Présenté par Céline Azevedo et Dylan Girault (Groupe 5)

RAPPORT DU JEU BLOCUS

Année universitaire 2020-2021

Table des matières

Introduction 2
Déroulement du Jeu 2
Structure du programme 3
Explication des données 4
Fonctionnalité du programme 4
Pendant une partie 7
Conclusion 17

BLOCUS

Introduction

Pour le projet tutoré, il nous a été demandé de réaliser un jeu nommé « Blocus » en langage C sans emprunt extérieur hormis celui de la bibliothèque graphique de l'IUT.

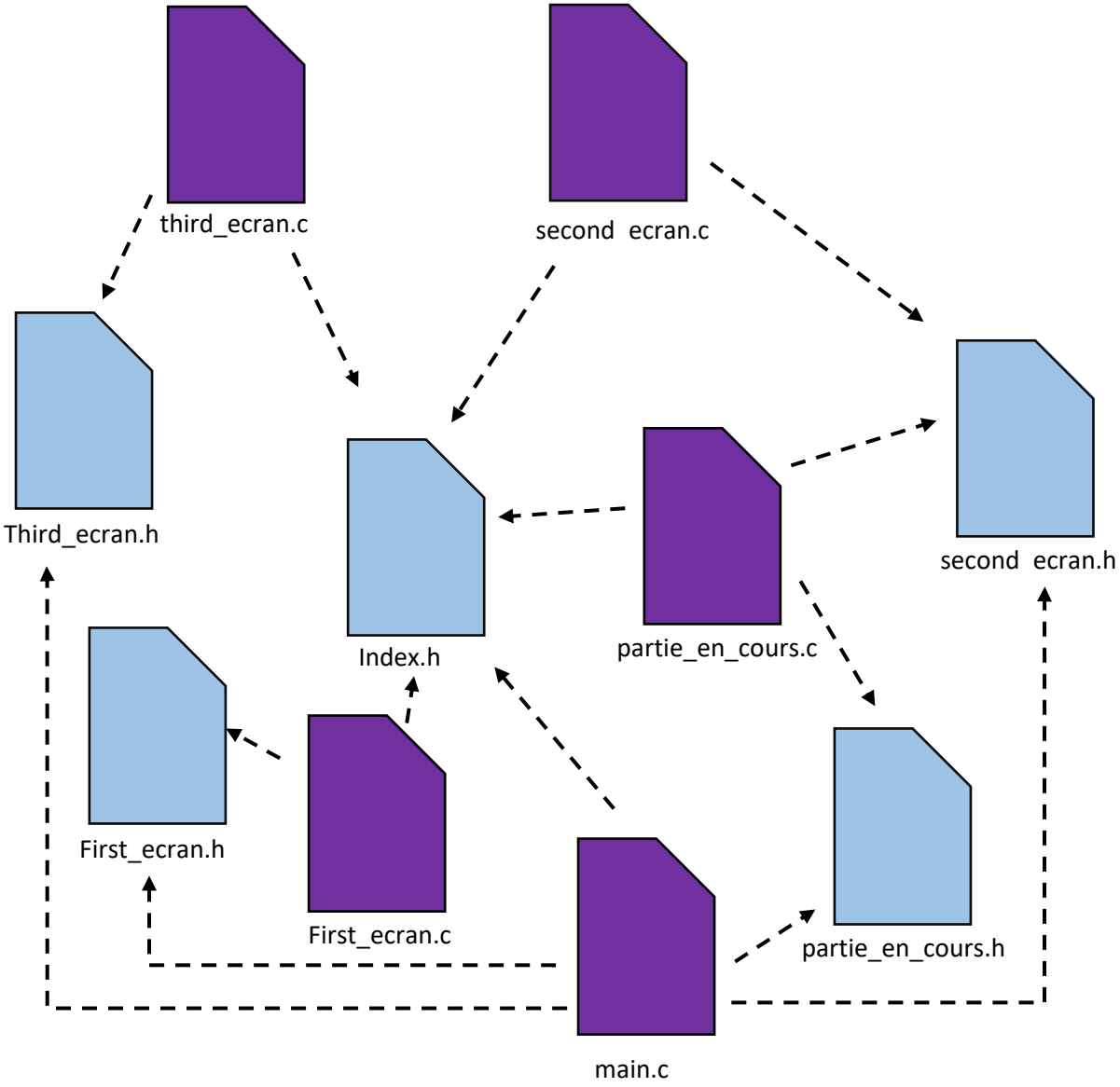
Déroulement du Jeu

Le jeu commence par une page d'accueil, où le joueur peut choisir entre jouer contre un BOT ou une autre personne, dans ce dernier cas, les deux joueurs doivent se partager la même souris.
Il choisit aussi, dans ce même écran, la taille de la grille carrée du jeu, entre 3 et 9.

Après sa sélection, un deuxième écran avec la grille choisie apparaît. Les deux joueurs choisissent une case où placer leur pion (les deux pions ne peuvent pas partager la même case).
Ceci fait, la partie et le premier tour commencent. Durant le tour d'un joueur, il doit déplacer son pion vers une case adjacente (y compris en diagonale), puis choisir une case libre qui sera condamnée. Les joueurs alternent leur tour.
Le premier joueur qui ne peut pas déplacer son pion (car toutes les cases adjacentes sont occupées) a perdu.

Un troisième écran se lance alors affichant le gagnant et la possibilité de quitter ou de rejouer.

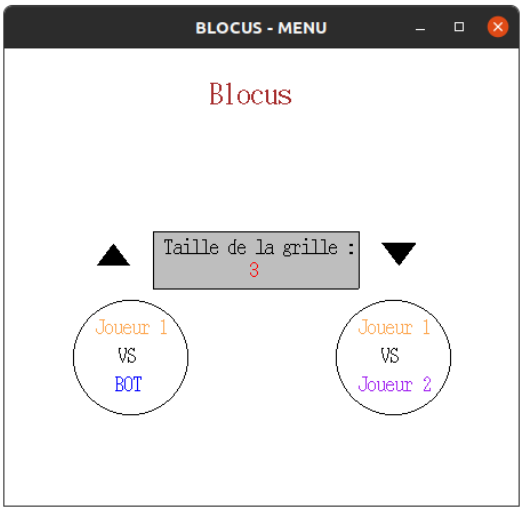
Structure du programme



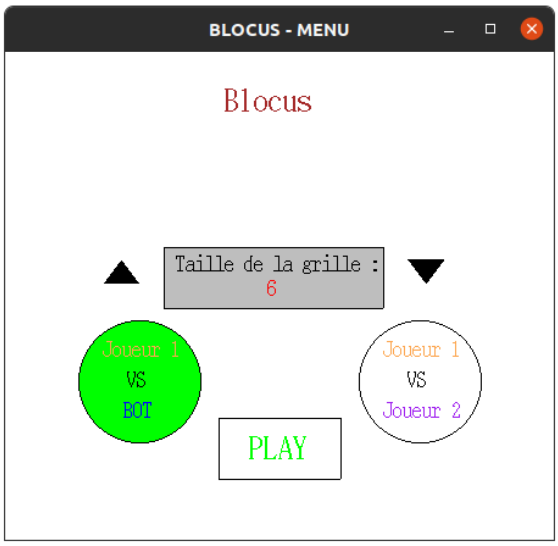
Explication des données

Fonctionnalité du programme

L'utilisateur peut augmenter ou diminuer la taille de la grille qui sera générée dans le jeu grâce aux flèches qui sont autour de l'encadré, situé dans le premier écran, qui indique la taille actuelle de la grille.
Au lancement du programme, la taille de la grille est déterminée à 3.



Par la suite, l'utilisateur peut sélectionner soit la case circulaire gauche permettant de joueur contre l'ordinateur « Joueur contre BOT », soit celle de droite permettant de joueur avec une autre personne « Joueur contre Joueur ». Sélectionner l'une des options fait apparaître le bouton « PLAY » permettant de lancer la partie.



