Rapport d’avancement

Blocus

Table des matières :

Introduction du sujet :

Le blocus se présente sous la forme suivante :

Tout d'abord, notre jeu est assez original car ce n'est pas un blocus banal, car nos personnages qui s'affrontent sont en réalité un personnage qui représente le logo de Linux et un autre qui représente le logo de Windows, nous avons fait ce choix pour garder un peu d'humour et nous pensons que cela plaira ne serait-ce qu'un petit peu à notre professeur :).

En exécutant le programme, nous arrivons sur une page de menu, où les choix qui s'offrent à nous sont multiples : choisir le mode de jeu (joueur contre joueur ou PvsP et joueur contre robot donc PvsBot), ou quitter, le bouton quitter permettant donc de fermer le programme. En choisissant le mode de jeu, nous arrivons ensuite sur une page qui permet de choisir la taille de notre grille, avec des boutons cliquables allant de 3 à 9.

Après avoir choisi la taille de notre grille, cette grille s'initialise d'un point de vue numérique (qu'on ne voit pas à l'écran), mais une grille se dessine aussi devant les yeux de l'utilisateur, et il est prêt à jouer. La partie de jeu se présente comme cela :

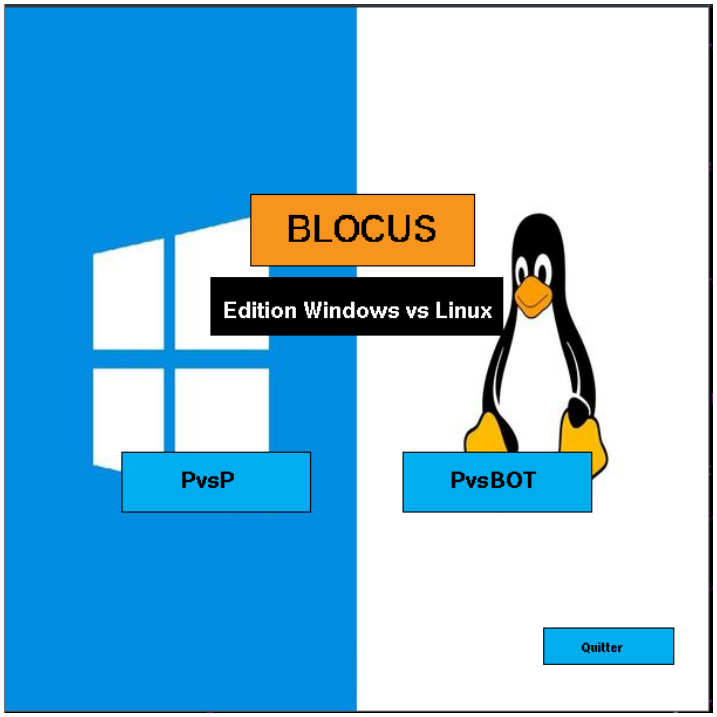
Chaque joueur choisit l'un après l'autre la case où il souhaite se placer sachant qu'ils ne peuvent pas se mettre sur la même case. Ensuite la partie commence. Le premier joueur qui est dans notre cas à nous le joueur Linux commence donc par déplacer son pion, puis il place une croix dans la grille en essayant de bloquer son adversaire, puis ce dernier (windows) fait la même chose (déplace son pion puis place une croix).

Quand un joueur ne peut plus se déplacer, il perd donc, et un écran de fin avec le gagnant s'affiche à l'écran, avec les deux choix possibles qui sont de quitter où de rejouer.

