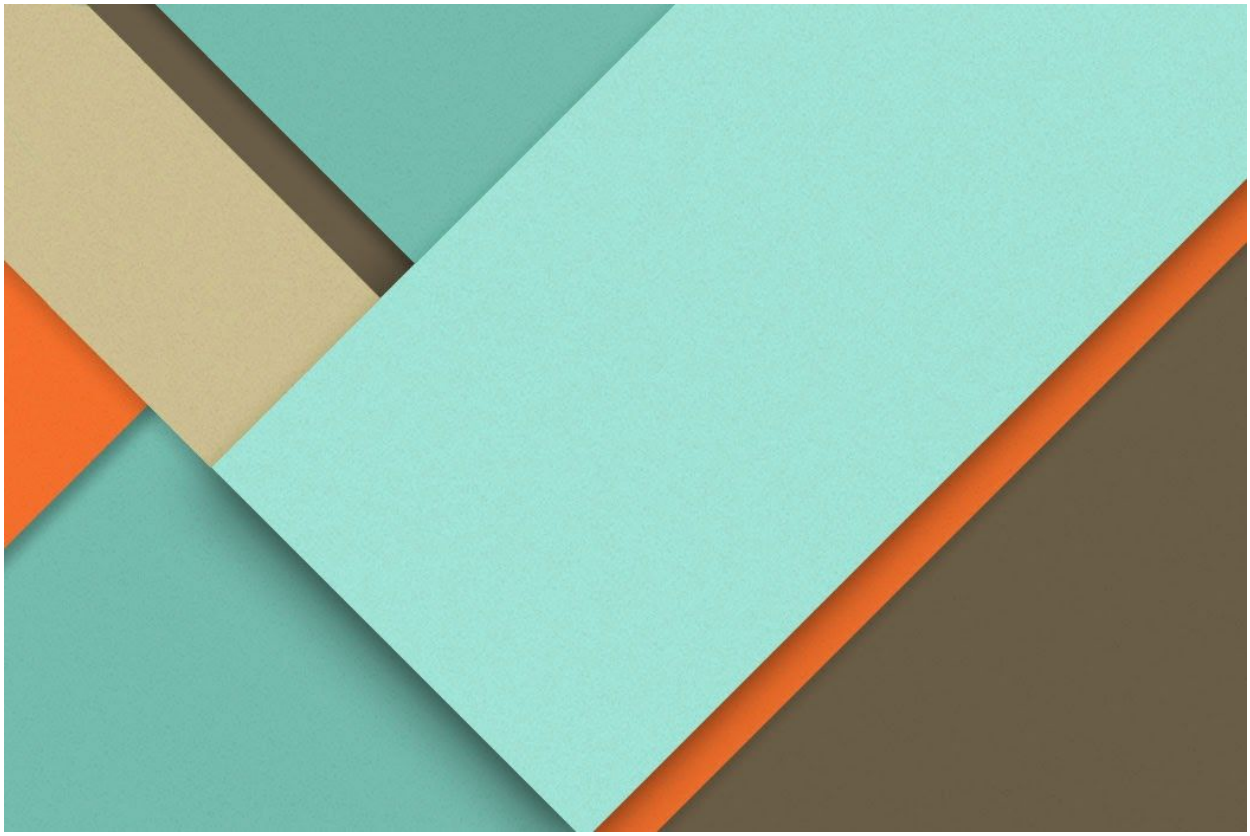


Bataille Navale

13.12.2020

Es-sebbani Naim, Guestin Jason

Projet C Avancé

Sommaire

I) Introduction

La Bataille Navale est un jeu qui est né en 1931 et qui oppose deux joueurs. Les deux joueurs disposent de 5 bateaux sur un plateau de taille 10x10 et dont le but est de détruire toute la flotte ennemie.

Nous avons dû réaliser une adaptation de ce jeu sur terminal avec le langage C. Le jeu comporte quelques spécificités telles que l'ajout de différents tirs spéciaux ou la création d'une IA contre laquelle l'utilisateur peut choisir de jouer.

II) Découpage du projet

Pour réaliser ce projet de façon modulable, nous avons fait le choix de le diviser en plusieurs petits fichiers correspondant à chaque "module" du jeu.

Plateau:

Nous avons dû mettre en place un module permettant de gérer une matrice, l'initialiser, la détruire, l'afficher... Cette matrice nous sert de plateau de jeu.

Jeu:

Un module Jeu permettant de gérer la plupart des variables de la partie, gérant aussi les premières interactions avec l'utilisateur avant le lancement de la partie et exécutant les fonctions nécessaires au moment opportun.

