Hamelin Yannis
Mickaël Albarello

Rapport de projet tutoré

SNAKE

SOMMAIRE

INTRODUCTION

Le projet est un jeu de snake que nous avons appelé le Dark Snake en raison de sa difficulté croissante au fur et à mesure de la partie. Dans ce jeu le joueur contrôle un serpent se déplaçant sur un damier de couleur verte et verte foncée.

Description des fonctionnalités du programme

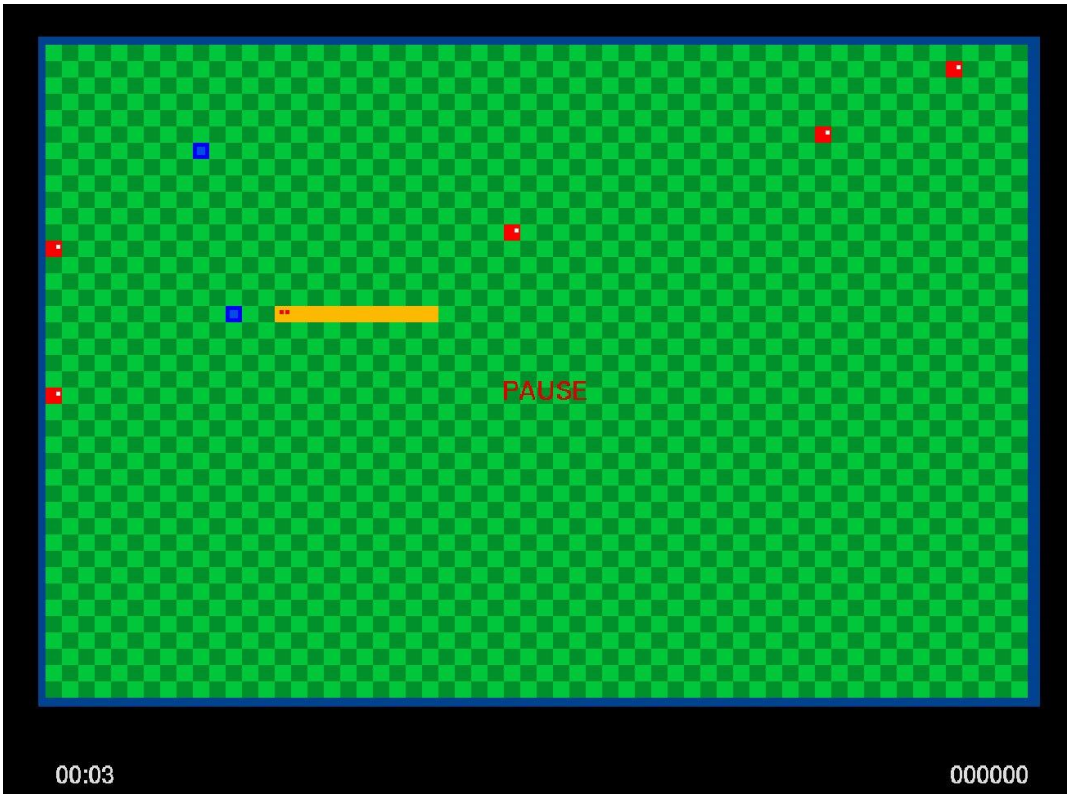
Le serpent peut manger des pommes qui le feront grandir de deux cases, et emprunter des trous de ver afin de prendre un raccourci. Si le serpent se mord la queue ou se cogne contre les extrémités, c'est la fin de la partie. Chaque pomme consommée rapporte 5 points.

Toutefois il est à prendre en compte qu'au fur et à mesure que le temps passe, le serpent se déplace de plus en plus vite, et également plus grand à force de manger des pommes. Ainsi il deviendra plus dur à contrôler.