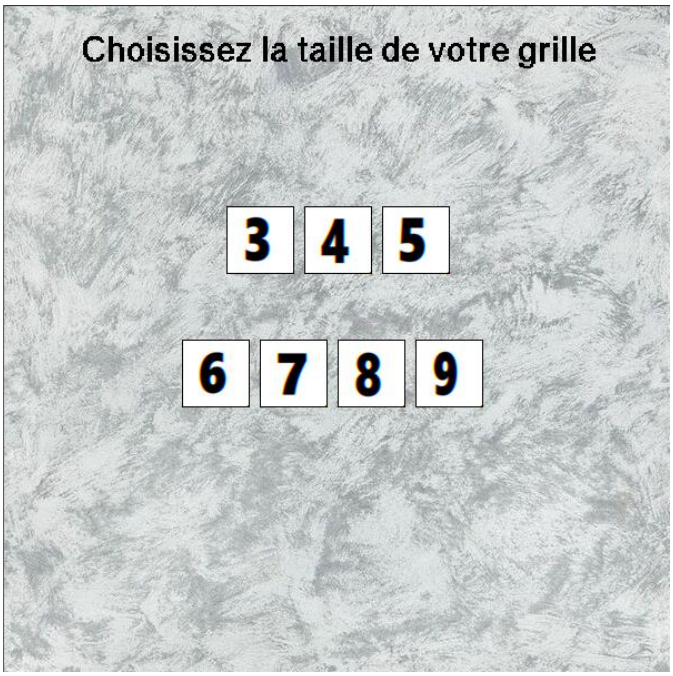
Ce projet est donc un projet réalisé en C89 avec comme seules aides la bibliothèque graphique et la bibliothèque standard du C.

Fonctionnalités du programme :

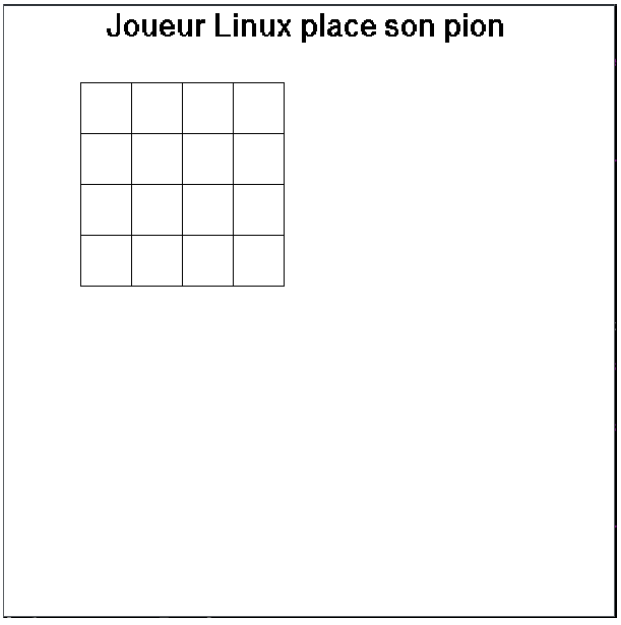
Je vais maintenant développer les parties du programme dites dans l'introduction



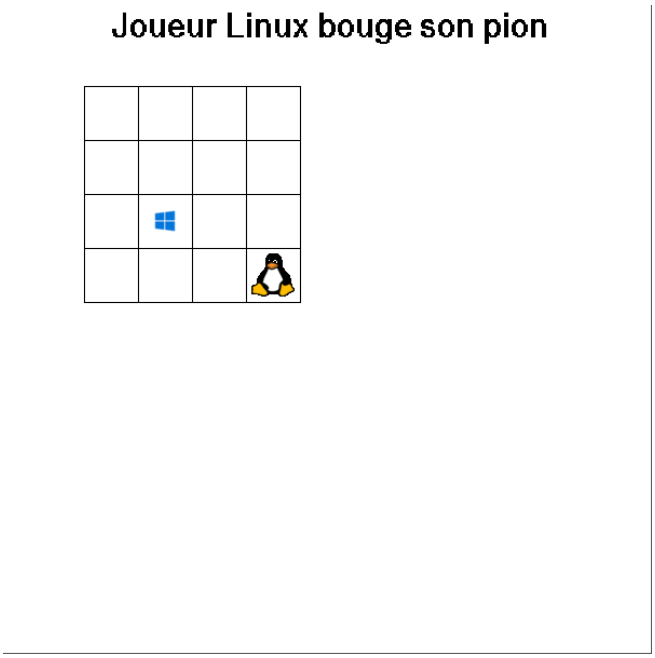
On arrive en exécutant le programme sur cette page de menu, ou nous pouvons donc choisir de quitter (ce qui ferme la fenêtre et quitte directement le programme), ou choisir notre mode de jeu.