User:

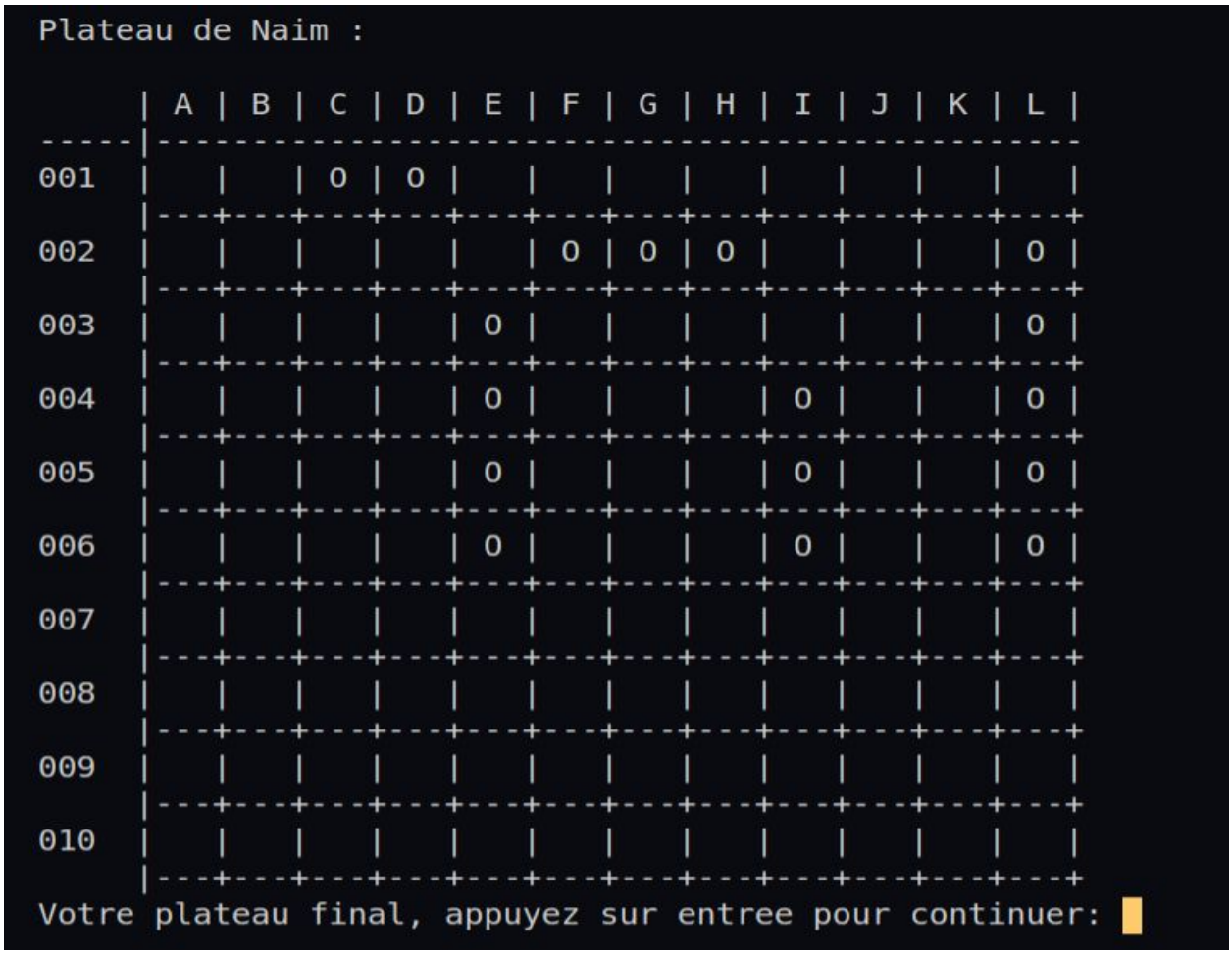
Ce module nous permet de gérer des interactions avec le joueur tel que lui demander les coordonnées de tir, choisir le tir spécial...