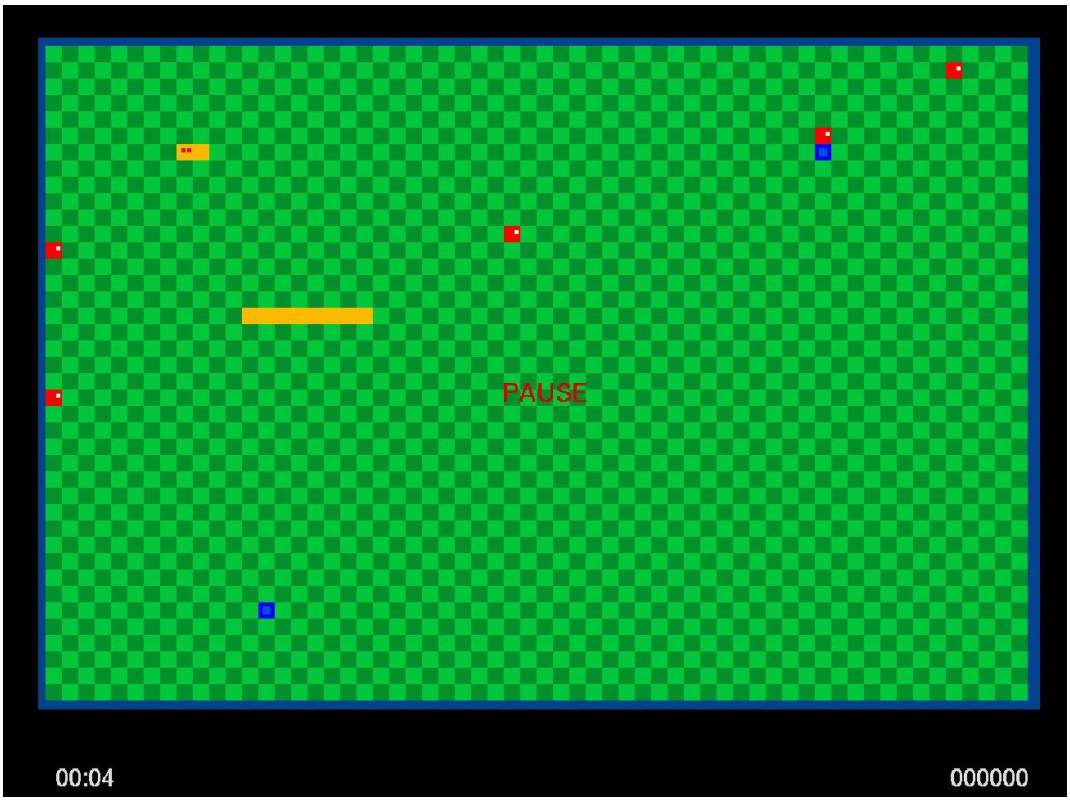
Le joueur déplace le serpent à l'aide des flèches du clavier, et peut à tout moment mettre le jeu en pause en appuyant sur la barre d'espace. Dans le mode pause, le joueur peut quitter le jeu en appuyant sur la touche "échap".

Il est également possible de changer la taille de la fenêtre de jeu, en cours de partie, avec les touches "+" et "-".