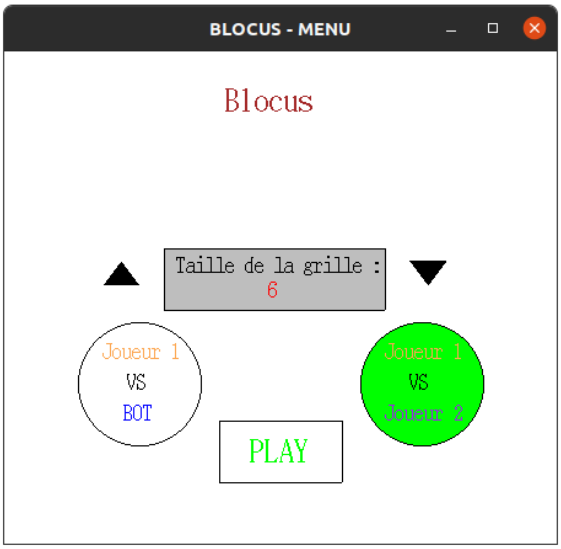


Diagramme représentatif des choix de modes :

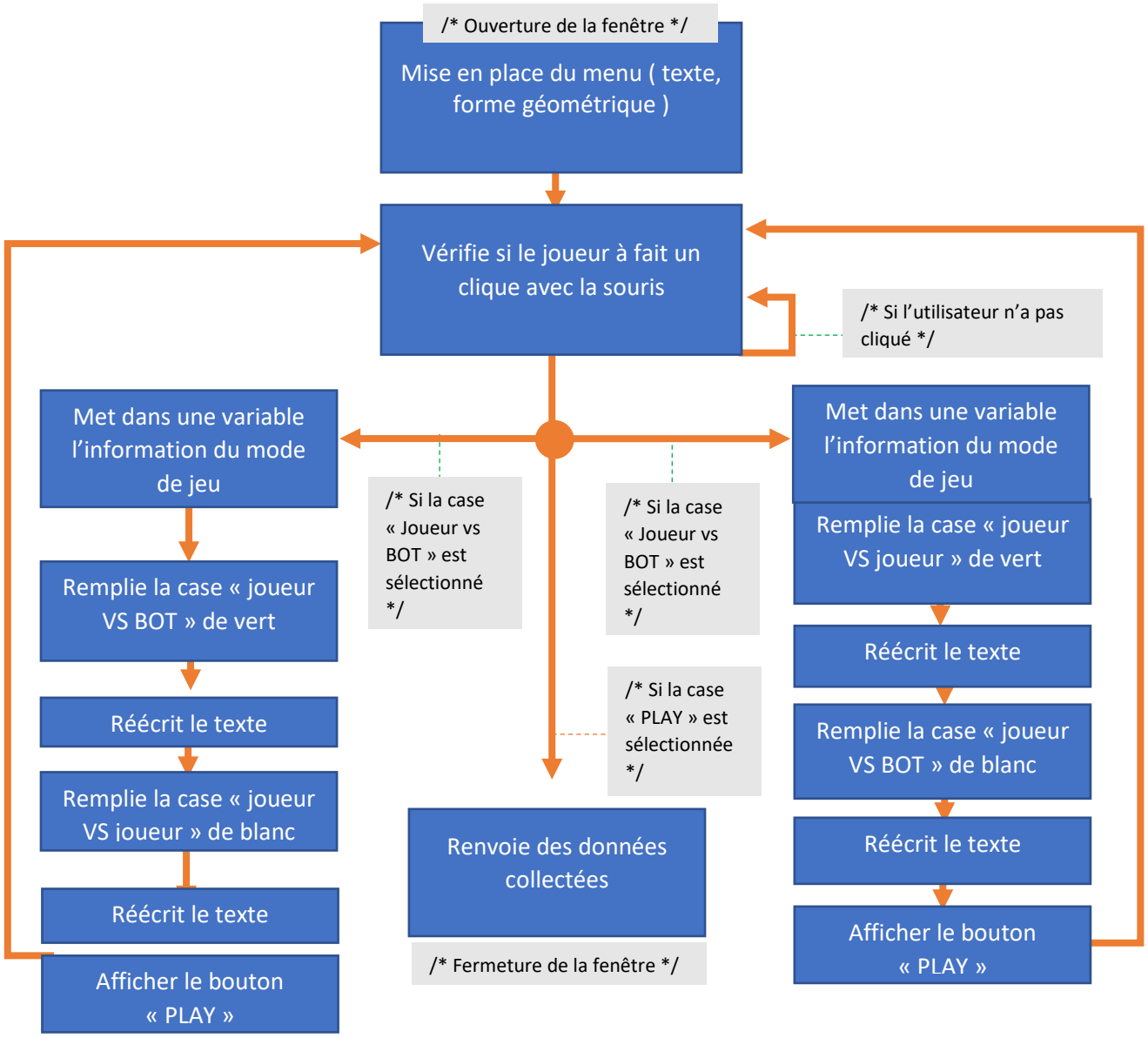
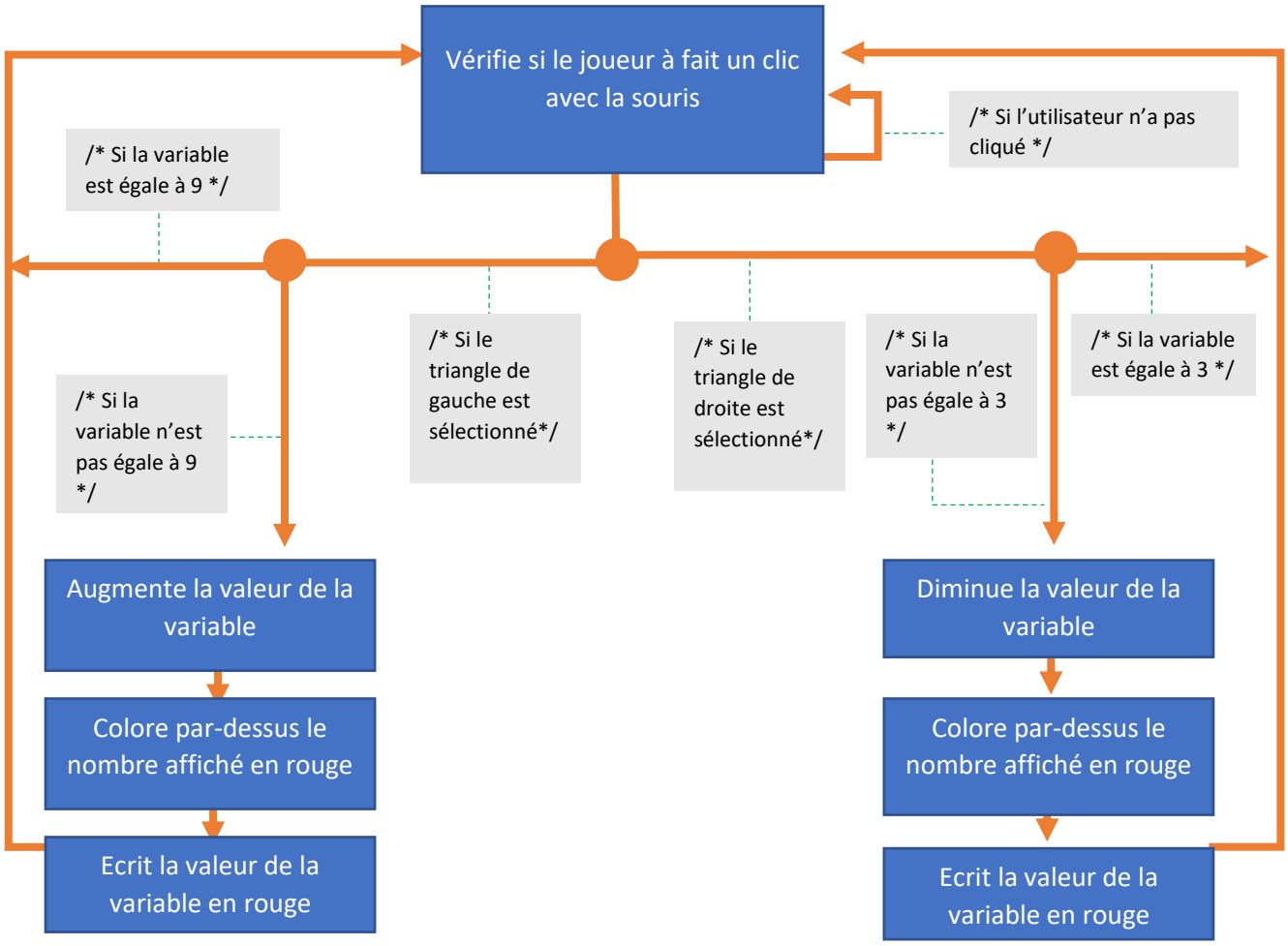


