# RAPPORT

## Création jeu du snake

## Sommaire :

# Introduction du jeu

Le but du jeu est d'avoir le plus haut score. Pour ce faire, on incarne un serpent de 10 segments dans un terrain de 60*40 cases. On peut le diriger grâce aux flèches directionnelles du clavier. Pour jouer et ainsi augmenter le score, on doit manger des pommes. Il y a 5 pommes placées aléatoirement sur le terrain. Si on mange une pomme, on grandit de deux segments. Si on sort du terrain, on perd. Si on avance sur son propre segment, on perd. Pour ajouter plus de difficulté, 50 obstacles d'une case chacun sont mis à disposition sur le terrain. Si vous vous heurtez à un obstacle, vous perdez.

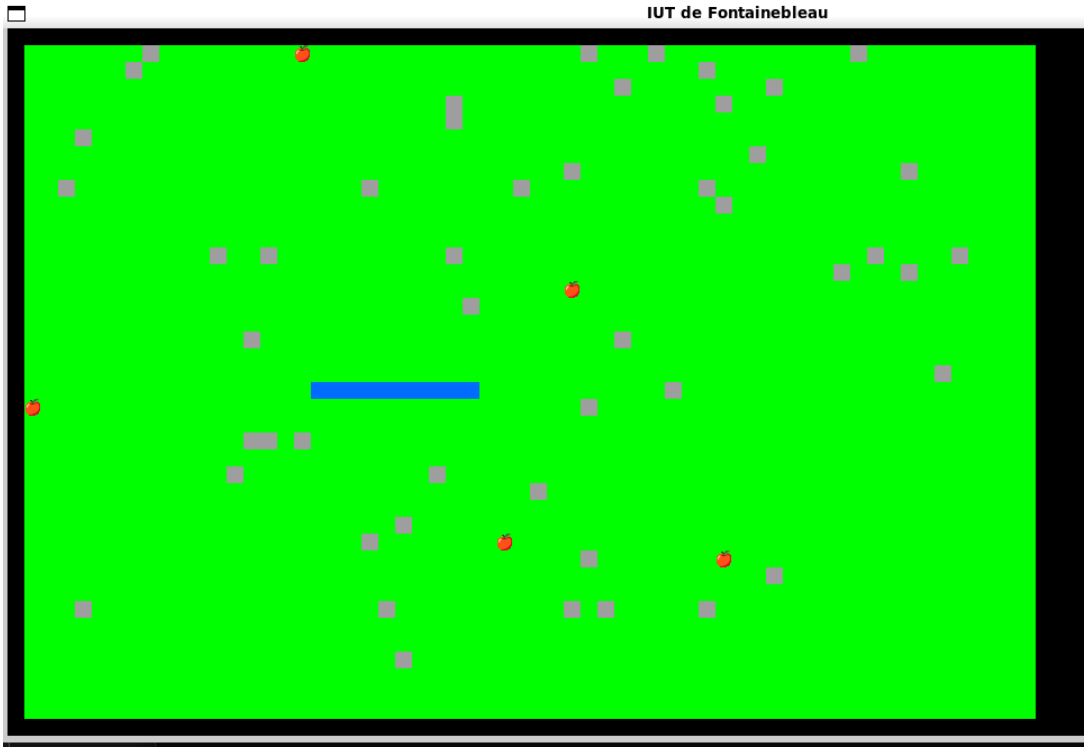
# Descriptions des fonctionnalitées

Le programme est composé de plusieurs fonctionnalités.

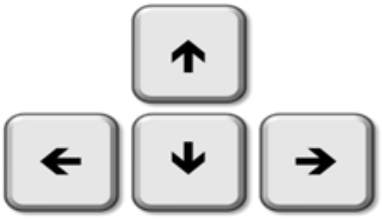
<u>Menu d'accueil personnalisé :</u>
Le programme démarre avec un menu d'accueil personnalisé, ou l'utilisateur peut choisir grâce aux cliques de la souris, comme des boutons, entre : **Jouer** ou **Quitter**. Le bouton **Jouer** permet de jouer au jeu, tandis que le bouton **Quitter** permet de quitter le programme.