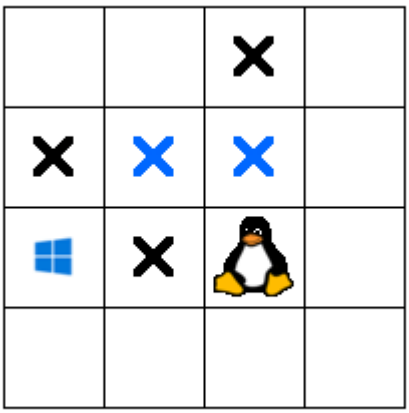
En fonction de sur quel bouton vous cliquez, cela vous emmènera sur une fonction différente, mais ces deux fonctions commencent par cette page, qui permet de choisir la taille de la grille, disons que pour l'exemple nous choisissons une taille de 4.



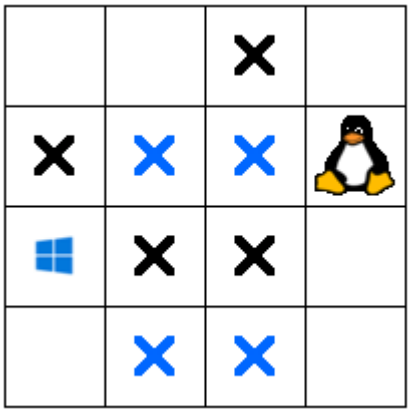
Cette page s'affiche ensuite avec en haut une phrase qui dit quelle action pour quel joueur effectuer



Les deux joueurs placent leurs pions, puis la partie peut commencer.

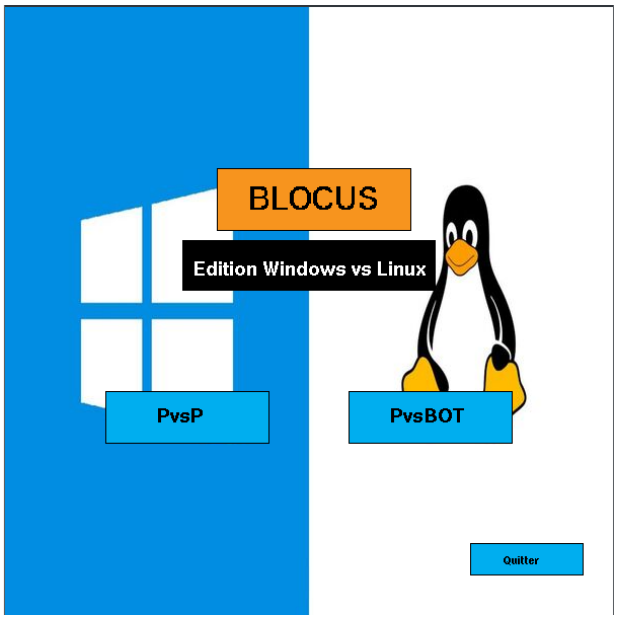


La bataille est rude, mais on voit que le joueur linux domine la partie.

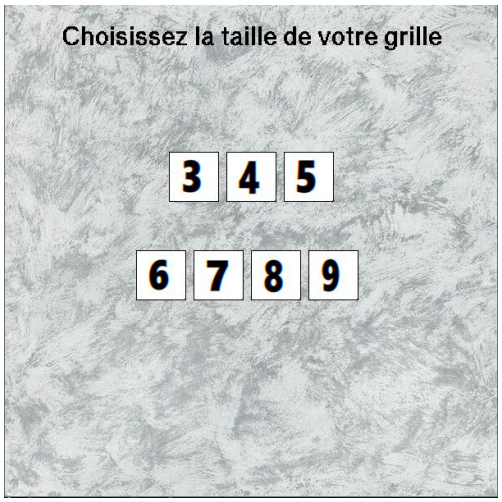


Le joueur windows va perdre cette partie, il est déjà bloqué dans un coin, il n'a plus d'espoir...

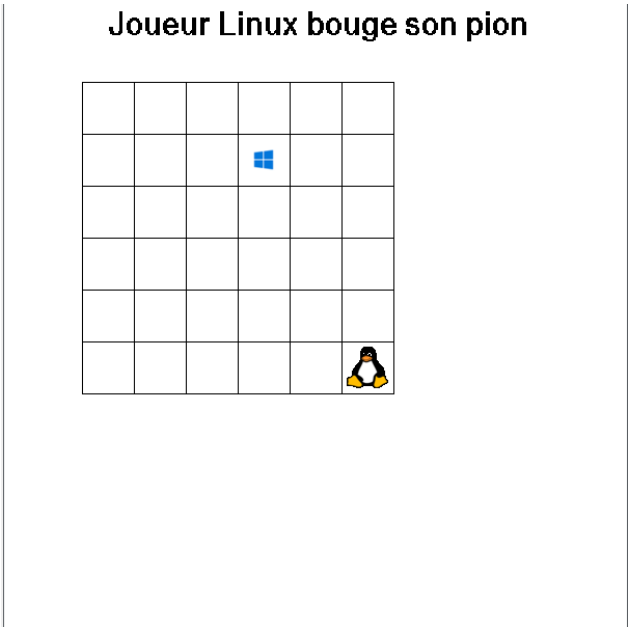
Et voilà, la partie est donc finie, le joueur linux a gagné et soit vous choisissez de rejouer soit vous quittez le jeu.



