Rapport de Projet : MasterMind

I) Modèle :

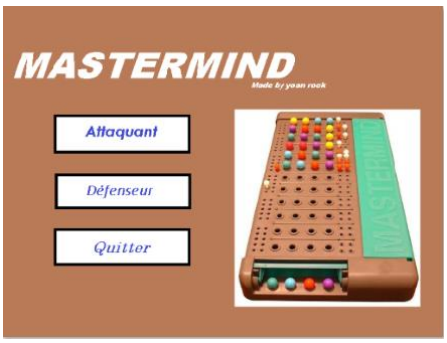
Les combinaisons de l'attaquant : stocker dans un tableau de dimension 2 (10 lignes) (4 colonnes).

Explication : Grace au tableau je peux accéder a tout moment à toute les combinaisons dans mon tableau multidimensionnel, si j'aurai utilisé un simple tableau, il aurai écrasé les précédentes combinaison **dans la mémoire.**
La combinaison du défenseur : stocker dans un tableau de capacité 4 contenant des char.

Explication : façon la plus simple de stocker et qui prend le moins de place, de plus pour ma part je trouve cela plus dur de stocker des int et de devoir connaitre la correspondance entre la couleur et son numéro alors que le lecteur du code comprend directement lorsque qu'il voit un 'r' !

Les vérifications : sont stockées dans des variables que j'utilise dans mon Main par adresses ou par valeur.

II) Vue :

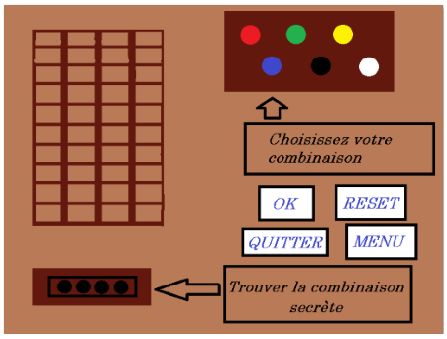


3 boutons présents :

- Attaquant ; (Envoi en Attaque)

- Défenseur ;(Envoi en défense)

- Quitter ;(Quitte le jeu)



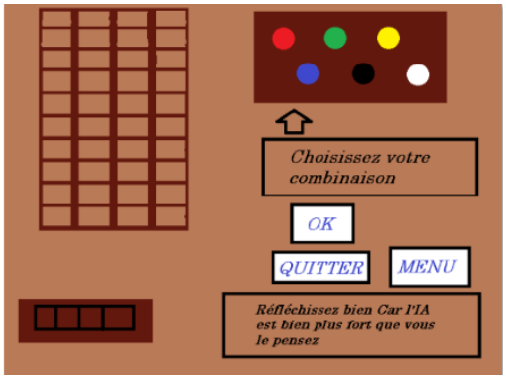
4 boutons présents :

- Ok : lorsque la combinaison rentré est celle souhaité, pour passer à la prochaine tentative.

- reset : supprime la combinaison rentrée pour recommencer.

-Menu : Retour au Menu.

- Quitter : quitte le jeu.



3 boutons présents :

- Ok : lorsque la combinaison secrète est rentrée, et pour passer à la prochaine tentative du IA.

-Menu : Retour au Menu.

- Quitter : quitte le jeu.

Victoire = Attaquant Gagne.

Défaite = Attaquant perd.

III) Contrôleur :

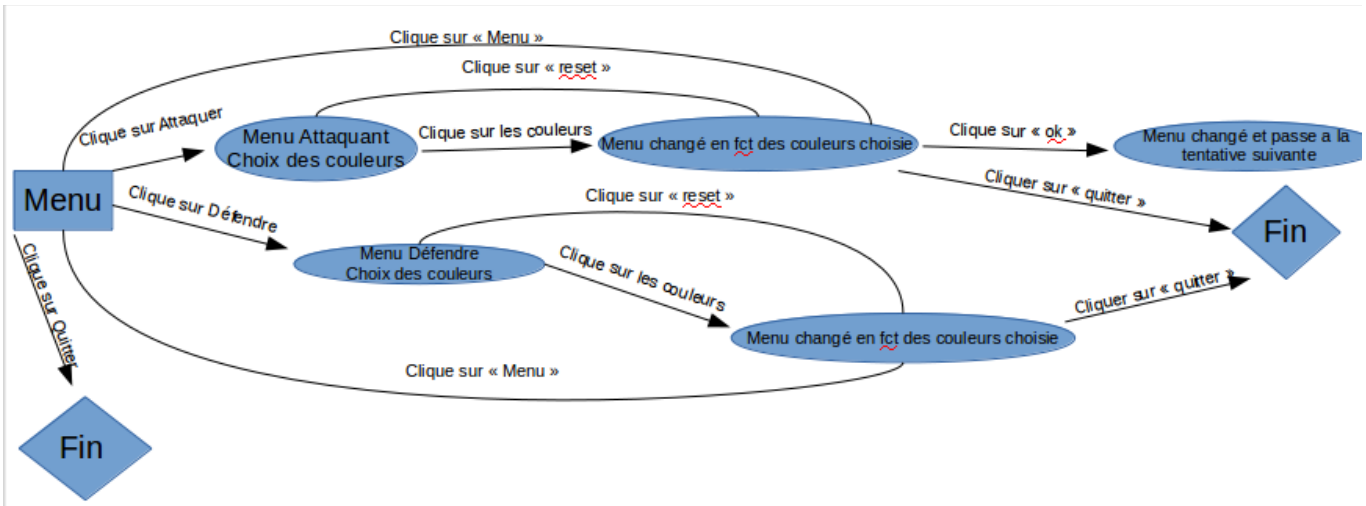


Diagramme des états et en transitions :

IV) Développement

Fenetre() => Initialise le graphique et créer une fenêtre (placer en dehors de toute boucle pour éviter message d'erreur « une seule on a dit ! »)
Tout d'abord je rentre dans ma boucle infinie menu.

