>	6
• <u>GUIDE DU JEU</u>	7
• <u>CONCLUSION</u>	8
• <u>BIBLIOGRAPHIE</u>	9



INTRODUCTION

“ MINDMAZE” propose une fusion captivante entre la mémorisation et la prévision des trajectoires, développée avec la puissante combinaison de **SDL** (**Simple DirectMedia Layer**) et le langage de programmation **C**.

Avec **deux modes** distincts, les joueurs mémorisent d'abord l'orientation des diagonales avant de prédire la trajectoire d'une bille, qui change à chaque contact avec une diagonale. Cette expérience unique **teste la mémoire visuelle** et **la prise de décision en temps réel**, offrant un défi stimulant.

Notre rapport **explore en détail** chaque aspect de ce jeu, de sa mécanique intrigante aux stratégies émergentes.

GAME STORY

Dans un futur dystopique où règne “**The Nexus**”, un réseau de **machines** développés, l'humanité est au bord de l'extinction. **Alex**, le dernier survivant, fait une découverte cruciale en mettant au jour le “**Core Nexus**”, le **centre du contrôle** exercé par The Nexus.

Selon les légendes, **un code dormant** pourrait perturber ce contrôle oppressant.

La mission d'Alex évolue ainsi de la simple survie à **une infiltration** du Core Nexus.

Les récits transmettent l'idée qu'un code inactif pourrait offrir **la libération à l'humanité**.

Cependant, l'entrée est protégée par **des énigmes complexes**, et le fait de les pirater **affaiblit les défenses du Nexus**.

En cas d'échec, **une poursuite implacable** est déclenchée.



TECHNOLOGIES UTILISÉES

Le développement de notre jeu s'est appuyé sur la bibliothèque Simple DirectMedia Layer (SDL2). La SDL2 offre un ensemble complet de fonctionnalités pour le développement de jeux en C et a constitué la base de notre architecture.

En plus de la SDL2, nous avons intégré plusieurs bibliothèques supplémentaires pour enrichir les fonctionnalités de notre jeu. La bibliothèque **SDL_image** a été utilisée pour le chargement et l'affichage des images, tandis que **SDL_ttf** a été employée pour la gestion des polices de caractères et l'affichage du texte. Pour l'aspect sonore, **SDL_mixer** a été inclus pour intégrer des effets sonores et de la musique d'ambiance.

L'utilisation de ces bibliothèques complémentaires a permis d'étendre les capacités de notre jeu, offrant une expérience plus immersive aux joueurs.

IMPLEMENTATION

L'implémentation du jeu a été réalisée en suivant une approche qui garantit une structure claire, comme la gestion des niveaux et parties, la gestion des scores ...

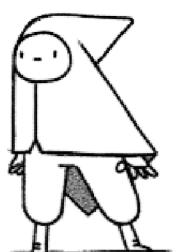
Le cœur du jeu repose sur la boucle principale (loop game), où les événements SDL sont capturés et traités pour mettre à jour l'état du jeu en conséquence.

Le chargement et l'affichage des graphismes sont gérés à l'aide de la bibliothèque **SDL_image**, permettant une intégration fluide des icônes et des arrière-plans.