Diagramme représentatif du choix de la taille de la grille :

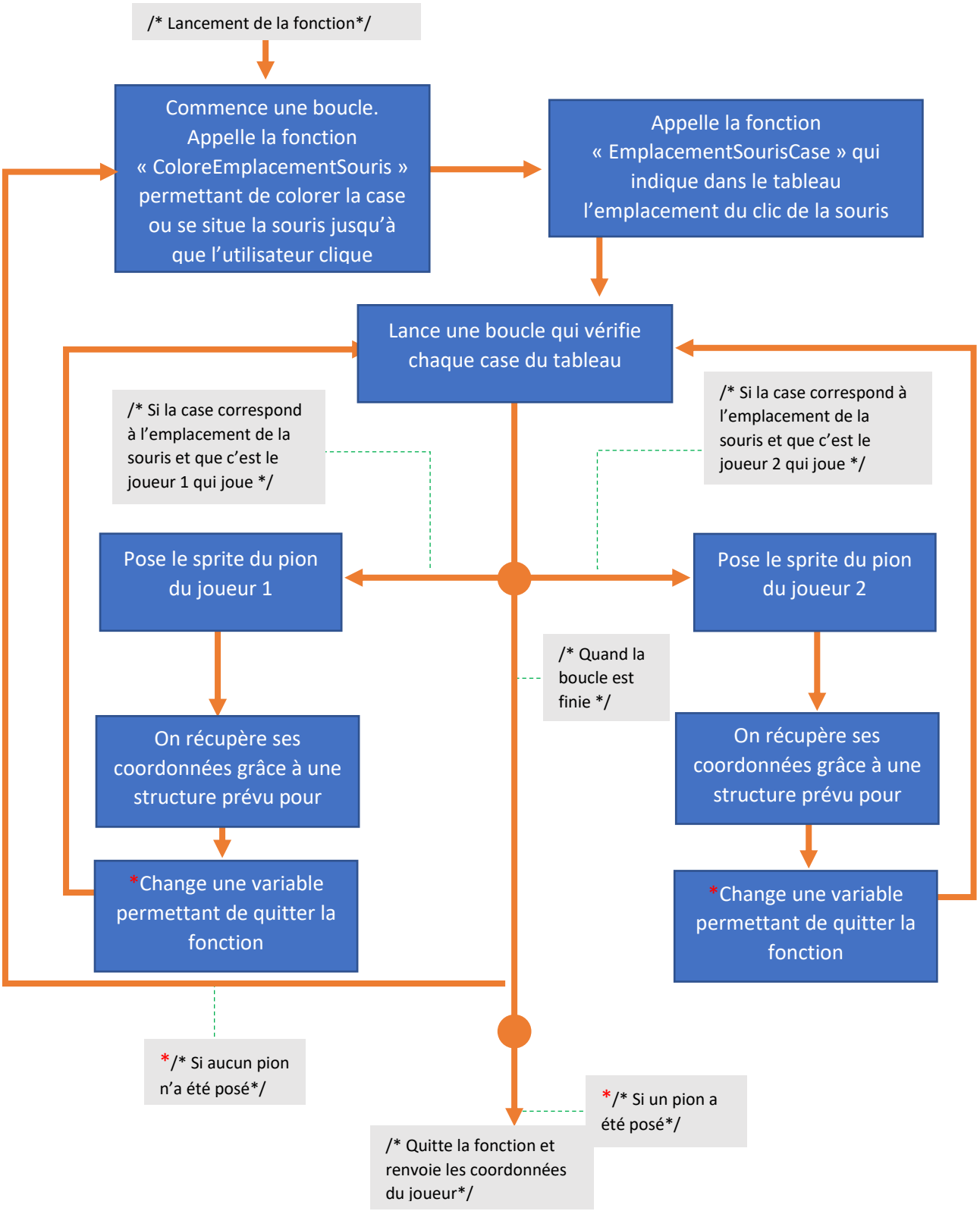


Après fermeture de la fenêtre, on crée un tableau carré correspondant à la taille de la grille. Chaque case du tableau représente une case de la grille, qui contient des valeurs symbolisant l'emplacement des pions et des cases condamnés.

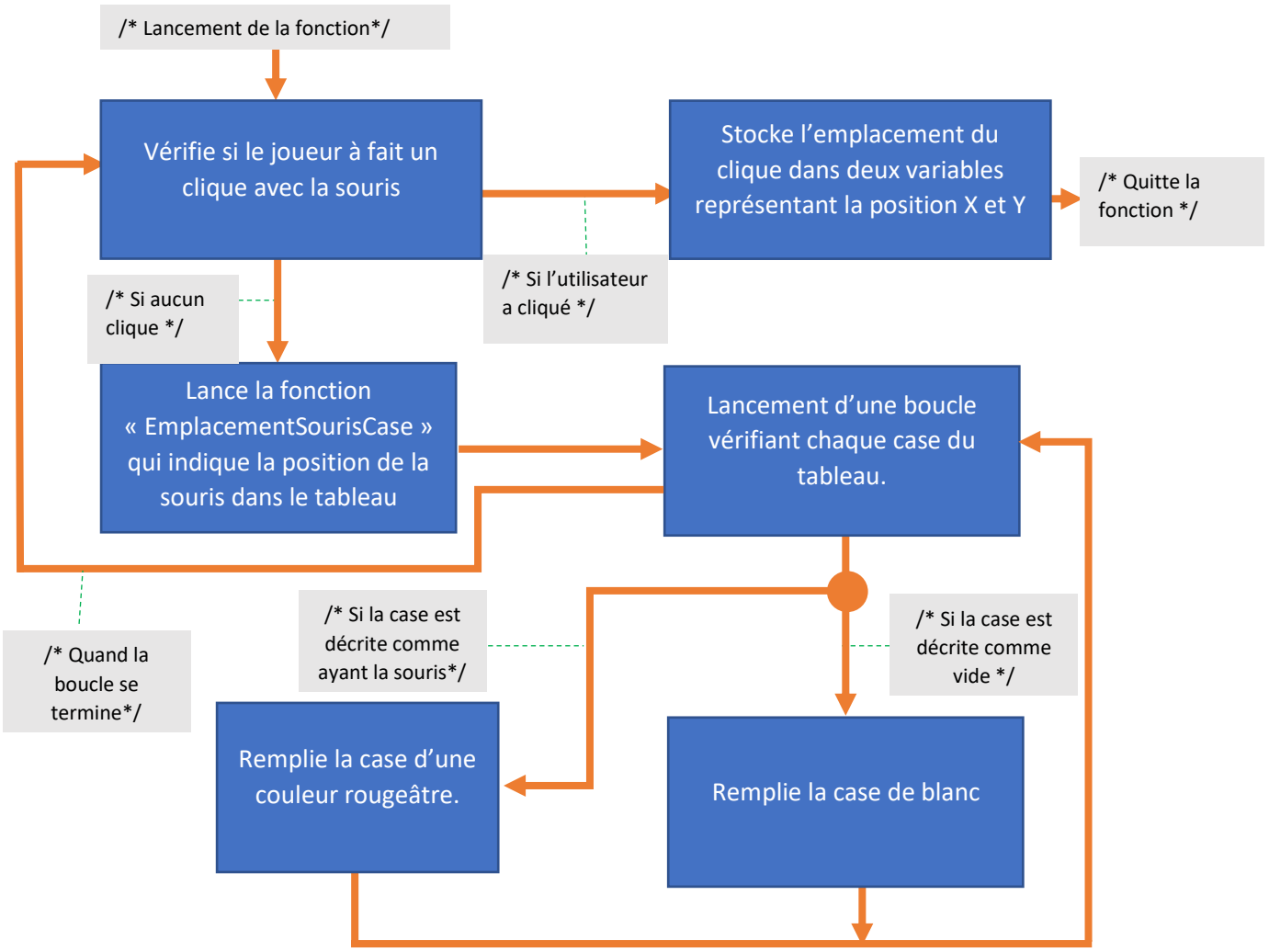
On ouvre une nouvelle fenêtre et crée la grille graphiquement à l'aide de constantes définies dans un fichier accessible à tous. Ces constantes définissent le nombre de pixels séparant une case d'une autre et l'épaisseur des cases.