Terrain de jeu interactif :

Le terrain de jeu est affiché avec des bordures noires, qui font la délimitation du terrain. Il est composé d'un serpent de couleur bleue, de pommes qui sont constitué d'une image de pomme ainsi que d'obstacles de couleur grise. Les pommes apparaissent aléatoirement par nombre de 5. Les obstacles apparaissent eux aussi aléatoirement par nombre de 50. Les pommes ne peuvent pas apparaître sur les obstacles, pour éviter de rendre une pomme immangeable.



<u>Gestion des mouvements du serpent et des touches du claviers :</u>
Le serpent peut se déplacer à **droite**, à **gauche**, en **haut**, et en **bas** en réponse des touches du clavier, les **flèches** directionnelles du clavier. Il avance en ajoutant un segment en tête du serpent et en supprimant en queue du serpent un segment.



Bien sûr, ceci a été conçu pour que quand on va dans une direction, on ne peut pas aller à son opposé. Par exemple, si je vais à droite, je ne pourrais pas aller à gauche (pour éviter que le serpent fasse un demi-tour sur lui-même et donc une collision immédiate).

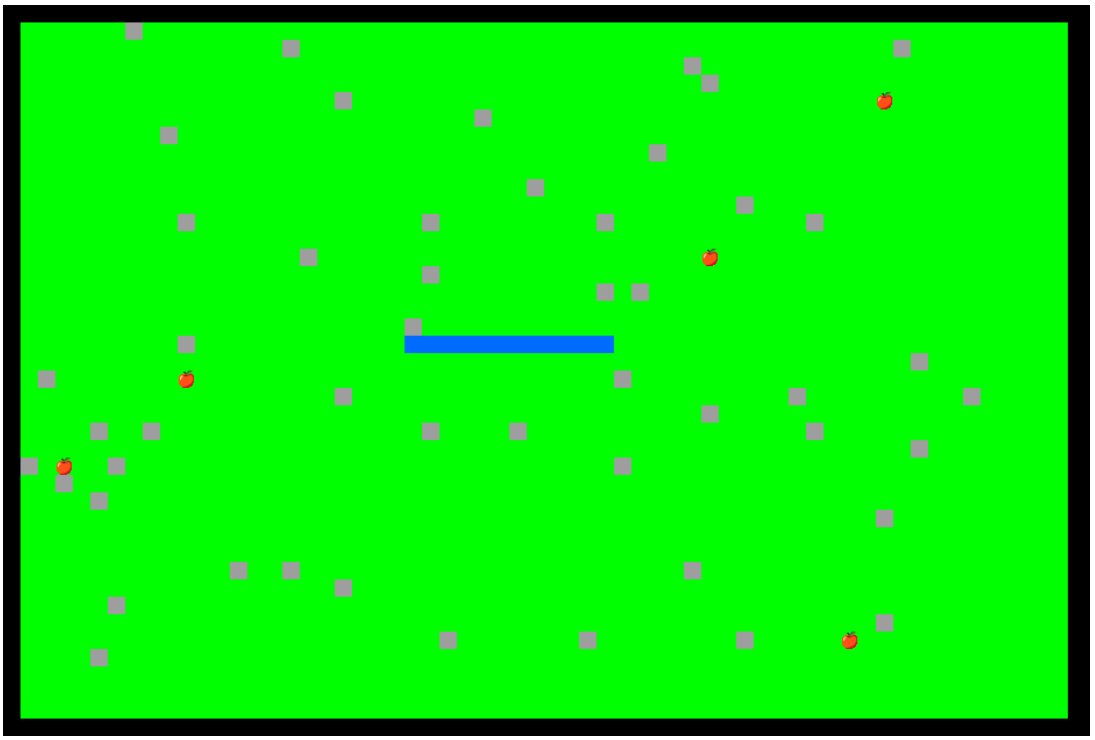
Nous avons aussi la touche **Echap** qui permet de quitter le programme.


Il y a aussi la touche **Espace** qui permet de mettre en pause le programme.

Gestions des collisions :
Dans ce code, la détection marche avec les coordonnées de chaque chose. Si le segment qu'on veut ajouter à la tête sur serpent se trouve sur une case où on veut créer la collision, la collision s'effectue.