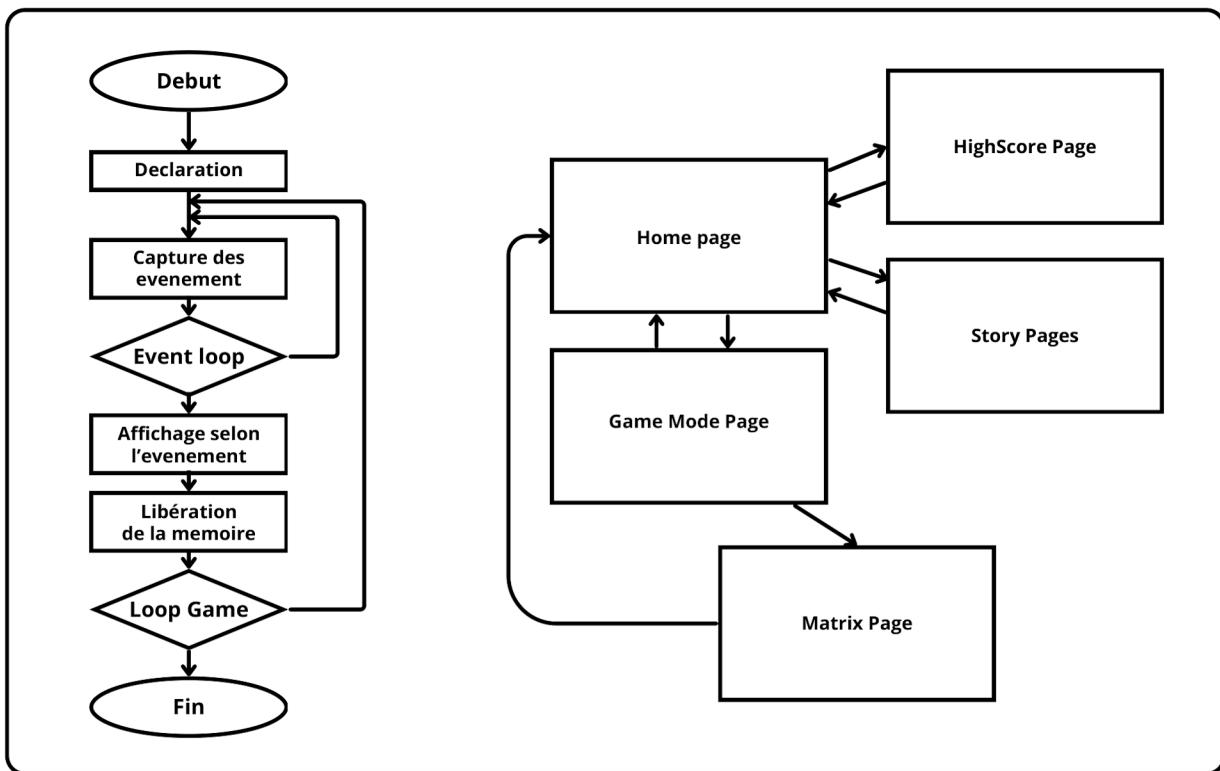
En outre, la gestion du son et de la musique a été intégrée à l'aide de la bibliothèque **SDL_mixer**, offrant une expérience audio immersive.

Les fonctionnalités de sauvegarde et de chargement ont également été mises en place pour permettre aux joueurs de reprendre leur progression ultérieurement.

Dans cette phase d'implémentation, des tests approfondis ont été effectués tout au long de la phase pour garantir la stabilité du jeu, ainsi à un produit final fonctionnel, robuste et conforme aux objectifs fixés au début du projet.



Conception du jeu



GRAPHISMES

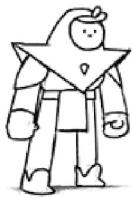
Dans cette section, nous explorons l'utilisation de la bibliothèque [SDL_image](#) pour la gestion des graphismes dans notre jeu.

Le chargement des images se fait de manière efficace, en utilisant les fonctionnalités de [SDL_image](#) pour supporter différents formats d'image. Nous avons mis en place une gestion optimisée des sprites, permettant une animation fluide et réactive du jeu.

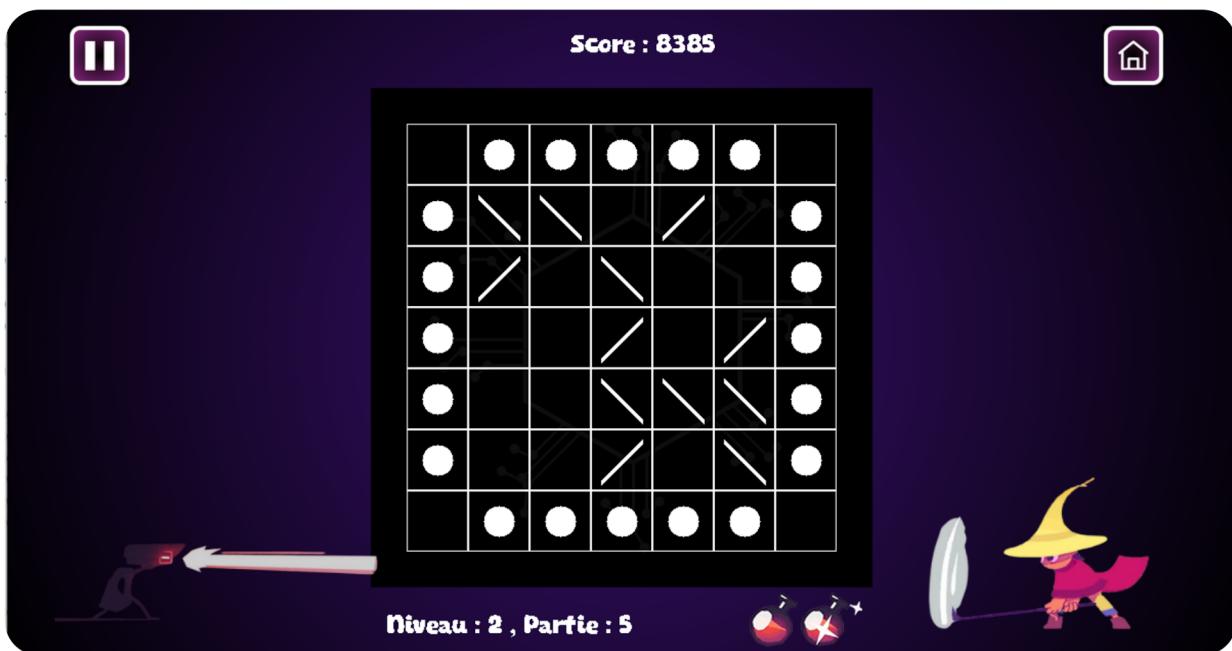
Au cours de l'implémentation des graphismes, nous avons également pris en compte les performances du jeu. Des techniques d'optimisation ont été utilisées pour minimiser la charge de la mémoire.