Elle me permet de contrôler mon menu, si je veux quitter, rester dedans, aller dans mes futur menu..

F_display_menu() ; => affiche L'image de fond de mon Menu

F_choixecran() => renvoi un char ('a','d','q') indiquant le bouttton appuyer par l'utilisateur

A) Attaque :

F_alea_att(tabdef) ; => prend en argument le tableau du défenseur IA et placer une combinaison aléatoire de 4 couleurs.

On rentre ensuite dans le for pour Afficher les couleur en fonction de la position du clique de l'utilisateur, il a le choix entre 6 couleur et 4 bouton (10 if)

Je rempli donc le tabcolor (tableau multidimensionnel par la char correspondant à la couleur)

Et j'affiche la couleur.

Si on clique sur ok **: f_verif()** => affiche les pions noir et blanc pour continuer à deviner la combinaison secrète.

C'est ici que j'ai eu le plus de mal car je me suis compliqué a faire plein de if au lieu de réfléchir simplement de créer des tableau qui « lockerai les vérifications effectuer » et de faire seulement 2 boucle for !

B) Défense :

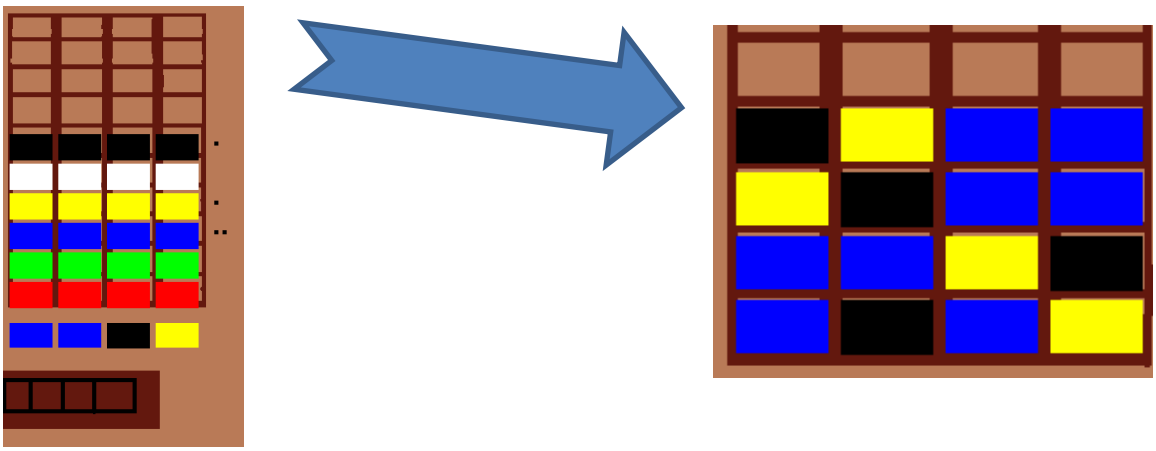
Déjà j'initialise mes tableau de '0'.

F_combi_def() = me permet de faire choisir la combinaison secrète par le joueur.

J'ai eu un deuxième problème ici : mon programme tournais a chaque une tour de boucle pour rien donc indiceligne s'incrémentai de 1 et a cette ligne rien ne s'affichait ; mais c'était simplement que dans **f_conv_int_char()** ; je n'avais pas penser au faite qu'e le nombre que je lui envoyer devait être compris entre 1 et 6 (nombre aléatoire grâce au modulo 6+1) mais seulement a l'initialisation l'indiceligne étais égale a 0 et je lui demander de me traduire indiceligne a 0 au er tour de boucle donc 0 , je faisais donc un énorme déplacement de tableau et j'ai résolu cela en changeant mon nombre aléatoire sortie en attaque : entre 0et 5 maintenant et je return a et non a-1 du tableau (donc à l'indice [-1] qui n'existe pas bien sur ^^)

C) Idée de l'IA : fonction f_ia() :

Il remplit les combinaisons de la même couleur tant qu'il n'a pas cumulé 4 pions noirs, cela lui permet de connaitre les proportions et les couleurs présentent dans la combinaison



J'utilise donc deux variable m'indiquant l'indice auquel je dois écrire dans le tableau et jusqu'où vpn et ancienvpn !

Ex : (Cela nous coute 6 tentatives) mais Grâce a ça, ici on sait directement qu'il y aura forcement 2 bleu, 1 jaune et 1 noir.

'J'aurai même aimé coder (pas réussi) si je trouve 3 pion noir et qu'il reste 1 couleur a tester, ne surtout pas faire la tentative et en déduire de suite dans le tabfind.

Je stocke donc ces 4 donnée (char) dans mon tableau tabfind dans les bonnes proportions et il me reste plus qu'à essayer toute les combinaisons ex ici : NJUU/UUNJ/UNUJ/…

L'IA aura une probabilité plus élevée a trouver la bonne combinaison que si elle était aléatoire.