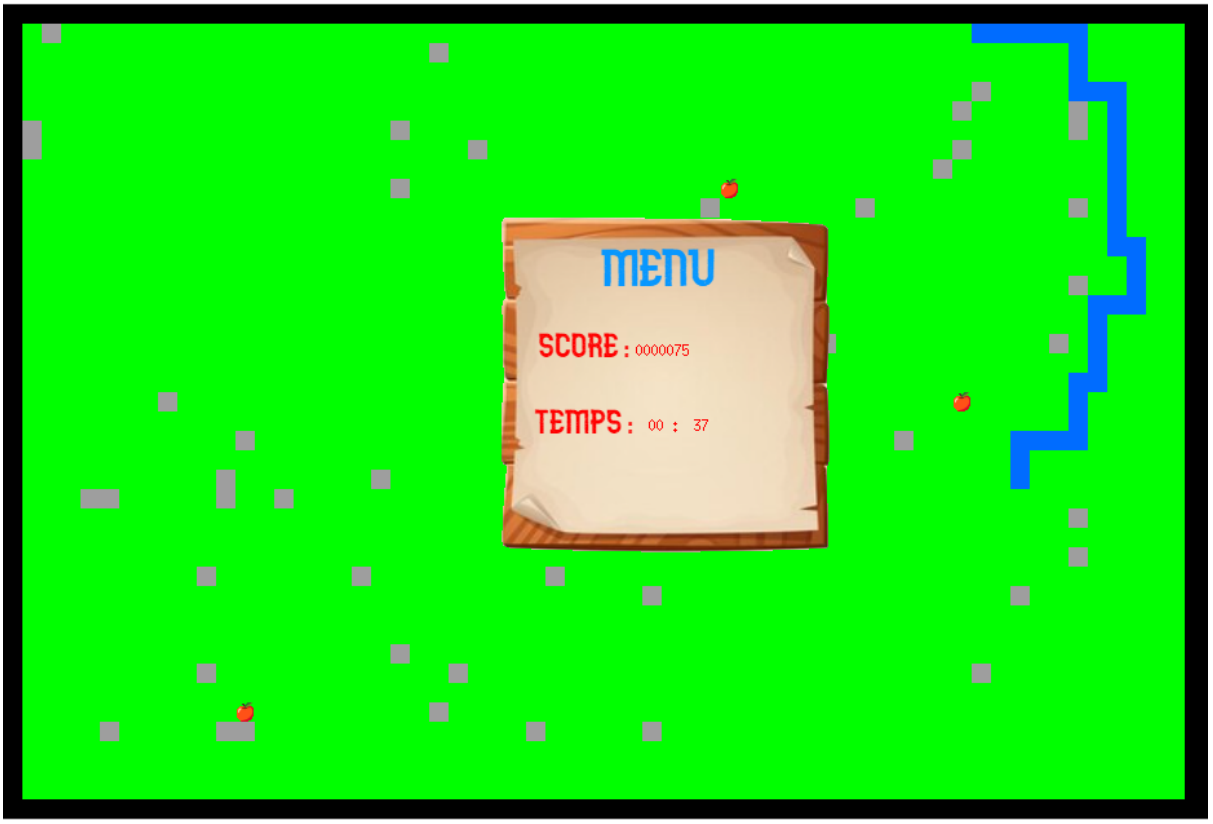
Le jeu détecte plusieurs collisions. La collision avec la bordure du terrain, qui conduit à la mort du serpent et revient à perdre la partie. Il y a aussi la collision avec le serpent lui-même et la collision avec les obstacles qui mène également à la mort du serpent. Un menu de fin s'affiche affichant le score effectué ainsi que le temps qu'a duré la partie. Pour quitter ce menu, il faut appuyer sur **Espace**.


Le jeu détecte également les collisions avec les pommes. À la différence d'autres situations, celles-ci n'aboutissent pas à la fin du serpent. Au contraire, elles entraînent une croissance de sa taille de deux segments. Lorsque la pomme réapparaît, elle ne réapparaîtra pas sur le serpent ou sur un obstacle.