Les graphismes mettront en lumière les challenges rencontrés et les solutions apportées pour garantir une expérience visuelle exceptionnelle pour les joueurs.





MODE PLAYER



Dans le mode joueur (Player), la machine prend l'initiative en générant automatiquement les diagonales sur la matrice grâce à la fonction RenderdDiagonals: (maximum 8 diagonales au niveau 1) . Le joueur observe attentivement les diagonales qui émanent de ce point pendant quelques secondes avant leur disparition (3 sec au 1er niveau , + 0.5 sec à chaque niveau supérieur), ensuite la machine affiche le point de départ avec la fonction generationPointDepart.

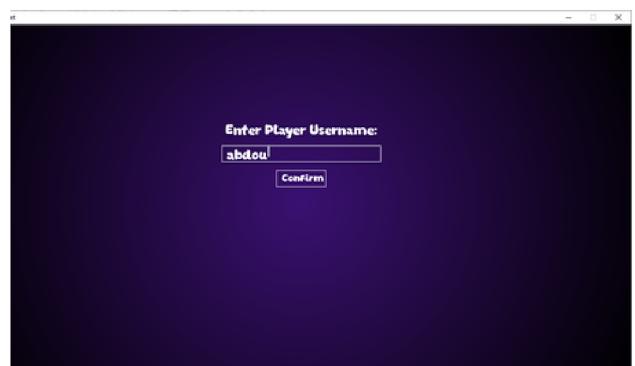
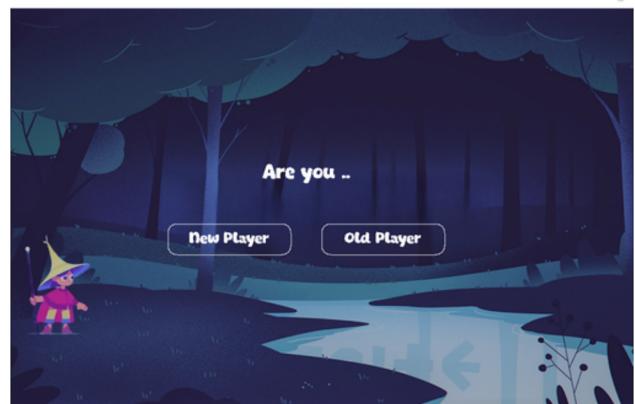
Après cette phase, le joueur doit sélectionner le point de sortie en se basant sur sa mémoire des diagonales en faisant un clic droit avec la souris ou bien en déplaçant la balle avec le bouton “ESPACE” sur le clavier et en cliquant sur “ENTRER”. Suite à cela, une animation est générée pour afficher la trajectoire de la matrice avec la fonction animationTrajectoire. La machine génère une autre matrice pour la prochaine partie. Trouver le point d’arrivée signifie qu'il a remporté la partie. Mais, le joueur doit gagner plusieurs parties pour passer au niveau suivant.

Lors de la partie, l'utilisateur peut retourner à la page d'accueil ou bien quitter le jeu à n'importe quel moment. Il peut aussi mettre la partie en pause. (Dans le cas où il clique sur pause avant la disparition des diagonales, elles réapparaîtront dès qu'il reprend la partie).