Pendant une partie

Une fois la grille formée, on lance la fonction « TourJoueur » (qui appelle deux autres fonctions) permettant de commencer la partie en donnant en argument un tableau, la structure « option » et la constante permettant d'identifier à quel joueur correspond le tour en cours. La fonction renvoie la position du joueur. Il existe également une structure pour stocker des coordonnées (x et y) appelées « emplacement ».



/* Fonction « **ColoreEmplacementSouris** » qui colore la case où se situe la souris */



/* Fonction « **EmplacementSourisCase** » qui met l'emplacement de la souris dans le tableau */

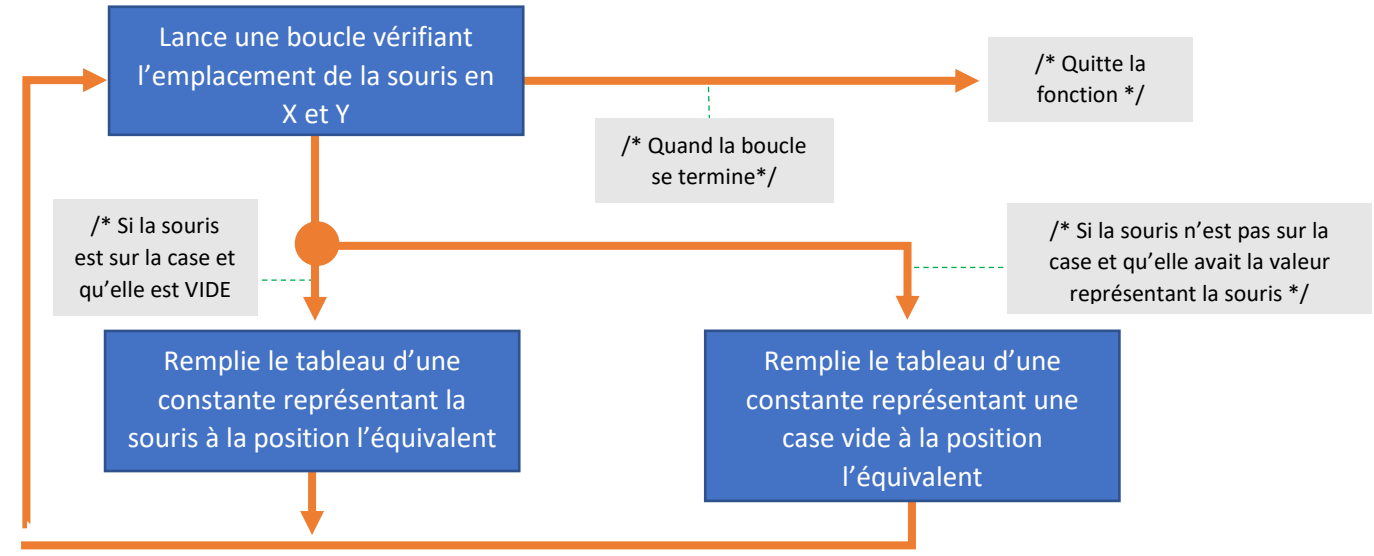
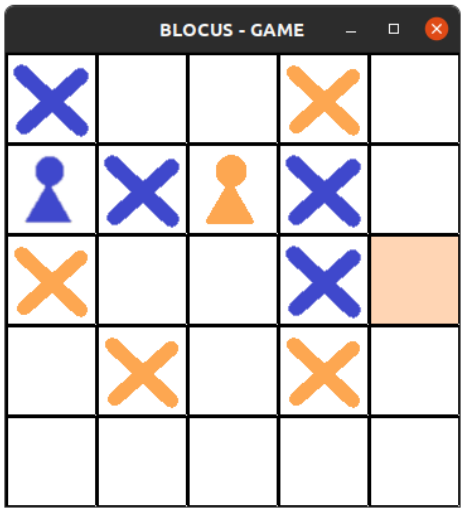
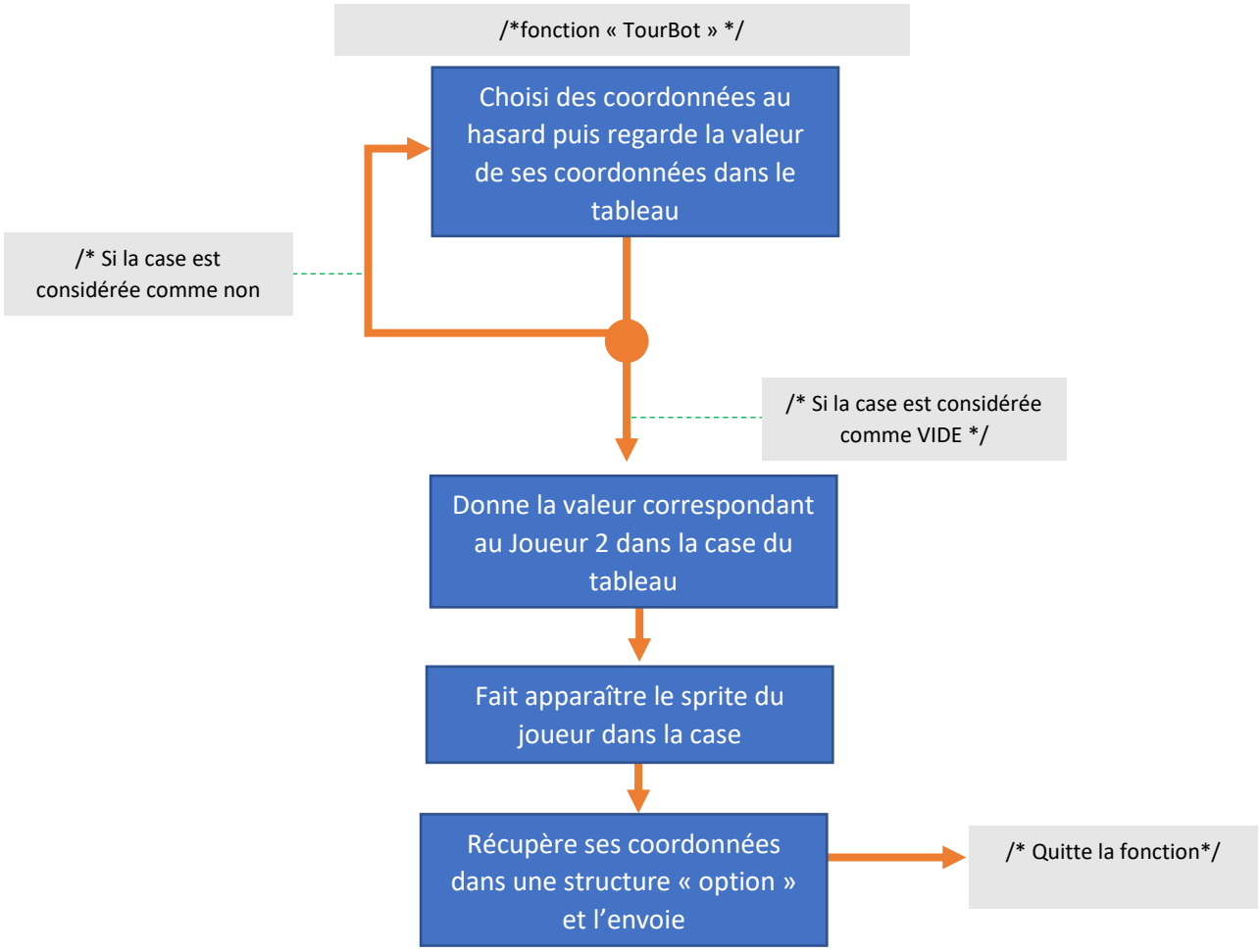