Menu pause :

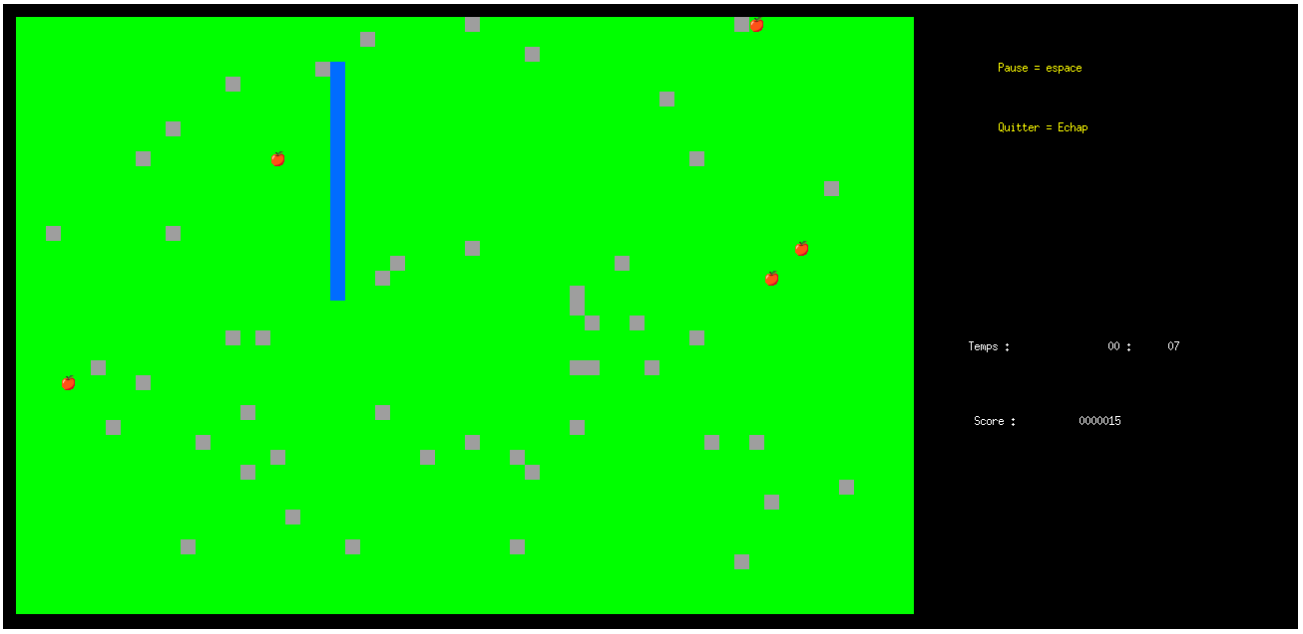
Le jeu, permettant au joueur de mettre en pause, offre cette fonctionnalité. Pour ce faire, il vous suffit d'appuyer sur la touche **Espace**. Un écran de pause s'affiche. Il affiche notre temps de jeu, ainsi que notre score actuel. Le temps s'arrête de lui-même quand on appuie sur la touche permettant la pause, pour ne pas prendre en compte le temps dans chaque pause.




Pour revenir au jeu, on a juste à appuyer sur la touche **Espace**.

## Affichage du score ainsi que du temps :

Le temps du jeu est affiché grâce à la gestion du temps incluse dans la bibliothèque graphique. Le temps est affiché en minute et en seconde. Il commence à se mettre en route quand on appuie sur le bouton jouer dans le menu d'accueil. Il s'arrête quand on appuie sur la touche espace et se remet en route quand on appuie sur la touche espace à nouveau. Quand on meurt, le temps s'arrête.
Pour le score, dès qu'une collision a lieu entre le serpent et une pomme, une variable de type int qui s'appelle score augmente de 5. On a juste affiché la valeur de cette variable pour avoir le score.
Le score et le temps sont affichés à droite du jeu, sur le fond noir.




Lorsqu'on est dans le menu pause ou bien le menu de fin, le score et le temps s'affichent également dans le menu de pause.

# Découpage des fichiers

Le programme du jeu a été décomposé en plusieurs fichiers afin de rendre le code plus lisible et facile à maintenir. Chaque fichier source a une responsabilité spécifique, contribuant ainsi à une organisation efficace du code.

**Fichiers sources :**

main.c
Le fichier **main.c** est la clé du programme. Il permet de gérer toutes les fonctions utilisées pour les exécuter. Il contient toute la fonction **main();** .

menu.c et menu.h
Le fichier menu.c et son en-tête menu.h permettent de gérer l'affichage du menu.

