NSI	GUI	

Les dominos

I- Présentation

Le site : https://www.regles-de-jeux.com/regle-domino/ Nous apprends la règle du jeu.

Ce jeu très ancien est toujours populaire pour sa simplicité de jeu. Ce jeu se joue normalement de 2 à 4 joueurs. Pour ce projet, on jouera à deux, ou seul contre l'ordinateur.

Le but du jeu du domino est d'être le premier joueur à avoir posé tous ses dominos.

Chaque joueur reçoit 7 dominos. Attention, les dominos doivent être distribués points cachés. Le reste des dominos fait office de pioche.

Le joueur ayant le double le plus élevé (le double 6 donc) commence la partie de domino. Si personne ne possède ce domino, ce sera le joueur ayant le double le plus fort. Le joueur suivant doit à son tour poser un domino ayant le même nombre de points sur au moins un côté du domino précédemment posé. Si le joueur possède un domino correspondant, il le pose à la suite du domino. Sinon, il pioche un domino et passe son tour. Au fur et à mesure de la partie, les dominos forment une chaîne.

Pour gagner au domino, il suffit d'être le premier joueur à avoir posé tous ses dominos. Il se peut que le jeu soit bloqué. Alors le joueur ayant le moins de points est déclaré vainqueur.

II- L'interface graphique

On vous fourni une interface graphique à travers la classe GUIblackjack que vous pouvez importer depuis le module fourni. Si le répertoire GUI_domino est dans le même répertoire que votre fichier : :

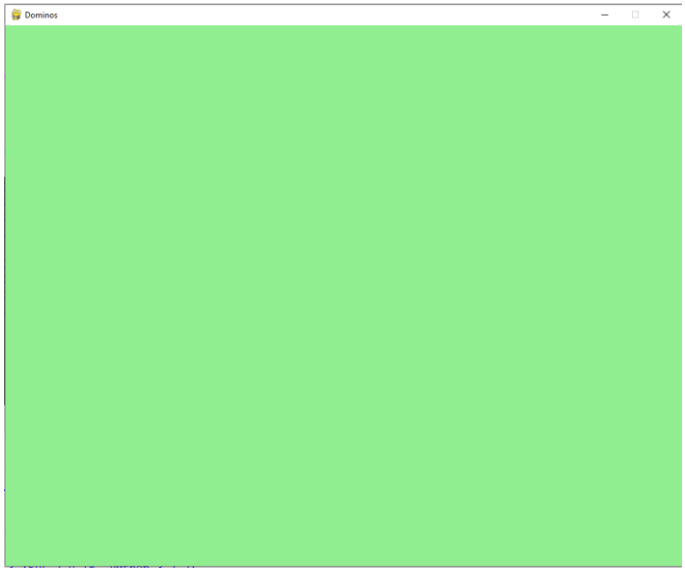
from **GUI_domino.guiDomino** import **GUIdomino**

Constructeur de la classe :

- Attribut public : **aucun**.
- Le constructeur ne prend aucun paramètre. Il permet de créer un objet graphique vierge.

Par exemple :

GUI = GUIdomino()