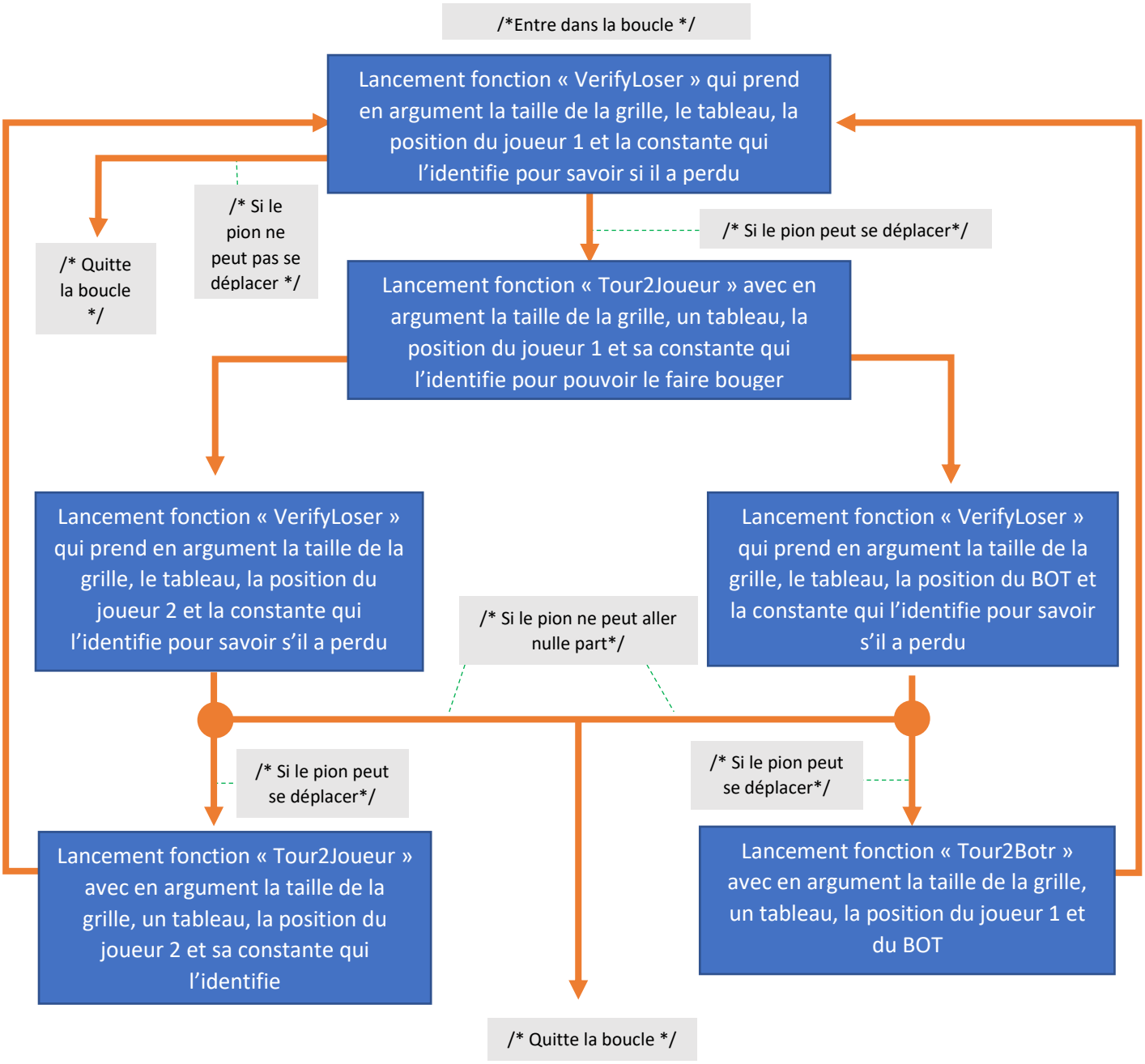
Image du jeu dont on peut voir une case coloré représentant l'emplacement de la souris :



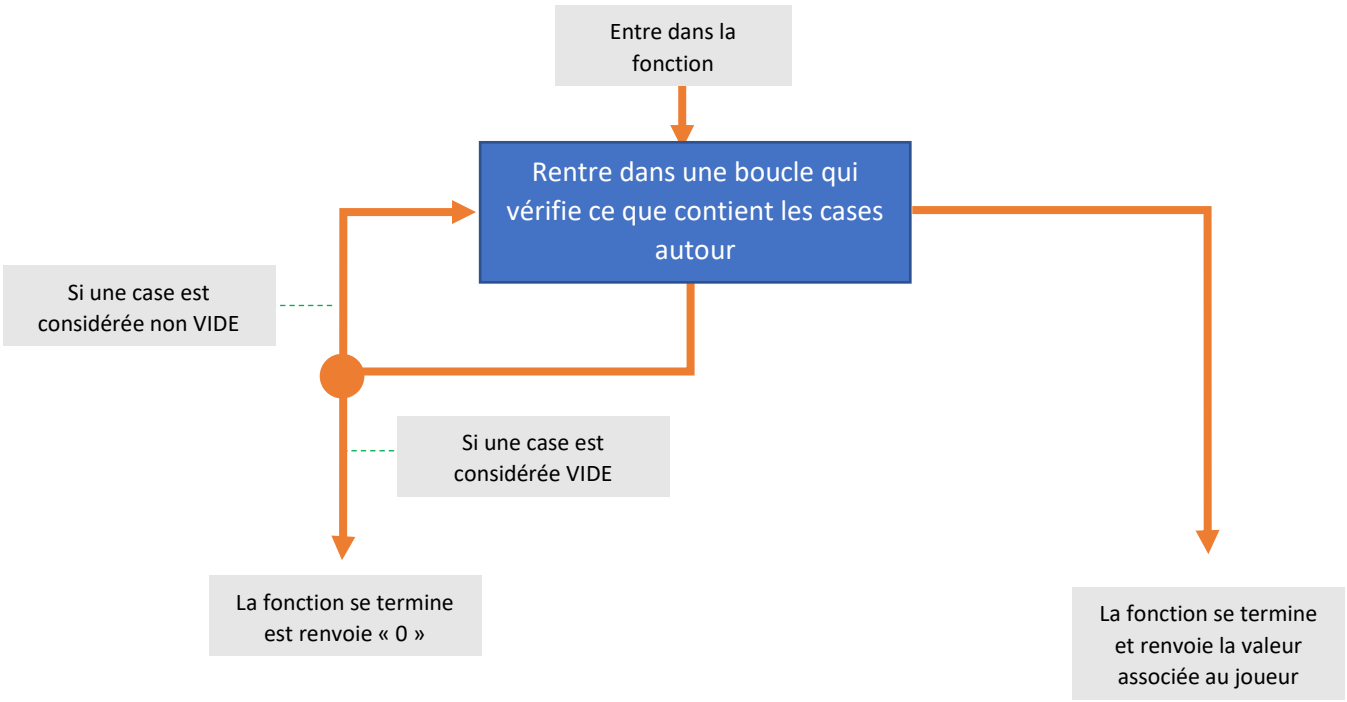
Ensuite selon le mode de jeu choisi, on entrera dans une boucle qui lance de nouveau la fonction « TourJoueur » avec comme identification qu'il s'agisse du joueur 2 si le mode choisi est « Joueur VS Joueur ». Or si le mode est « Joueur VS BOT », cela lancera la fonction « TourBot » qui prend en argument une structure « option » et un tableau.