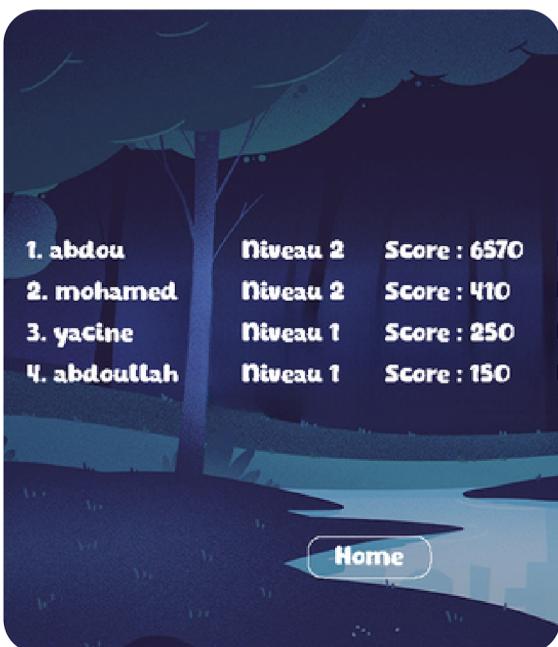
EVOLUATION DU NIVEAU

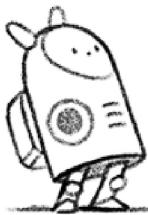
Le jeu consiste en des niveaux divisés en dix parties, où le joueur doit identifier le point d'arrivée de la bille quatre fois de suite pour progresser. Les configurations diagonales doivent être mémorisées, et le joueur doit anticiper la trajectoire correcte. Chaque échec entraîne la perte d'un cœur, avec une régression au niveau précédent après trois erreurs. Le défi inclut la recherche de solutions efficaces et l'évitement d'erreurs récurrentes. La complexité augmente graduellement avec de nouvelles configurations diagonales à chaque niveau.



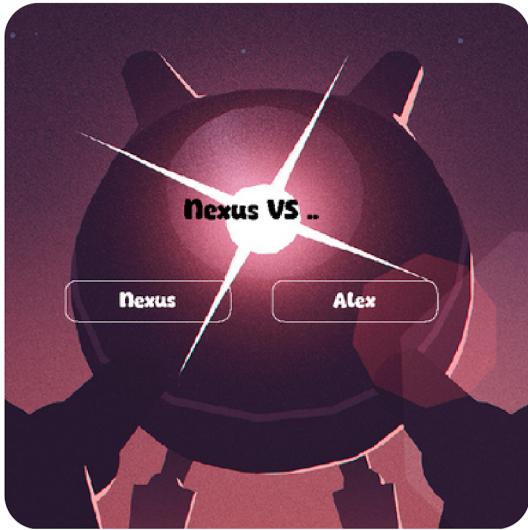
SCORE

Le programme évalue le score en accordant **10 points** pour chaque diagonale touchée par la balle dans la trajectoire correcte, en plus de **5 points** pour chaque diagonale restante dans la matrice au niveau 1. À chaque avancement de niveau, les points accumulés **sont doublés**.