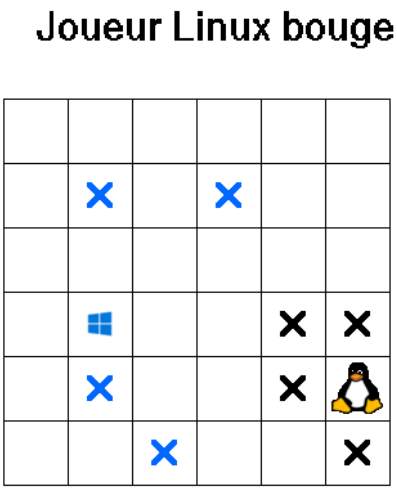
Maintenant rejouons mais en mode joueur contre ia.



Choisissons cette fois une grille de taille 6 pour l'exemple



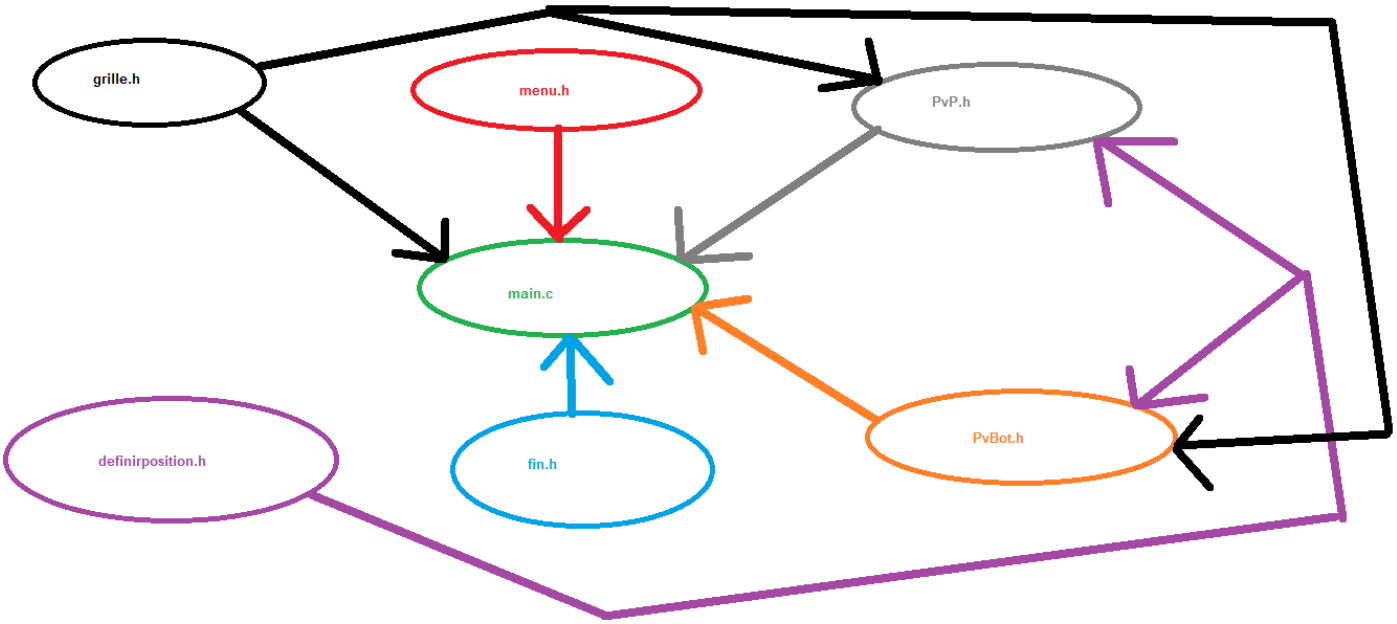
Nous faisons le placement des pions, puis vous connaissez la suite.