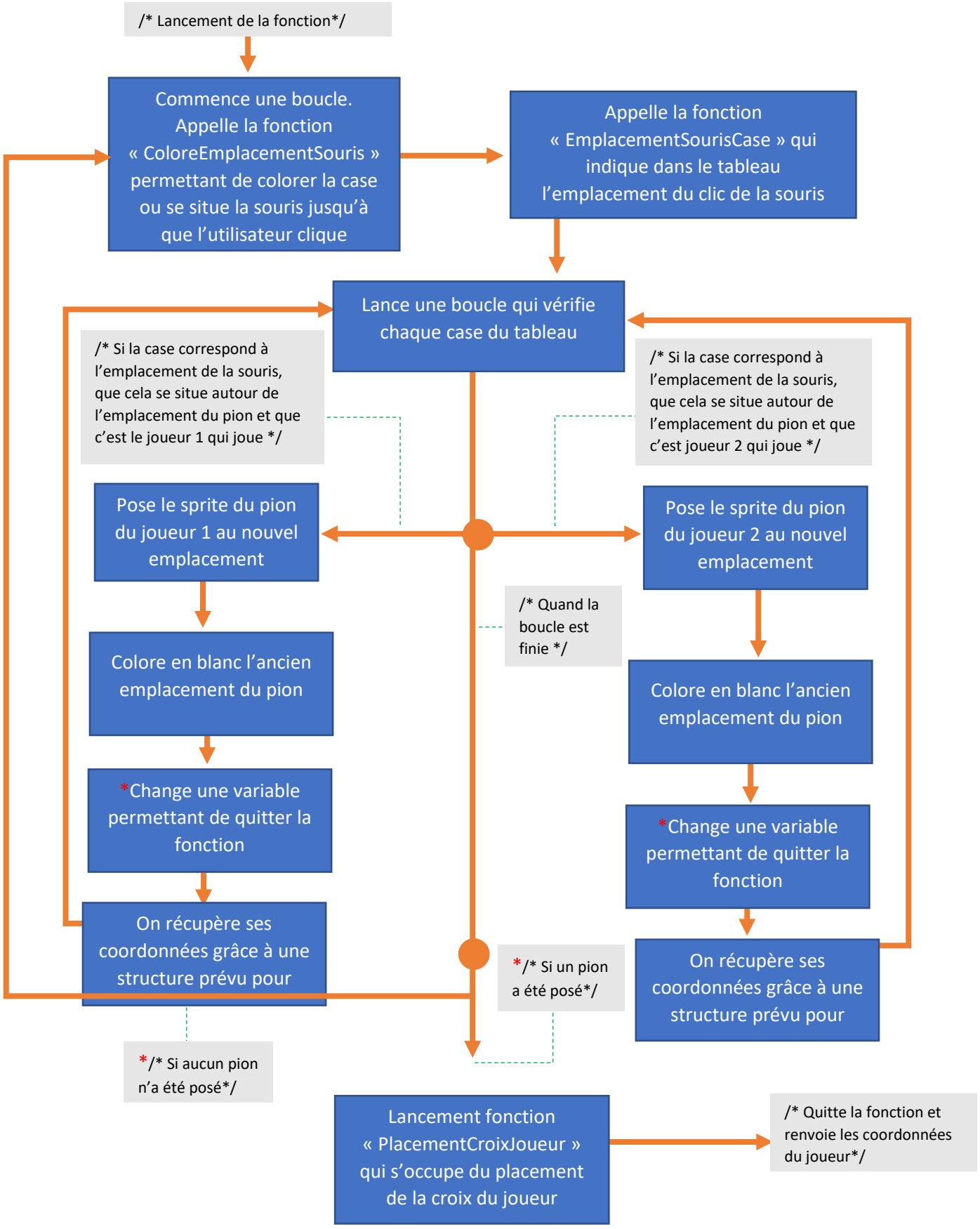
Après que les deux pions soient posés, on entre dans une boucle qui va lancer des fonctions. La première concerne le déplacement du premier joueur, la deuxième vérifie si le joueur n'a pas perdu et la troisième, qui est, selon le mode de jeu, le déplacement du joueur 2 ou du BOT.



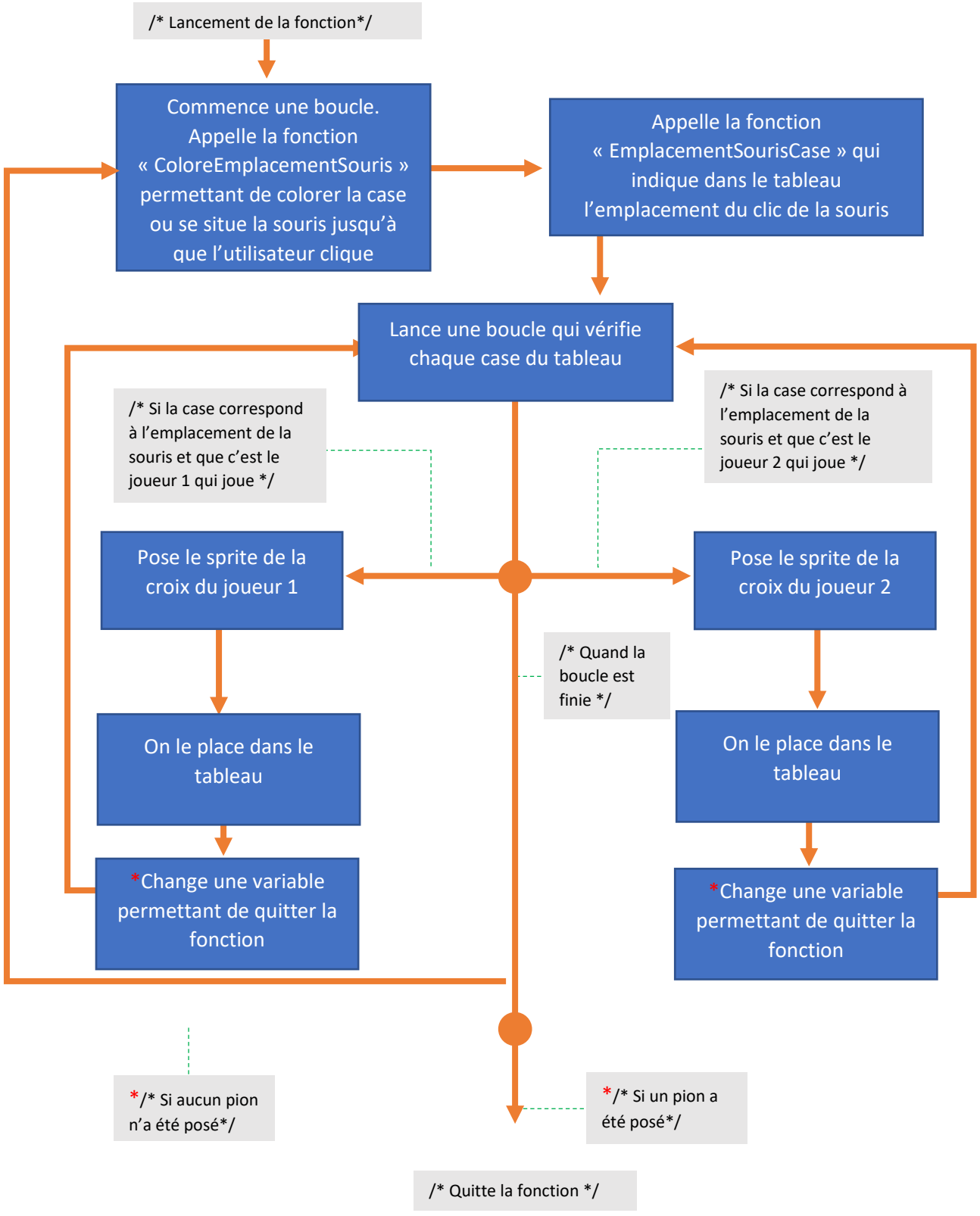
/* Fonction « **VerifyLoser** » qui vérifie si le pion mis en argument perd en renvoyant la constante correspondant au joueur s'il perd*/