MODE MACHINE



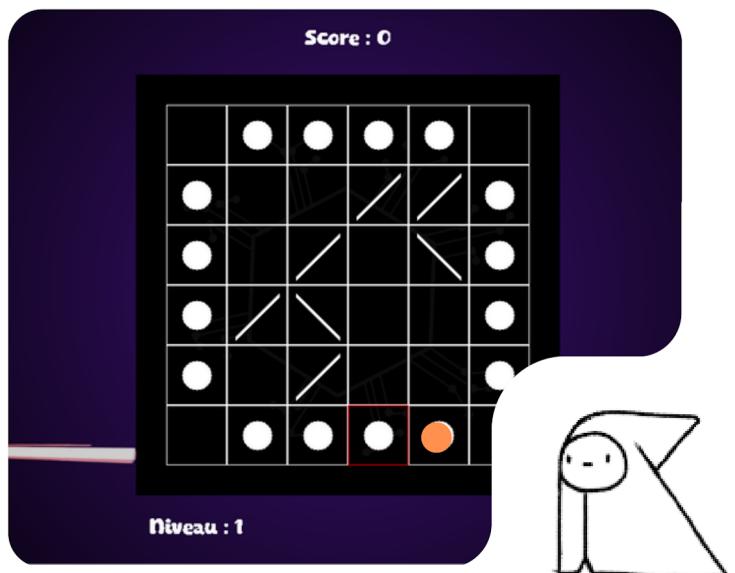
MODE MACHINE VS HUMAIN

Dans ce jeu, le joueur [interagit avec la machine](#) en fournissant les diagonales. Il a la possibilité de [remplir les diagonales](#) avec un clic gauche de la souris sur la position souhaitée ou en choisissant la cellule avec le clavier, suivie d'une pression sur la touche "ENTRER". Une fois qu'il a sélectionné une bille, il est considéré comme le point de départ, et le jeu débute. La machine [crée la matrice remplie de diagonales](#) à partir du rendu ([capture d'écran](#)) en utilisant [la fonction CreateMatrixWithPixel](#). Elle calcule ensuite la trajectoire de la bille et génère l'animation correspondante. Tout au long du jeu, le bouton "home" reste accessible, permettant au joueur de revenir à tout moment à la page d'accueil.

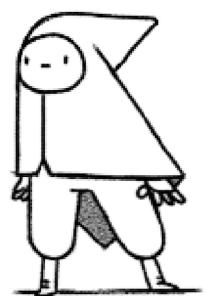
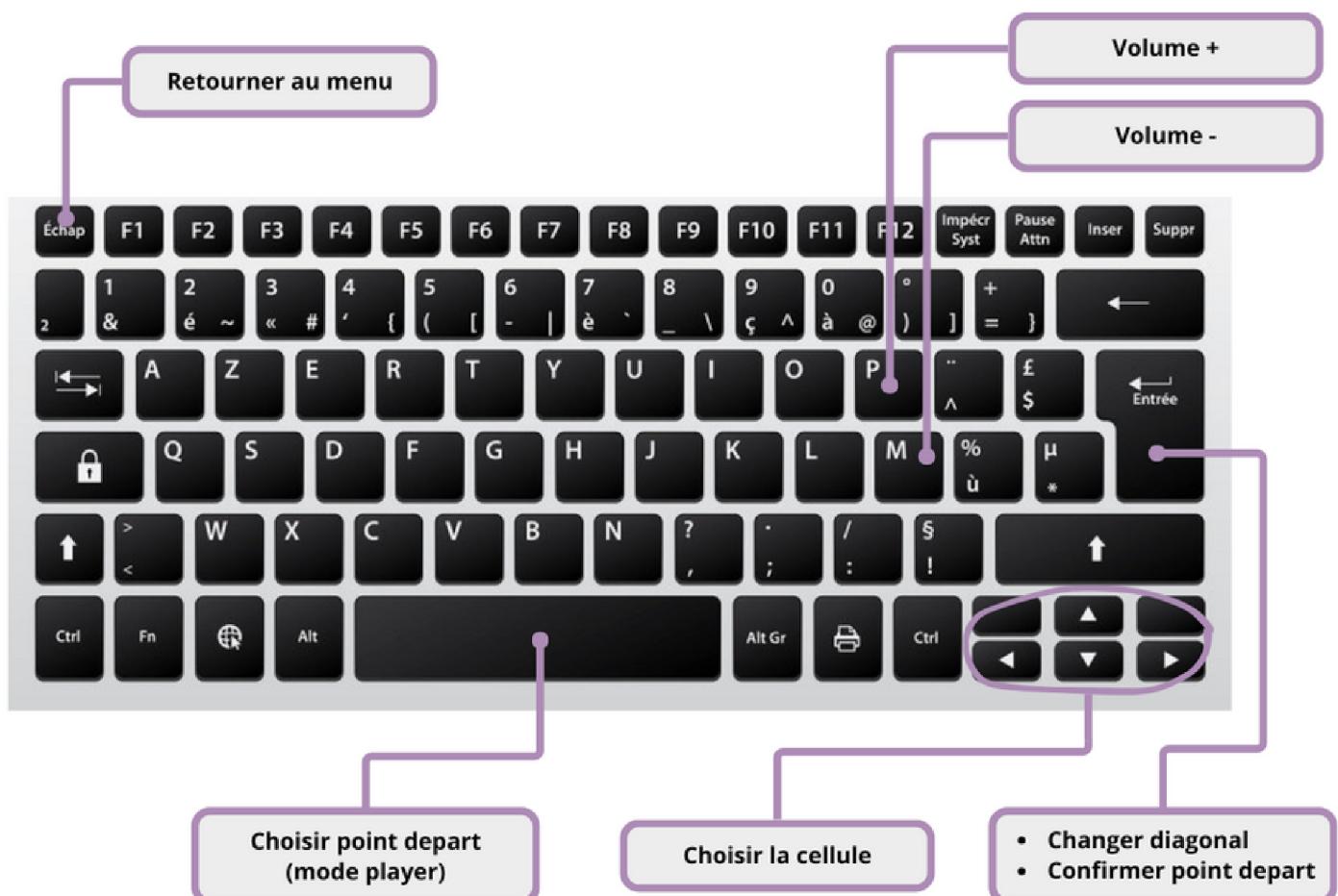
MODE MACHINE VS MACHINE