IA:

Le module contenant l'IA, sa logique et les quelques autres fonctions nécessaires à son bon fonctionnement

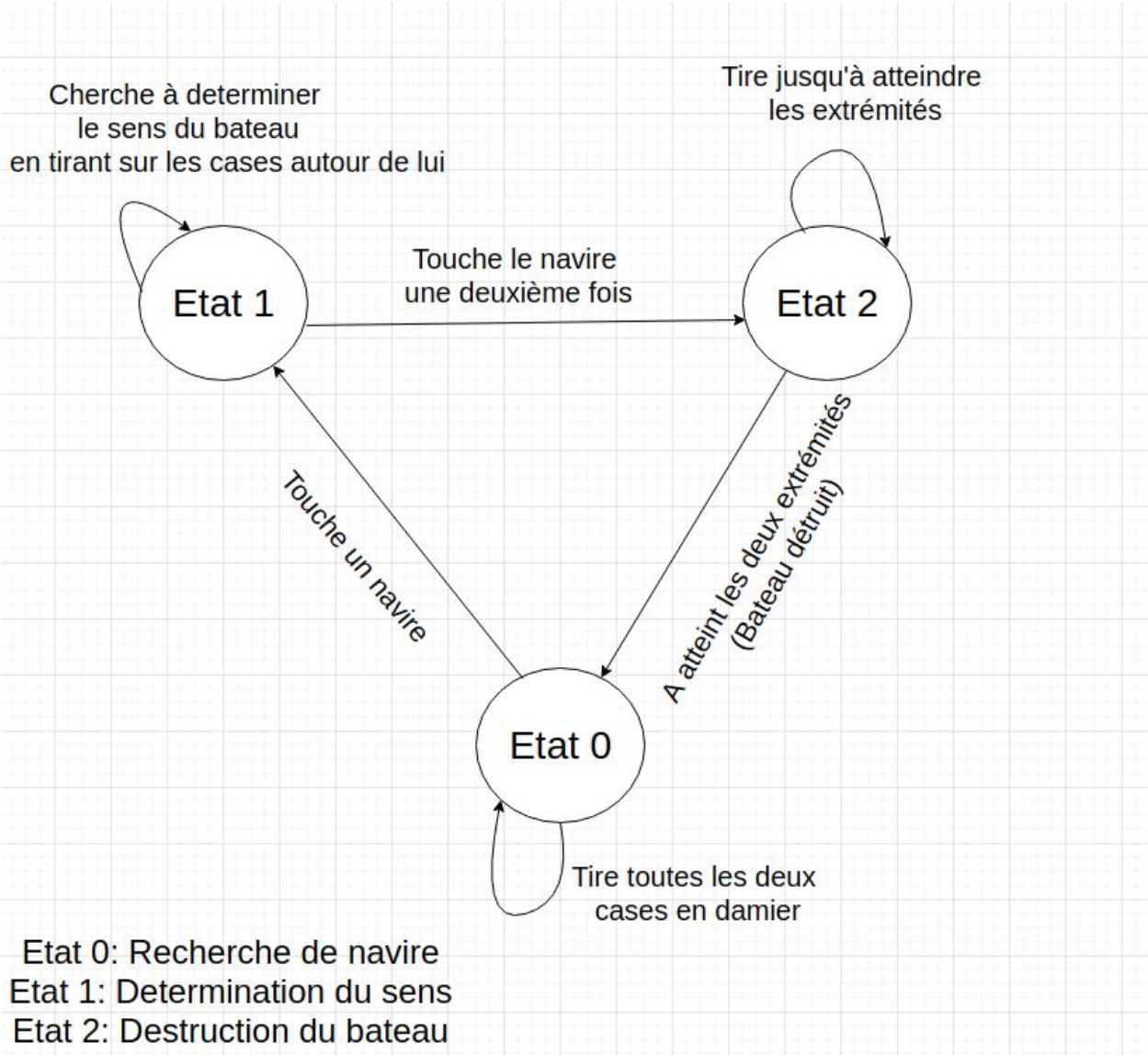
III) Interaction avec l'utilisateur

Lorsque la partie se lance, on demande à l'utilisateur son prénom, le nombre de lignes et de cologne du plateau. Contre qui il veut jouer, puis s'il souhaite disposer ses bateaux lui-même ou s' il désire le faire automatiquement.



On récupère toutes les interactions via le prompt.

IV) IA



De la même façon que l'automate de la consigne, notre IA possède 3 états.

L'état 0 correspond à la recherche d'un navire adverse. Dans cet état l'IA va tirer une case sur deux, à la manière d'un damier jusqu'à toucher un bateau. Si le

damier comporte des cases ayant déjà été tirés alors l'IA va simplement les sauter et continuer son damier

```
if(AI->state == 0){
    // Dans l'état 0, l'IA tire en damier en passant les cases sur lesquelles elle a déjà tiré
    while(alreadyHitten(AI->x, AI->y, ennemyPlat)) {
        AI->x += 2;
        if(AI->x >= 'a' + plat->nb_colonne) {
            if(AI->odd == 1) {
                AI->x = 'a';
                AI->odd = 0;
            }
            else {
                AI->x = 'b';
                AI->odd = 1;
            }
            AI->y++;
        }
    }
}
```

Lorsque l'IA a touché un bateau, elle passe à l'état 1.