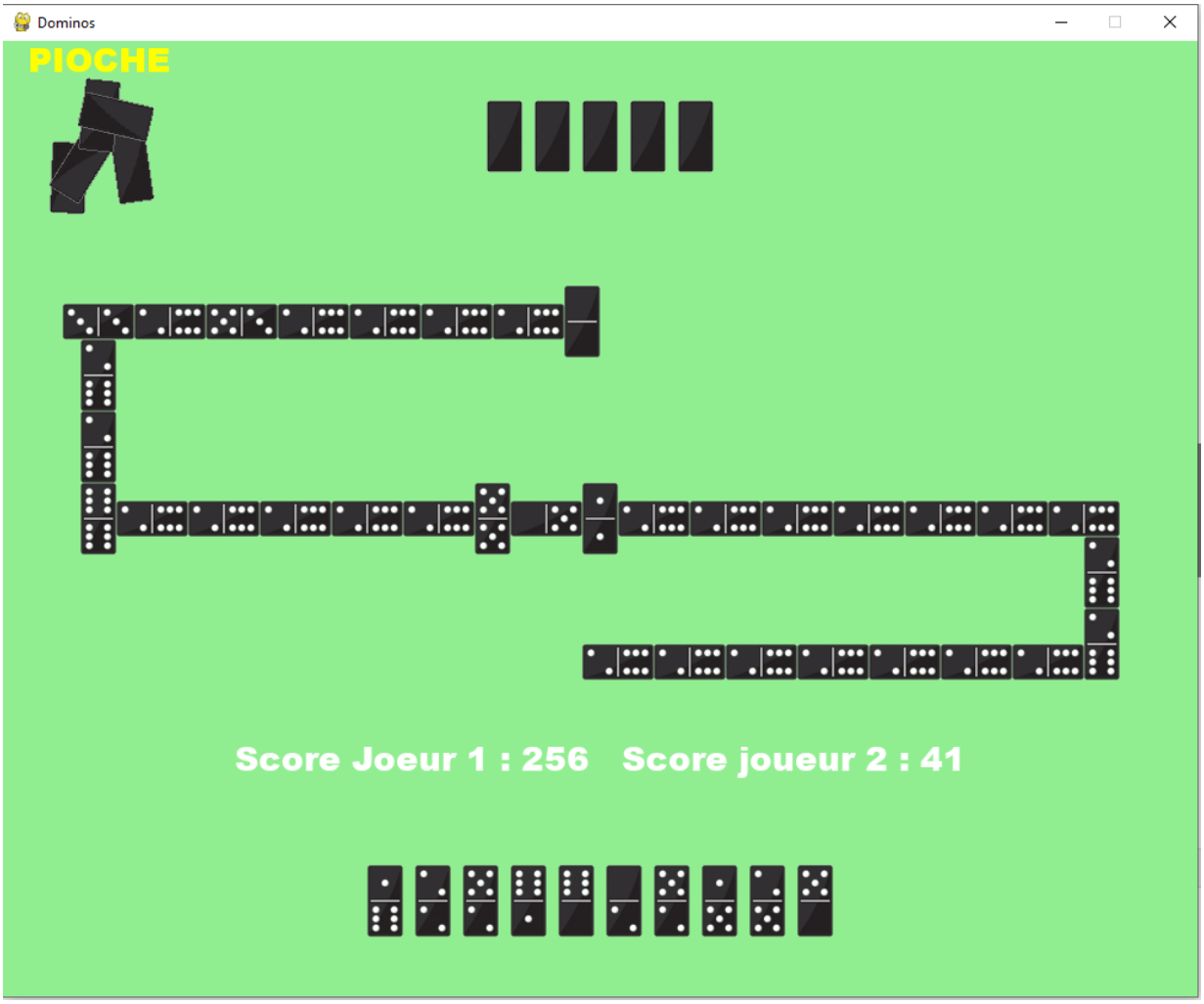
- Les méthodes :
  - **refresh(g, j1, j2, n, t = "", cacheJ2 = False)** : Cette méthode rafraichie l'affichage du jeu de domino conformément à la grille passée en argument.
    - **j1** et **j2** sont des objets indexables, dont les éléments sont de la forme (n, m). Où n et m sont deux entiers de 0 à 6 représentant le nombre de points de chaque coté du domino. D'autre part j1 et j2 doivent disposer d'une méthode __len__ pour répondre a la fonction len(). Il peut donc s'agir de listes, de tuples ou de tout autre objets répondants à ces critères.
    - **cacheJ2** est un booléen. Si cet argument est vrai, toutes les dominos de l'ordinateur seront affichés retournés.
    - **n** est le nombe de dominos dans la pioche.
    - **t** est un texte à afficher à destination du joueur.

Par exemple :

>>> j1 = [(randint(0,6), randint(0,6)) for i in range(10)]

>>> j2 = [(randint(0,6), randint(0,6)) for i in range(5)]