Dans ce mode, la machine utilise la [fonction generationDiagonals](#) pour générer [aléatoirement les diagonales](#). Le point de départ et la direction sont également générés à l'aide des fonctions "[generationPointDepart](#)" et "[Direction_depart](#)". Une copie du rendu (render) est récupérée, et à partir de celle-ci, une matrice est construite avec [la fonction CreateMatrixWithPixel](#).

La machine calcule ensuite la trajectoire, en affichant le parcours sous forme [d'animation](#). À la fin de l'animation, une nouvelle fenêtre s'affiche, demandant à l'utilisateur s'il souhaite [continuer à jouer](#) ou [retourner à l'accueil](#). Après chaque série de quatre parties, la machine [incrémente le niveau](#) jusqu'à atteindre une taille de 20 (Niveau 15).



GUIDE DU JEU

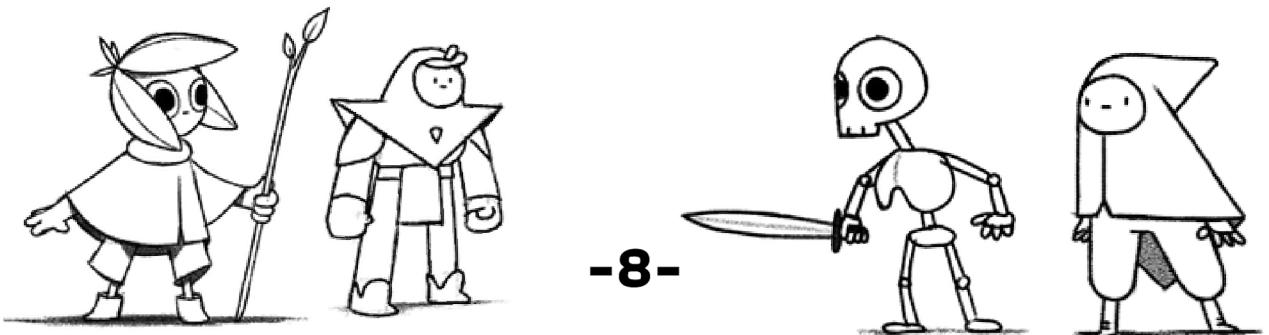


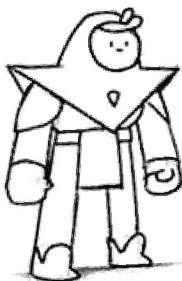
CONCLUSION

En conclusion, “MINDMAZE: Alex vs Nexus” offre une expérience de jeu passionnante où mémoriser et anticiper sont les clés du succès .Par apprentissage de la manipulation du langage C et la bibliothèque SDL2 , le jeu propose deux modes uniques qui testent la mémoire visuelle et la prise de décision rapide des joueurs.

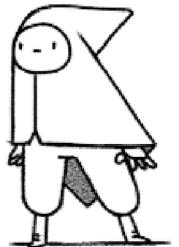
Notre rapport détaille les aspects fonctionnels du jeu, de son mécanisme intéressant aux stratégies pratiques que les joueurs peuvent adopter.

En résumé, ce jeu promet d'offrir du plaisir tout en mettant à l'épreuve les compétences de mémorisation et de réflexion en temps réel des joueurs.





BIBLIOGRAPHIE



- <https://wiki.libsdl.org/SDL2/Tutorials>
- <https://wiki.libsdl.org/SDL2/FrontPage>
- <https://alexandre-laurent.developpez.com/tutoriels/sdl-2/>
- <https://www.youtube.com/watch?v=7xFaWRzWqr0&list=PL894088275284BDEA>
- <https://chat.openai.com/> (Game story inspiration)
- https://drive.google.com/file/d/1i5LV0_NHBlcUmFottJpbYBLGUSZw3aMQ/view?usp=sharing (Consignes du projet)
- <https://www.midjourney.com/> (Graphismes)
- <https://elevenlabs.io/> (Sound of the story)