Pour l'exemple je vais faire exprès de perdre.

Ce petit message s’affiche quand l’IA gagne.

Structure du programme :



Comme il n'y a pas énormément de dépendances de fichiers, j'ai choisi de toutes les mettre dans un seul schéma.

Definirposition.h contient seulement les fonctions liées à la position du joueur c'est pourquoi on n'en a besoin seulement dans PvBot.h et dans PvP.h.

Grille.h contient des fonctions liées à la grille, comme le fait de dessiner la grille, ce qui est donc nécessaire pour PvP.h et PvBot.h.

Puis nous faisons tous nos appels de fonctions dans le main.c donc tous les fichiers .h sauf definirposition.h y sont reliés.

Données représentant une partie en cours :

Tout d'abord, dans le menu, cliquer sur un bouton pour chaque mode nous renvoie un int qui est le code de sortie du mode, qu'on retourne dans le main. En suite, en fonction de se code, on fait soit appel à la fonction pvp ou à la fonction pvbot. Au début de la fonction on choisit la taille de la grille, à laquelle on ajoute 2 pour pouvoir définir toutes les bordures à 9, ce qui nous permet de simplifier la condition de victoire. Cette grille est allouée dynamiquement pour optimiser l'utilisation de la mémoire et car c'est mieux adapté à nos besoins. Nous envoyons en suite la taille (à laquelle on a refait -2 pour la retourner à la normale) à la fonction pour déssiner la grille de jeu et la partie commence. Quand l'utilisateur clique on vérifie tout d'abord s'il clique bien dans notre grille puis nous assignons les images de personnages à cet endroit, mais nous mettons aussi dans notre tableau multidimensionnel la valeur associée au personnage (8 pour linux, 7 pour windows). Après, nous mettons la vérification de victoire au début de chaque tour car il ne faut pas laisser le personnage bouger si il a déjà perdu. Et donc nous récupérons les anciennes positions des personnages pour pouvoir les effacer derrière eux au moment du déplacement. Nous limitons aussi leurs déplacements aux cases adjacentes en vérifiant simplement si l'utilisateur a cliqué sur une case qui est autour du pion du joueur. Pour vérifier la victoire nous regardons seulement si toutes les cases autour du joueur sont différentes de 0 (0 étant une case libre) et nous retournons 1, 2 ou 3 (1 pour victoire de windows, 2 pour victoire de linux et 3 pour victoire de l'IA). Et quand la condition de victoire est trouvée nous affichons donc l'écran de fin, avec le victorieux ou nous retournons un int pour savoir si l'utilisateur veut rejouer.

Conclusion du rapport :

Kouami KPEGLO :

Au début de ce projet je le voyais comme une énorme épreuve à surmonter car il m'a fait réaliser à quelle point mes compétences sont encore limité, j'ai surtout constaté cela à travers mon binôme qui contrairement à moi s'en sortait beaucoup mieux que moi, que ce soit niveau compréhension, écriture de code, j'avais l'impression de ne vraiment pas servir à grand-chose au début. Mais plus le temps à passé plus j'ai compris que je n'avais pas besoin d'autant douté de mes compétences car je peux toujours parfaire mes compétences, c'est ce que travailler sur ce projet avec mon binôme m'a appris. Ce projet représente pour nous la somme de nos efforts cumulé, et pour moi personnellement une preuve de du fait que je peux encore me surpasser.

Rayan BISSON :

J'ai éprouvé de grandes difficultés tout au long de ce projet, comme des moments où j'avais l'impression que mon cerveau était en surchauffe, où même une grande quantité de stress en me demandant si je finirais à temps, au bout d'un moment je me suis même demandé si j'étais bien fait pour l'informatique. Je crois que j'avais tout simplement sous-estimé cette SAé et peut-être même le BUT Informatique. La vérité est que je n'avais pas un bon rythme de travail et un projet comme celui-ci m'a fait réaliser mes erreurs, j'étais à la limite des larmes aux yeux quand j'avais enfin terminé et résolu le dernier problème. Malgré tout cela, je me rends maintenant enfin compte de l'envergure des études supérieures et je suis maintenant motivé plus que jamais à continuer de coder, et à continuer en corrigeant les lacunes que j'avais et dont je me suis rendu compte pendant ce projet. Merci pour cette réalisation.