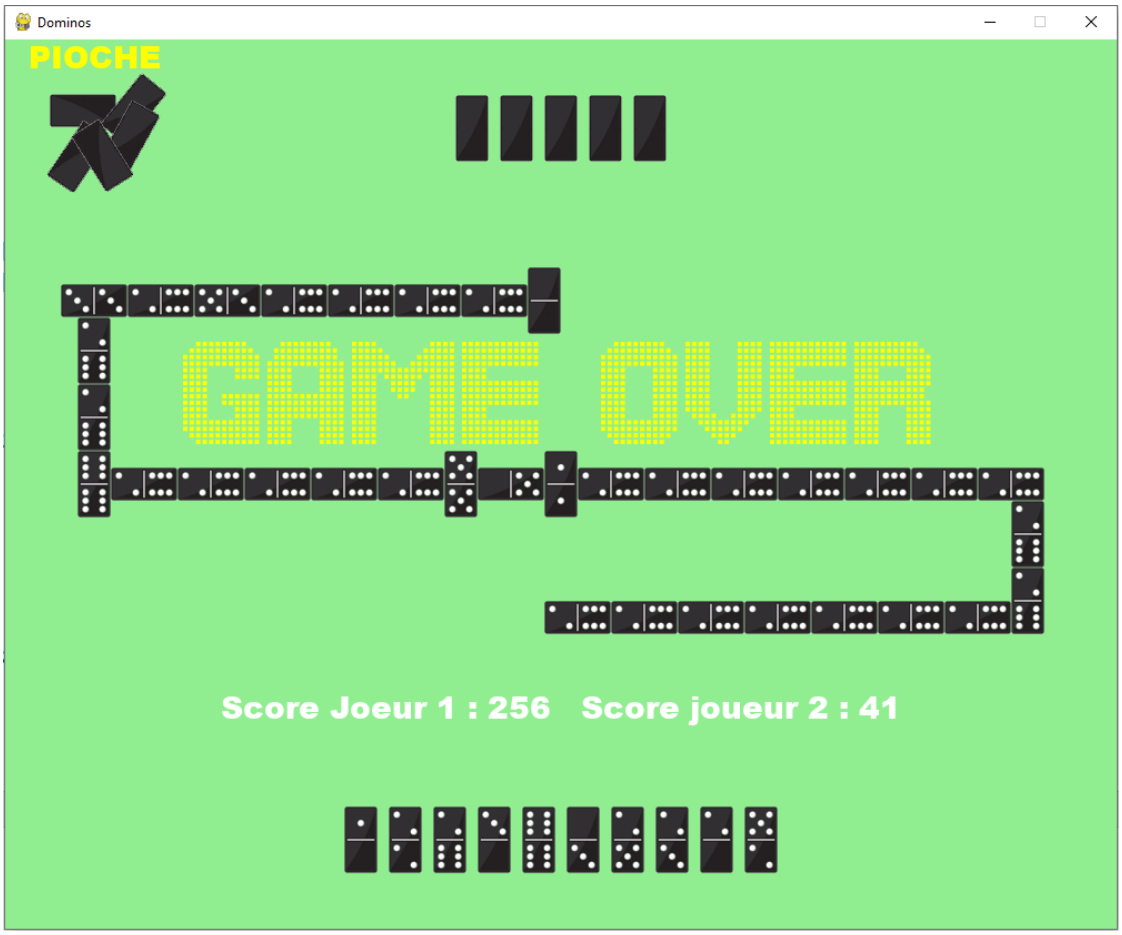
>>> GUI.refresh(g, j1, j2, 5, "Score Joeur 1 : 256 Score joueur 2 : 41", True)



- **messCentre(mess)** : Affiche le message mess en gros, en plein milieux de l'écran. Cette méthode est principalement destinée a afficher la victoire ou la défaite du joueur.

Par exemple :

GUI.messCentre("GAME OVER")



- **waitClick()** : Cette méthode attend l'action d'un joueur. Elle gère trois types d'actions :
  - **demande fermeture de la fenêtre** : fermeture propre de la fenêtre pygame et fin du programme python.
  - **click sur la fenêtre** :
    - click sur la pioche : renvoi '#P'
    - click sur l'extrémité droite du jeu : renvoi '#D'
    - click sur l'extrémité gauche du jeu : renvoi '#G'
    - click sur la pioche : renvoi '#P'
    - click sur les pièces du joueur 2 (en haut) : renvoi '#A'
    - click sur une des pièces du joueur 1 (en bas) : renvoi le numéro de la pièce (de type int).
  - **appui sur des touches** spéciales :
    - fleche RIGHT : retourne '_R';
    - fleche LEFT : retourne '_L';
    - fleche DOWN : retourne '_D';

    - fleche UP : retourne '_U';
    - touche BACKSPACE : retourne '_B';
    - touche RETURN : retourne '_E';
    - touche ESCAPE : retourne '_S';
- **appui sur une autre touche du clavier** : retourne le caractère unicode correspondant.

Attention, une fois exécutée, on ne peut sortir de cette méthode que par l'une de ces trois actions.