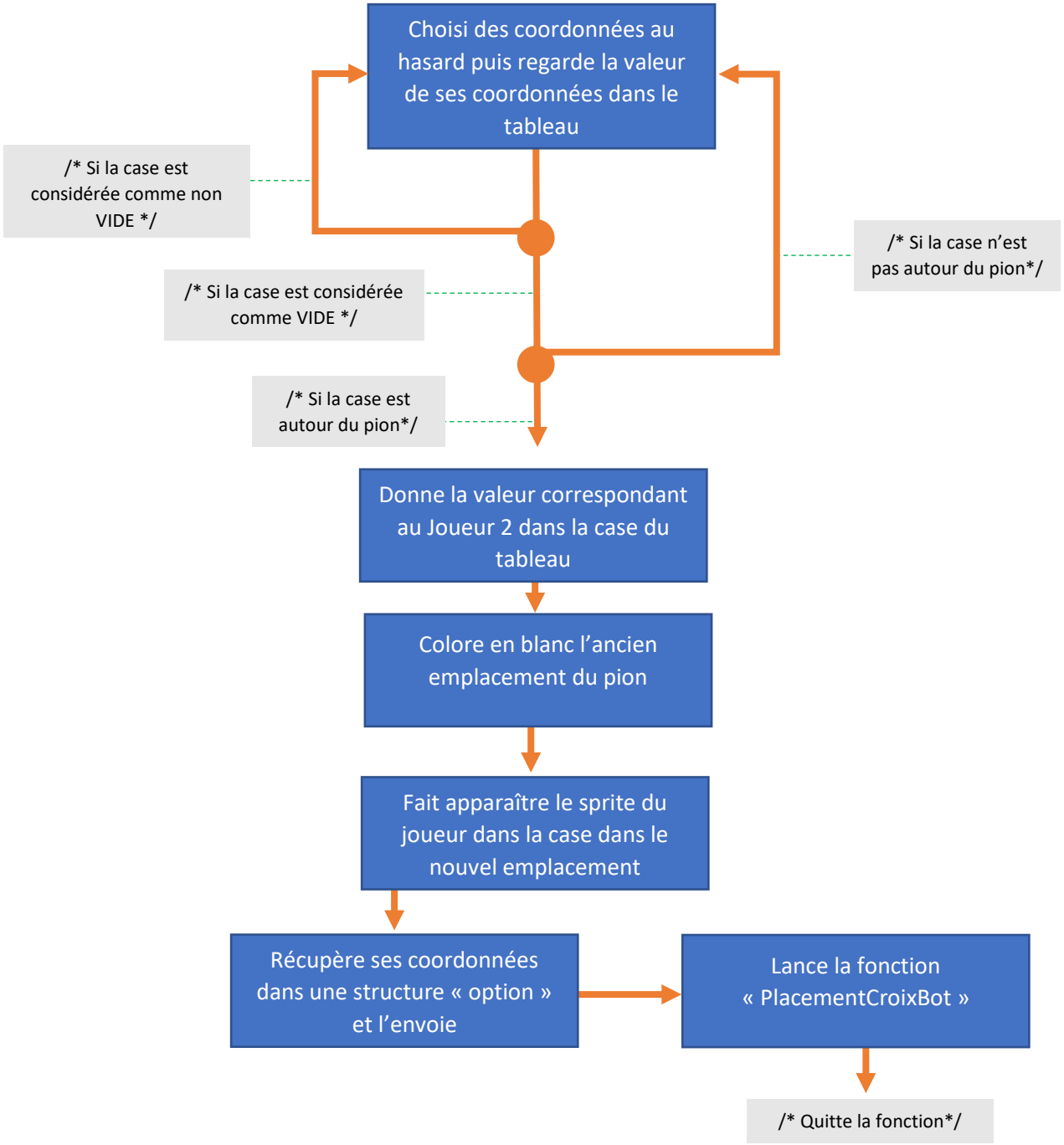
/* Fonction « **Tour2Joueur »** semblable à la fonction TourJoueur en permettant en plus de pouvoir condamner une case mais permet uniquement de se déplacer autour du pion */



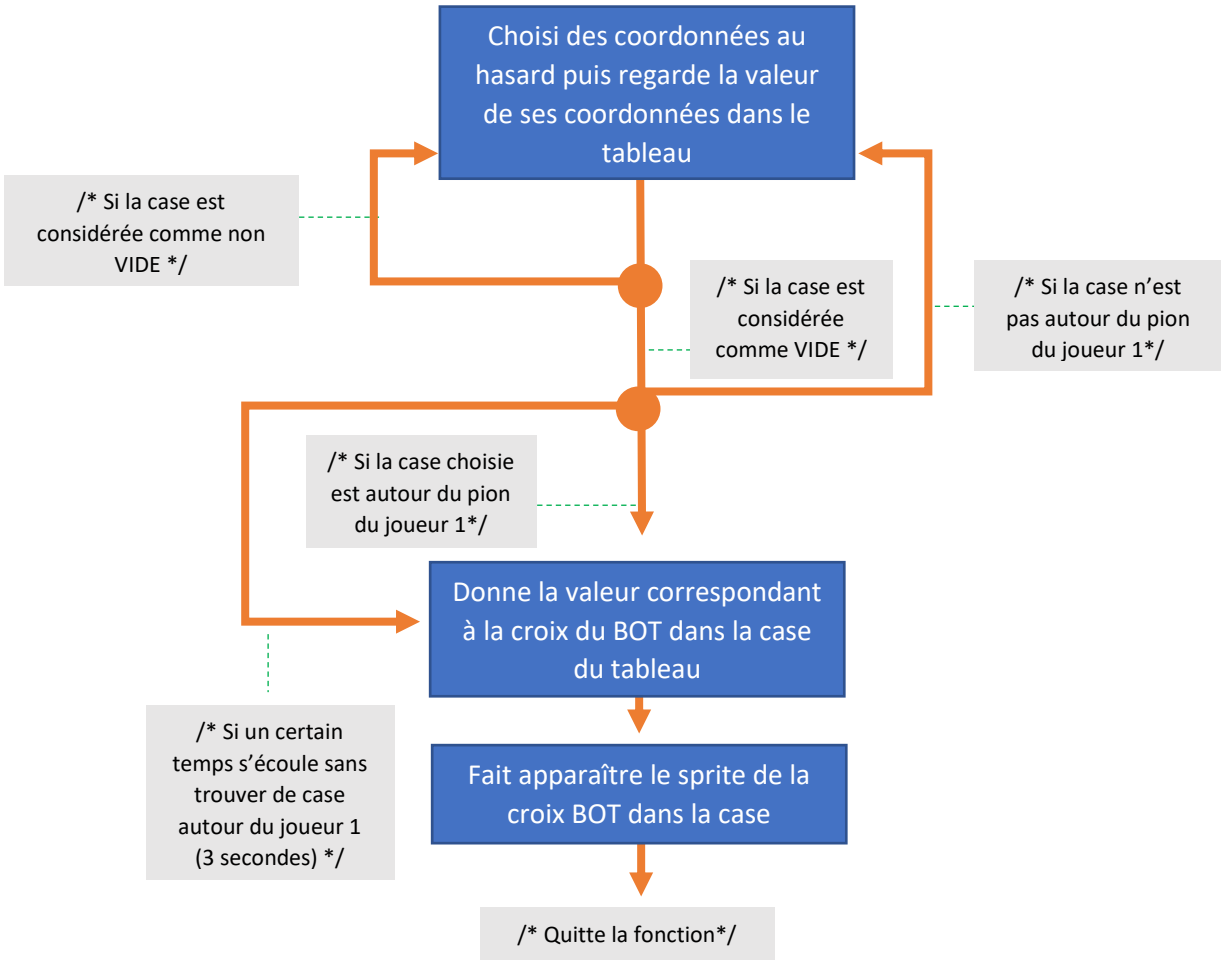
/* Fonction « **PlacementCroixJoueur** » qui s'occupe du placement des croix représentant les cases condamnées*/