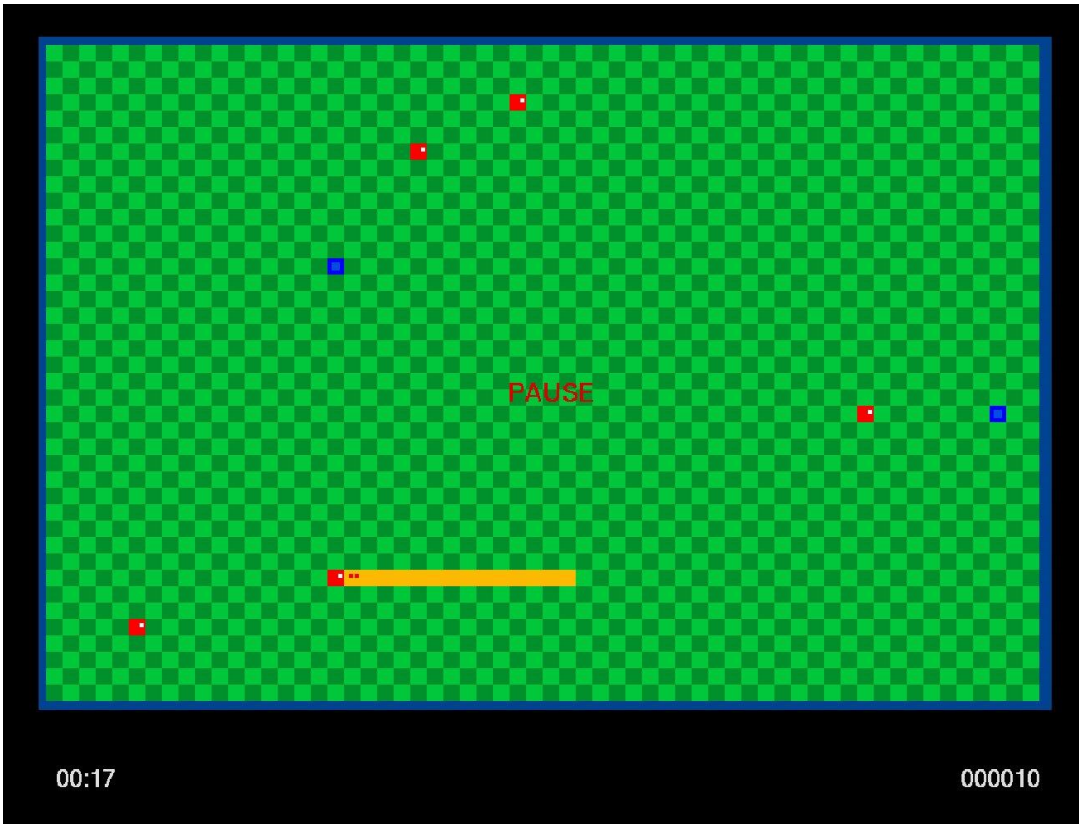
Snake avant de traverser un portail



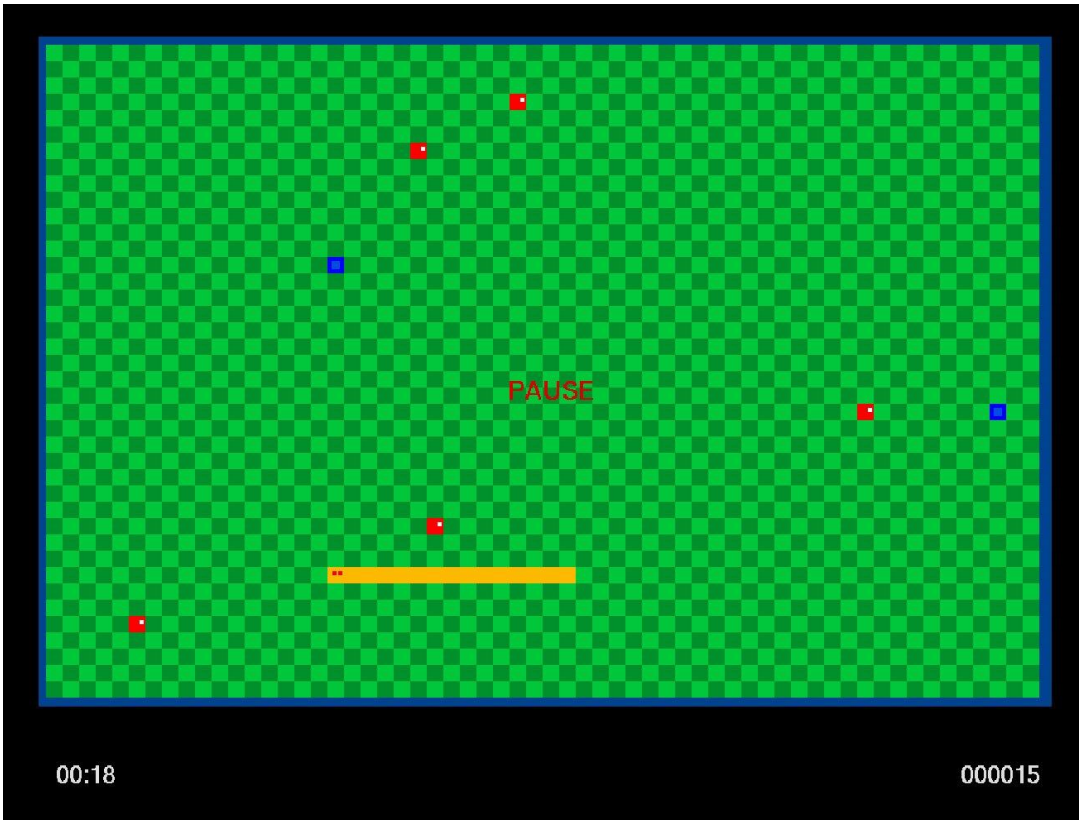
Snake traversant un portail



Snake avant de manger une pomme

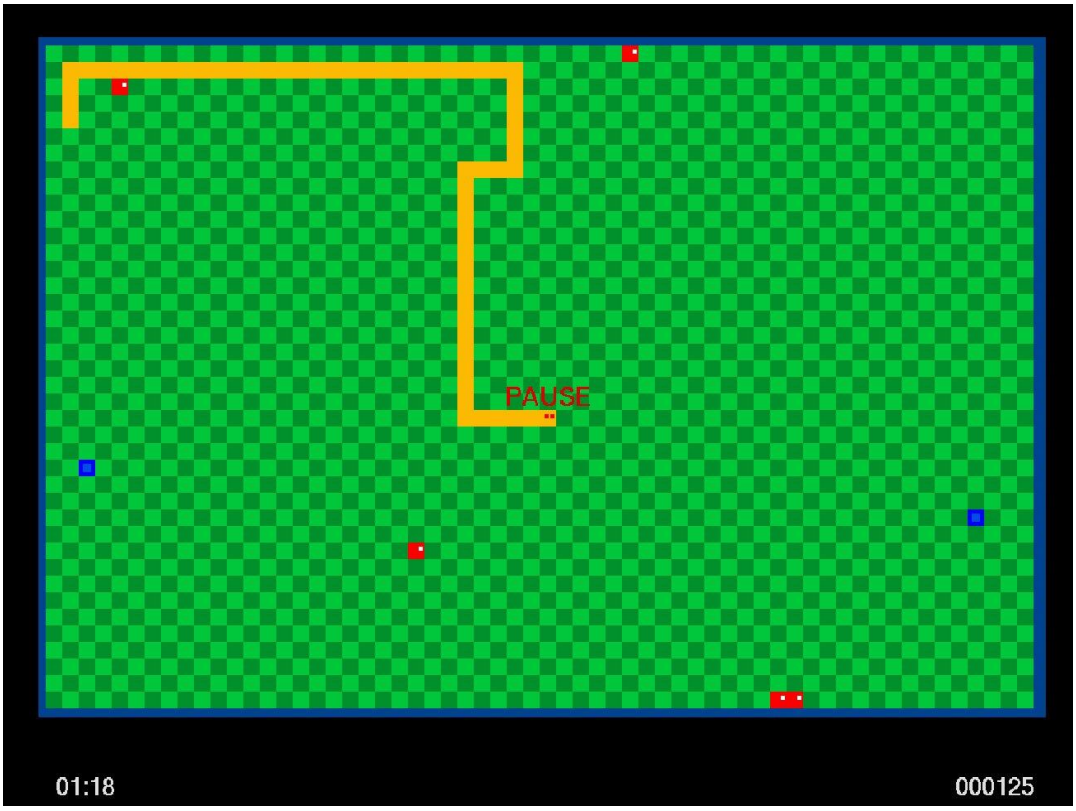