Lors de l'état 1, l'IA va tenter de déterminer le sens du bateau en tirant tout autour de la case dans laquelle l'état 1 s'est déclenché. Elle va commencer par la case de droite (si elle n'a pas déjà été tirée et qu'elle est dans les limites du tableau sinon elle passera au sens suivant), puis par la case du bas, puis par la case de gauche et enfin celle du haut.

```
if(AI->state == 1) {
    if(!justChanged && hit) {
        // Si l'IA a trouvé le sens du bateau, elle passe à l'état 2.
        AI->state = 2;
    }
    else if(!justChanged && !hit) {
        // Si l'IA n'a pas trouvé le sens du bateau, elle continue en testant le prochain sens valide
        if(AI->dir == 'r') {
            if(isCorrectPos(AI->x, AI->y+1, plat)) {AI->dir = 'b';}
            else if(isCorrectPos(AI->x-1, AI->y, plat)) {AI->dir = 'l';}
            else if(isCorrectPos(AI->x, AI->y-1, plat)) {AI->dir = 't';}
        }
        else if(AI->dir == 'b') {
            if(isCorrectPos(AI->x-1, AI->y, plat)) {AI->dir = 'l';}
            else if(isCorrectPos(AI->x, AI->y-1, plat)) {AI->dir = 't';}
            else if(isCorrectPos(AI->x+1, AI->y, plat)) {AI->dir = 'r';}
        }
        else if(AI->dir == 'l') {
            if(isCorrectPos(AI->x, AI->y-1, plat)) {AI->dir = 't';}
            else if(isCorrectPos(AI->x+1, AI->y, plat)) {AI->dir = 'r';}
            else if(isCorrectPos(AI->x, AI->y+1, plat)) {AI->dir = 'b';}
        }
        else if(AI->dir == 't') {
            if(isCorrectPos(AI->x+1, AI->y, plat)) {AI->dir = 'r';}
            else if(isCorrectPos(AI->x, AI->y+1, plat)) {AI->dir = 'b';}
            else if(isCorrectPos(AI->x-1, AI->y, plat)) {AI->dir = 'l';}
        }
        AI->xBis = 0;
        AI->yBis = 0;
    }
```

Lorsqu'elle a touché une deuxième fois le bateau (et donc ayant réussi à déterminer son sens de rotation) elle passe à l'état 2.

Dans l'état 2 elle continue à tirer dans le sens du bateau jusqu'à atteindre la première extrémité (Case ne contenant pas le bateau ou hors du plateau). Une fois trouvé, elle revient à la première case trouvée du bateau (celle ayant permis de passer de l'état 0 à l'état 1) puis elle continue dans le sens opposé pour trouver la deuxième extrémité. Une fois cette deuxième extrémité atteinte, on considère le bateau comme détruit.

On s'assure à chaque étape de l'état 2 de ne pas atteindre une case hors du plateau ou une case sur laquelle un tir a déjà été effectué.

V) Comment compiler

Pour compiler nous utilisons un Makefile.

Si vous voulez juste jouer et profiter pleinement du jeu faites juste:

'make && ./BatailleNavale'

SI vous souhaitez l'utiliser avec Valgrind ou debugger faites:

'make DEBUG && ./BatailleNavaleDEBUG'

Pour obtenir le projet en archive:

'make zip'

Pour nettoyer le répertoire du projet tout en conservant l'exécutable:

'make clean'

Pour nettoyer le projet en enlevant l'exécutable:

‘make mrproper’

V) Conclusion

Ce projet nous a appris à développer d’une façon plus modulable avec notamment l’utilisation des pointeurs de fonction. Étant donné que la taille du plateau varie, que l'ennemi peut être un autre joueur ou une IA et que les tirs spéciaux dépendent de certaines conditions. Nous avons donc essayé du mieux possible de programmer de façon à ce qu’on puisse changer le comportement du programme en changeant le code le moins possible.

Nous avons effectué un DUT et nous avons commencé le C cette année. Ce projet nous a donc offert la possibilité d’explorer différentes spécificités du langage et sommes donc maintenant beaucoup plus à l’aise avec ce langage.

Nous avons utilisé Git, Google Doc, Trello et DIscord comme outils collaboratifs.

Nous sommes globalement satisfaits de ce projet que nous pensons avoir plutôt bien réussi, nous avons aussi réalisé certains des bonus (Le joueur peut placer ses bateaux, la taille de la grille peut être modifiée et le joueur peut choisir de jouer contre l’IA ou un autre joueur en local). Le thème était très divertissant et ce fut un plaisir pour nous deux de travailler sur ce projet.