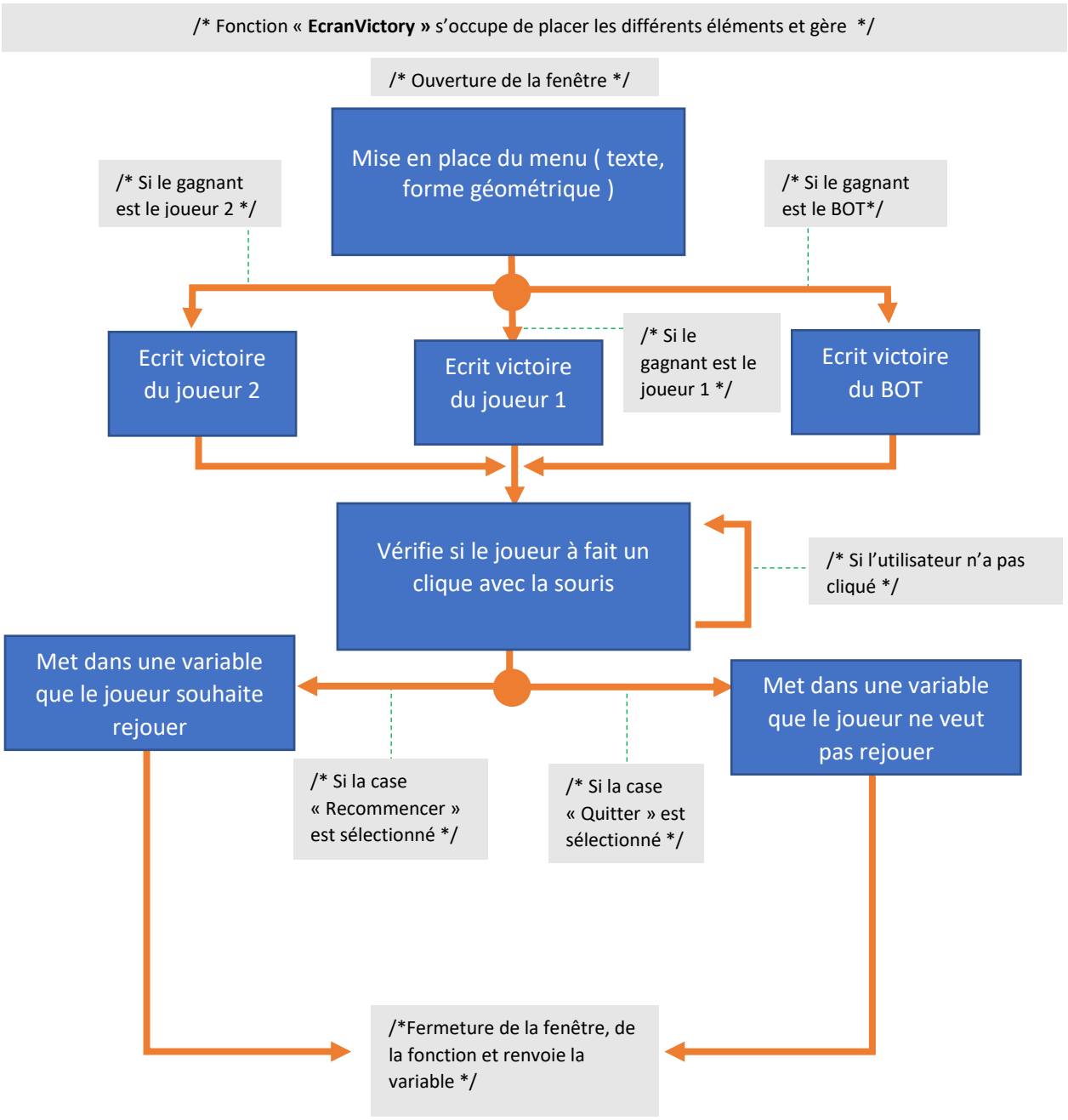
/* Fonction « **Tour2BOT »** semblable à la fonction TourBot en permettant en plus de pouvoir condamner une case mais permet uniquement de se déplacer autour du pion */



/* Fonction « **PlacementCroixBot** » permet de poser la croix du bot de façon logique ou aléatoire selon la situation */



Ainsi, si un joueur est considéré comme perdant par la fonction « VerifyLoser », on quitte la boucle pour ensuite fermer la fenêtre. On lance la fonction « EcranVictory » qui ouvre une fenêtre correspondant à l'écran de victoire.



Si le joueur souhaite rejouer, le jeu revient à la toute première fenêtre et recommence. Si le joueur ne souhaite pas rejouer, le jeu se ferme et le code se termine.

Conclusion

Dylan Girault :

L'ajout de fonctionnalité tel que colorer l'emplacement de la souris était compliqué mais très pratique pour voir la détection du jeu par rapport à la position de la souris. D'ailleurs, nous avons d'ailleurs voulu rendre le BOT plus difficile et je suis plutôt satisfait du système d'obligation à condamner une case autour du joueur ce qui complexifie le BOT. J'ai beaucoup apprécié produire le jeu en groupe car malgré la difficulté à produire un code lisible et simple à comprendre pour les autres, j'ai pu expérimenter de nouvelles choses. Avoir des outils faciles en notre procession pour produire quelque chose de nouveau et complexe fut très drôle mais surtout très gratifiant. Je pense sincèrement que la plus grande difficulté n'est pas le codage mais le travail d'équipe que demande le projet, s'avoir être organisé avec son partenaire et bien plus complexe qui n'y paraît et cette expérience fut très enrichissante dans ce sujet.

Céline Azevedo :

Ce projet m'a permis de mettre mes compétences en pratique en les utilisant dans un contexte beaucoup plus large qu'un simple exercice, car il s'agissait de créer un jeu ex nihilo. J'ai pu constater des mauvaises habitudes de programmation que je pratiquais régulièrement et améliorer mes compétences ainsi que mes connaissances du C. L'utilisation de la bibliothèque graphique était difficile pour moi au début, car je ne l'avais pas souvent utilisé, mais grâce à l'aide de mon binôme et à beaucoup de pratique, le tout m'est venue aisément.
De plus, ce projet n'a pas seulement amélioré mes compétences informatiques, il m'a permis de mieux travailler et communiquer en équipe, et plus précisément dans ce cas, en binôme. Nous nous sommes mis d'accord très rapidement sur la façon de procéder avec une répartition des tâches équilibrée. Néanmoins, nous nous sommes souvent aidés mutuellement sur nos lacunes respectives pour finalement arriver à un résultat très satisfaisant.
En conclusion, ce projet m'a permis d'améliorer deux grandes compétences, l'informatique et le relationnelle, que je juge fondamentale pour ma future insertion dans le marché du travail.