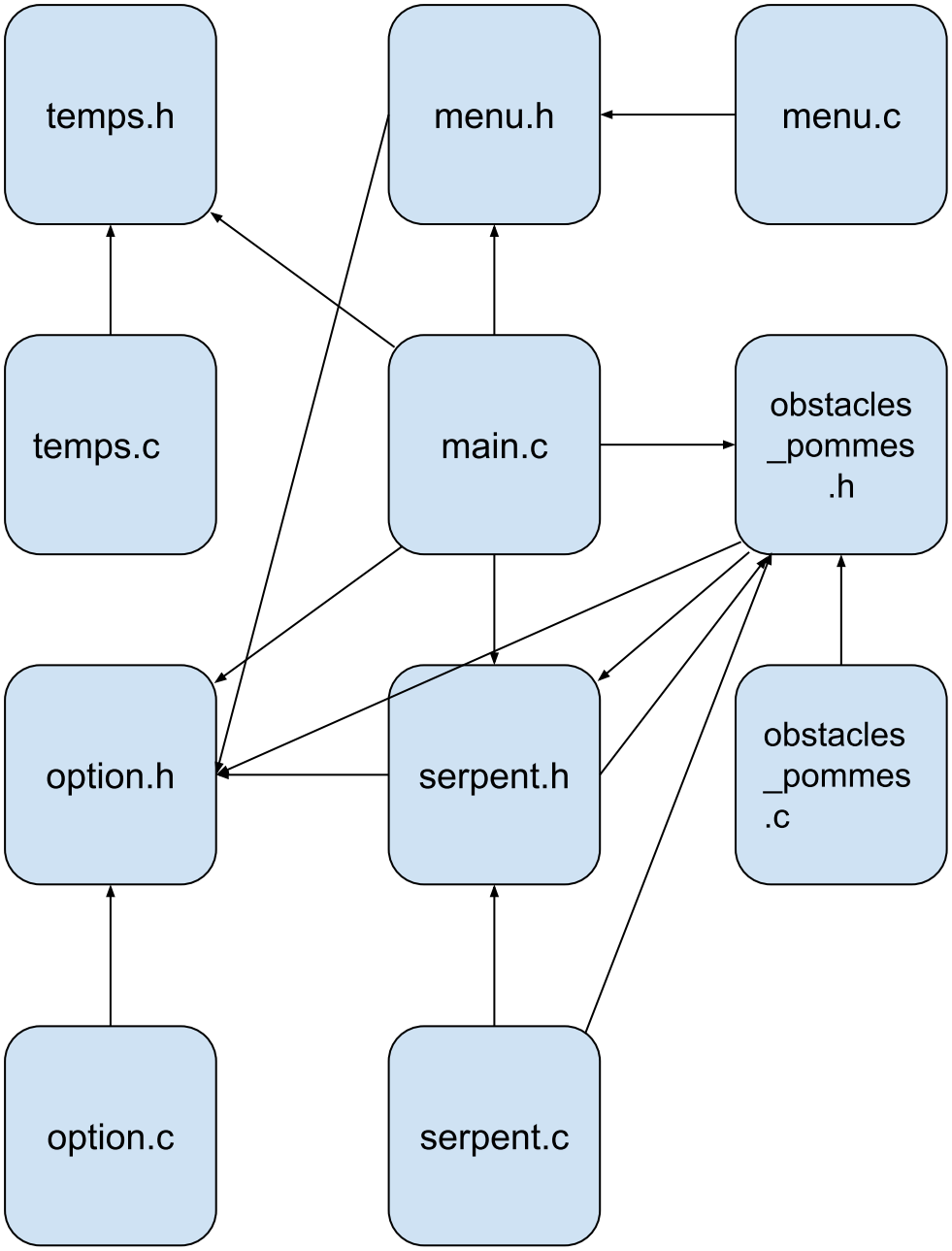
serpent.c et serpent.h
Le fichier serpent.c et son en-tête serpent.h permettent de gérer la logique du serpent, y compris l'initialisation, son déplacement, la détection de collision et de sa croissance.

obstacles_pommes.c et obstacles_pommes.h
Le fichier obstacles_pommes.c et son en-tête obstacles_pommes.h gère la création, le positionnement aléatoire et l'affichage des pommes ainsi que des obstacles sur le terrain de jeu.

temps.c et temps.h
Le fichiers temps.c et son en-tête temps.h permettent de gérer la mesure et d'afficher le temps écoulé lors du jeu.

option.c et option.h
Le fichier option.c et son en-tête option.h contiennent la définition des structures CaseSerpent, Obstacle et Pastilles.