Snake après avoir mangé une pomme



Ecran de fin de partie

Snake ayant mangé plusieurs pommes. Score à 125



Présentation de la structure du programme

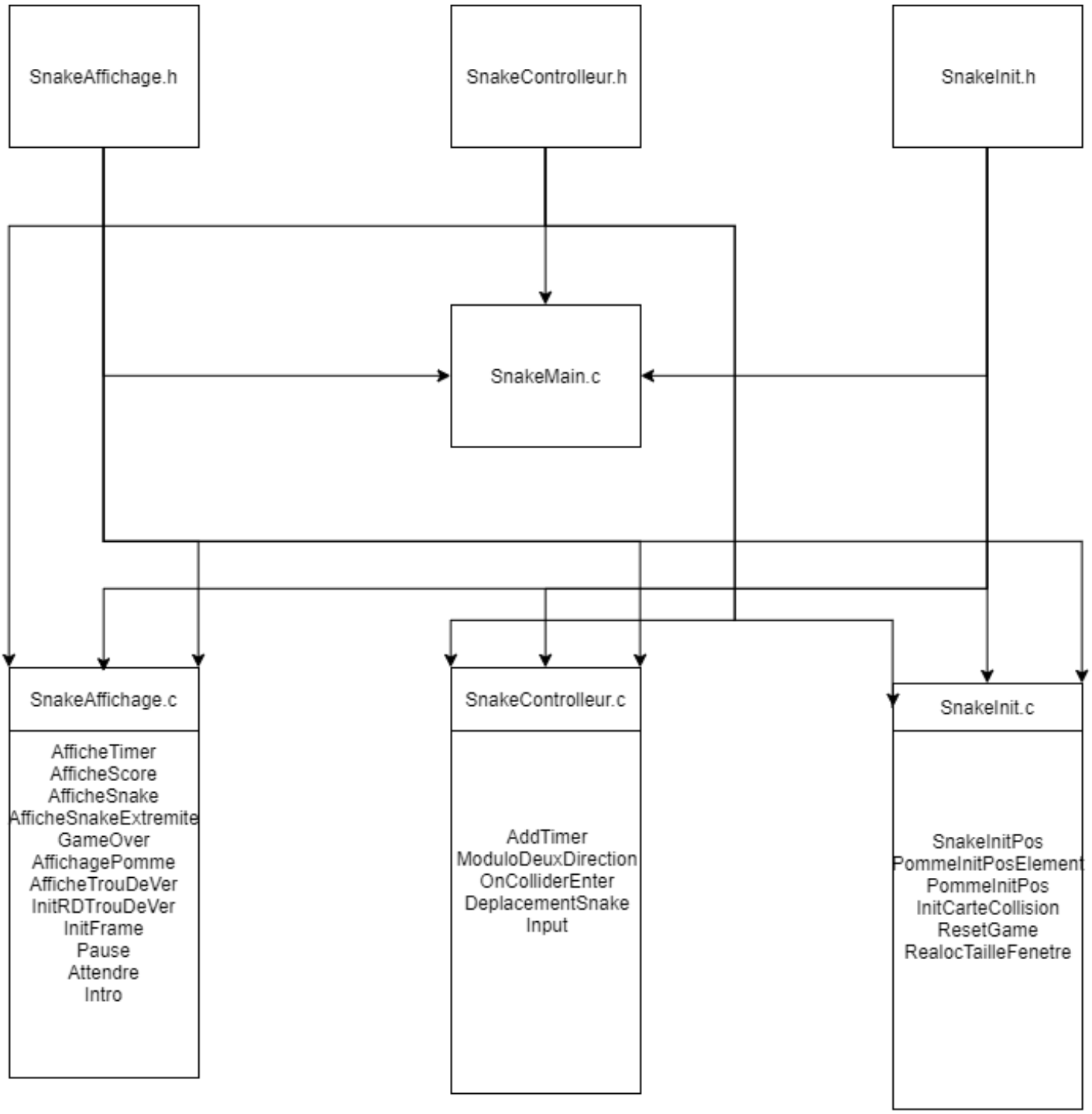
Le fichier **SnakeAffichage.c** sert à l'affichage du jeu. Chaque élément affiché l'est grâce à une fonction contenue dans SnakeAffichage.c . Il inclue SnakeAffichage.h, SnakeInit.h et SnakeControlleur.h .

