Mini-Projet : Morpion double-reversi

Description du fonctionnement global du programme

Le programme sera structuré en trois parties :

1 - Les objets avec : une classe ControleNode qui va contenir des variables constantes, comme le nombre de jeton restants etc... et modifiables à tout instant pour coordonner le jeu. Un objet Pion, attribué à chaque rectangle du tableau. L'objet Pion définira si le rectangle est vide ou s'il contient un pion rouge/bleu/clair/sombre. En fonction de cette valeur on affichera une image différente de jetons sur la case ou pas du tout. Il contiendra toutes les méthodes nécessaires pour faire interagir le joueur avec le jeu ou de mouvoir les autres pions sur le tableau.

2 - Les fonctions seront là pour appuyer les méthodes utilisées par l'objet pour par exemple vérifier si une des combinaison de pions gagnantes se trouve sur le plateau, réinitialiser le jeu etc...

3 - Le graphisme avec Tkinter et ces variables. Les variables sont surtout le chemin d'accès jusqu'au icones de jetons, les boutons tkinter et leur configuration, ainsi que des tuples qui permettent d'indexer la position des jetons. De cette manière il est facile pour l'objet Pion de se coordonner avec les cases alentour (voir méthodes PionObjectCoreInter())

Fonctionnalités implémentées

Le morpion fonctionnel avec un système tour par tour, les placements de pions sur un tableau 4x4 et les règles du double reversi pour changer la position de son adversaire. Description des méthodes et fonctions.

Programmation par : – Muhammad Khalid

Descriptions des méthodes et fonctions :

Fonctions :

– board() : dessine les rectangles du tableau de jeu

– pturn() : va alterner l'appartenance du tour pour chaque joueur

– Init_cases() : réinitialise toutes les variables de ControlNode et enlève les jetons à l'écran pour laisser un tableau de jeu vide.

– allowinteraction() : active toutes interaction (cliques) possible sur les cases

– blockinteraction() : désactive toutes interaction (cliques) possible sur les cases

– winningifloop() : vérifie la configuration de chaque jeton pour trouver une combinaison gagnante. Elle return True quelqu'un à gagner.

– winnerbool() : utilise la fonction winningifloop pour changer le message de score en WIN!!! Si combinaison victorieuse

– useful_updat_token() : utilise ControlNode pour trouver l'index d'une case et la supprimer au besoin

– relaunchgame() : comme son nom l'indique, elle relance le jeu

– giveup() : Pour capituler

Méthodes :

– ControlNode : c'est une classe qui contient 5 variables à utiliser de manière globale pour coordonner la partie

Objet Pions :

– movetoken() : en fonction de la direction indiqué elle initiera un déplacement du jeton vers celle-ci

– playtoken() : par souci d'event tkinter, cette méthode est dupliqué quatre fois, elle permettent toutes de poser un jetons sur la case cliquer et régler ControlNode en conséquence

– check_empty() : return vrai si il n'est pas vide

– remove_token() : supprime le jeton et réinitialise sa couleur

– board_owner() : Cette méthode coordonne le choix de jeton du joueur (clair ou sombre) avant d'alterner l'appartenance du tour.

– ultimove() : Utilisée par playtoken il permet de dessiner le jeton au bon endroit et supprimer l'ancien

– movecancel() : Permet d'annuler son event cliqué pour choisir une nouvelle case avant de d'avancer

– PionObjetCoreInter() : Cette méthode au nom barbare permet de changer la couleur des prochains pions placés en fonction du joueur possesseur du tour et créer les événements autour des cases du pion sélectionn