Diagramme du découpage :



# Explication des données sur serpent

Structure :
Dans le programme du jeu, les données représentant le serpent sont stockées dans une structure appelée **CaseSerpent**. Chaque élément de cette structure représente une partie du corps du serpent et contient les coordonnées (x, y) de la position de cette partie sur le terrain.

Initialisation :
Au début de la partie, la fonction **initialiser_serpent** est appelée pour positionner le serpent au centre du terrain. Chaque segment du serpent est espacé de 15 pixels, et sa longueur initiale est définie par la variable **longueurSerpent**.

Déplacement du serpent :
La fonction **DeplacerSerpent** gère le déplacement du serpent en fonction de la direction fournie en argument. Elle efface d'abord la queue du serpent, puis ajoute la tête du serpent. Le déplacement est géré par l'utilisateur. La fonction permet de mettre à jour les coordonnées de la tête du serpent en fonction de la direction choisie.

Collision du serpent :
La fonction **DeplacerSerpent** effectue également la détection des collisions avec les bords du terrain, le serpent lui-même et les obstacles. Si une collision est détectée, les variables **collision_terrain** ou **collision_serpent** sont mises à zéro, indiquant la fin de la partie. Elle détecte aussi si on entre en collision avec une pomme.

Croissance du serpent :
La fonction **manger_pastille** est appelée pour vérifier si la tête du serpent a rencontré une pastille. Si c'est le cas, la fonction ajoute deux segments au corps du serpent, permettant ainsi sa croissance.

La gestion du serpent est importante pour le bon fonctionnement du jeu. C'est l'élément principal du jeu.

# Avis personnel

Tarehi :

La création du jeu du Snake en langage C, une première pour moi, a été une aventure marquante. Ce projet a été l'occasion de repousser mes limites et de penser par moi-même dans un nouvel environnement de programmation. La manipulation du langage C représente un défi stimulant, nécessitant persévérance et ténacité pour comprendre et maîtriser ses subtilités.

La collaboration avec mon binôme a été cruciale dans cette expérience. La résolution conjointe des problèmes et l'échange d'idées ont non seulement renforcé notre compréhension collective, mais ont également souligné l'importance de l'entraide dans le processus d'apprentissage.

Ce qui distingue notre version du Snake, au-delà de son attrait classique, c'est sa simplicité et son économie. En tant qu 'adepte de jeux vidéo, particulièrement des variations du Snake, contribuer à la création d'une version personnalisée a ajouté une touche nostalgique à cette expérience. La réflexion sur notre propre enfance et la passion pour les jeux vidéo ont été des éléments moteurs de notre motivation.

Cependant, cette aventure n'a pas été exempte de défis. La compréhension des collisions a été particulièrement ardue au début, mais la persévérance a été la clé pour surmonter ces obstacles. Cette expérience a été plus qu'un simple exercice de programmation ; elle a été un test de résilience et d'aptitude à résoudre des problèmes de manière autonome.

En conclusion, la création du Snake en langage C a été une expérience formatrice qui va au-delà des compétences techniques acquises. Elle a été le reflet de la persévérance et de la ténacité nécessaires pour surmonter des défis inédits. Ce projet a non seulement enrichi mes connaissances en programmation, mais a également renforcé ma confiance dans ma capacité à aborder de nouveaux langages et à relever des défis techniques de manière autonome.

**Thomas :**

J'ai beaucoup aimé faire ce projet. Le sujet était quelque chose de plutôt compréhensible, sachant que je jouais déjà au jeu du snake sur google, donc je connaissais un peu les bases. Le projet a été beaucoup plus compliqué que ce que je pensais.
Personnellement, je pensais que cela allait prendre moins de temps que prévu, mais pas du tout. La preuve étant, on a beaucoup eu besoin d'une semaine supplémentaire pour finir tout ça. J'ai aussi eu des moments où je restais bloqué sur une erreur (plus ou moins bête) pendant un long moment. Il m'a permis aussi d'en apprendre plus dans le langage du C.
La fabrication du jeu montre l'important impact que l'utilisateur peut avoir sur le jeu, avec les déplacements, le menu accueil etc.
L'intégration des images dans le jeu permet un meilleur visuel au projet, améliorant l'esthétique du jeu.
Le projet m'a appris comment la création d'un code, d'un projet pouvait prendre du temps et qu'il s'y prendre dès le début.