Le fichier **SnakeControlleur.c** sert à permet de détecter les entrés que le joueur effectue (appuyer sur une touche), mais également au serpent de se mouvoir dans la fenêtre, de détecter les collisions, d'adapter la position de sa tête dans le tableau (donc dans la mémoire). Il inclue SnakeControlleur.h , SnakeInit.h , SnakeAffichage.h .

Le fichier **SnakeInit.c** sert à initialiser sur la carte l'emplacement des éléments de l'environnement avec lesquels le serpent peut interagir (pommes, trous de ver). Il permet également de redémarrer la partie, ou encore de contrôler la taille de la fenêtre en cours de jeu. Les fonctions contenues dans ce fichier se lance uniquement en début de partie ou lors d'une collision avec une pomme. Il inclue SnakeControlleur.h , SnakeInit.h , SnakeAffichage.h .

Le fichier **SnakeMain.c** est le fichier principal, dans lequel on initialise les variables, et on utilise les fonctions d'initialisation. Il sert également à créer la fenêtre de jeu. Dans ce fichier, on utilise une fonction (Input) qui permet de renvoyer une sortie en fonction de la touche sur laquelle le joueur appuie. Ces sorties sont: **-1** (et dans ce cas on quitte le jeu); **0** (et donc on est en cours de partie); **1** (on est en pause); **2** (on sort de la pause); **4** (redémarre la partie); **au-dessus de 20** (game over). La fonction Input est utilisée dans une boucle.

Le fichier sert également à gérer le temps du chronomètre et également l'intervalle de temps entre chaque action du serpent. Il inclue SnakeControlleur.h , SnakeInit.h , SnakeAffichage.h , Time.h et Graph.h .



Explication de la forme des données et leurs transformations au cours du programme

Le serpent est stocké dans un tableau de structure qui contient la position x et y de chaque élément du serpent. Ce tableau est défini à la taille maximum que le serpent peut avoir sur la carte, c'est-à-dire 2400. On prend comme éléments supplémentaires, la taille du serpent actuelle et la position de l'élément "tête" dans le tableau de structure. A chaque déplacement, on écrit dans la position suivante du tableau les valeurs des anciennes positions dans la case juste avant, plus le vecteur direction et on mais à jour la position de la tête, puis on affiche la tête et on remplace le cou. Par la suite, on décrémente la valeur de la tête avec celle de la taille pour obtenir la position de la queue, que l'on remplace par un damier.
La collision du serpent est placée au fur et à mesure qu'il se déplace dans un tableau à deux dimensions (60 par 40), la fonction OnColliderEnter regarde à la position de la tête dans le tableau des collisions. Si la case a une valeur supérieure à 0 alors il fait un des événements selon l'id de la case. Par exemple, pour l'id 2 , il mange une pomme et le score est incrémenté de 5 points. Dès que le serpent arrive à la fin du tableau, il est directement ramené au début grâce à la fonction (modulo deux directions). La lecture des éléments du tableau ne sort donc pas de la zone mémoire allouée.

Conclusion personnelle Yannis

La création d'un programme comme celui-ci est plus complexe qu'il n'y parait et que je m'y attendais. J'ai bien pu me rendre compte des enjeux qu'incombent un travail en groupe, notamment la répartition des tâches et surtout l'importance de la discussion dans un groupe, pour ne pas perdre du temps à faire la même chose chacun de son côté. La mise en communs des connaissances de chacun et une bonne entente permettent sans nul doute à l'aboutissement rapide d'un projet.

Conclusion personnelle Mickaël

J'ai pu me rendre compte que l'utilisation de la bibliothèque graphique qui nous était mise à disposition est plus simple d'utilisation qu'il n'y parait. Je n'ai pas pu utiliser de sprite car la fenêtre de jeu peut se redimensionner pour s'adapter à toutes les résolutions d'écran et la fonction AfficherSprite ne permet pas de redimensionner l'image. J'ai pu tester le programme sur un raspberry pi et il fonctionnait de manière fluide.