

SAÉ

1.01

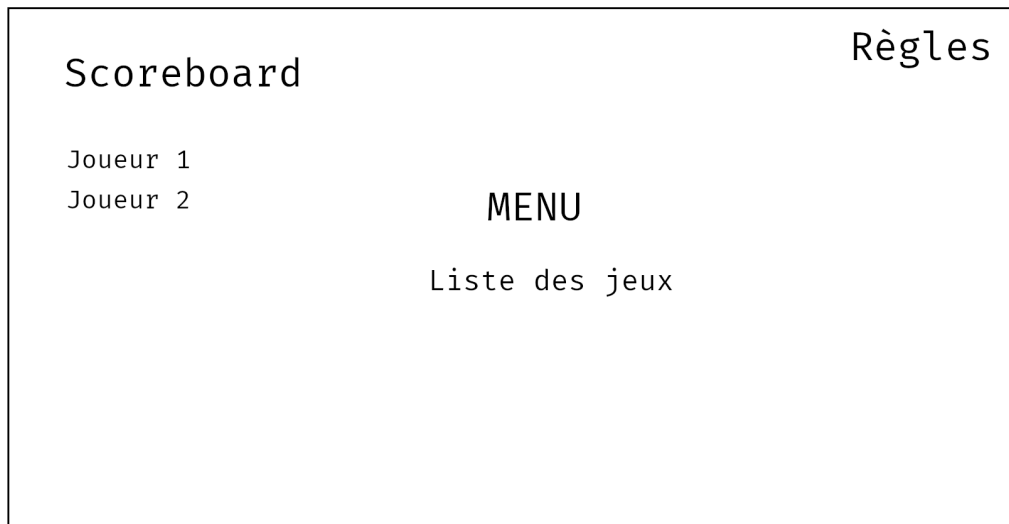
Table des matières

Préparation :	2
Menus :	2
Menu Principal :	2
Menu Règles & Menu Scoreboard :	3
Menu Devinette :	3
Menu Allumettes :	5
Menu Morpion :	6
Écran de victoire :	7
Code :	7
Bibliothèques externes à mon projet :	7
Termios et Tty:	7
Os :	7
Sys :	8
Fichier :	8
main.py :	8
rules.py :	8
scoreboard.py :	8
termUtils.py :	9
players.py :	9
ANSIColors.py :	9
devinette.py :	9
allumettes.py :	9
morpion.py :	10
puissance4.py :	10
Vidéos de présentations :	11
Potentiel problèmes de mon programme :	11

Préparation :

Menus :

Menu Principal :



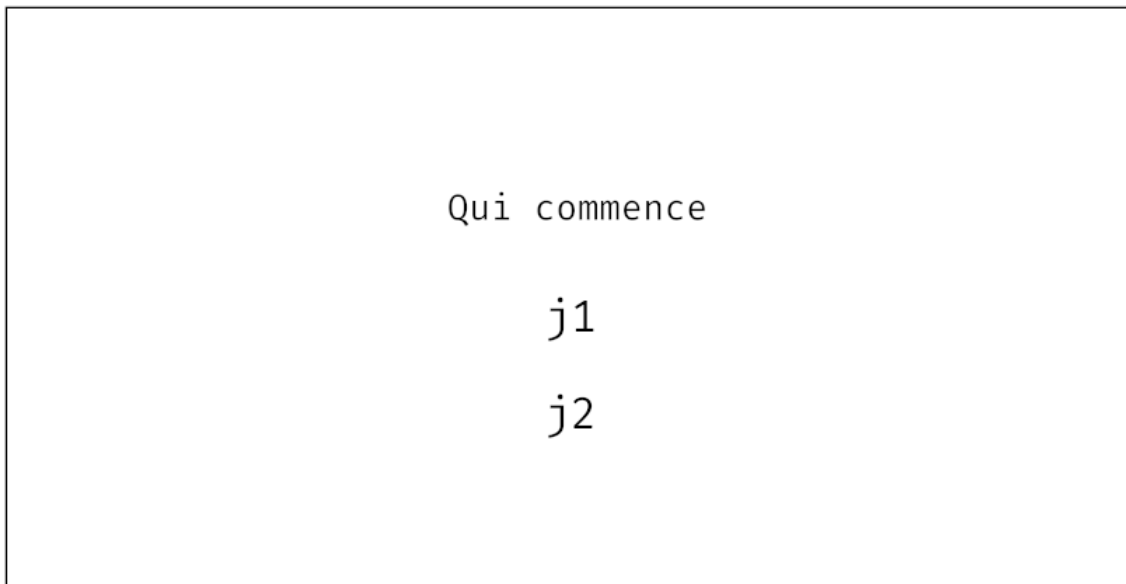
La première chose qui m'est venu à l'esprit pour le menu était de faire un carré prenant tout le terminal et afficher les différentes options, tout en utilisant les touches du clavier pour naviguer dans les menus.

Menu Règles & Menu Scoreboard :



Donc après j'ai imaginé le menu des règles et le menu des scoreboard en gardant le même style que le menu principal. Menus assez simple.

Menu Devinette :



Quel est le chiffre
(curseur)

Quel est votre supposition
(curseur)

Quel est le stade de la supposition

Plus grand — 1 strike
Plus petit — 1 strike
C'est bon — Pendu

Donc dans l'ordre je demande qui commence pour éviter le redémarrage du jeu si je veux changer les joueurs veulent changer de sens

Après je demande à la personne qui fait deviner le chiffre à faire deviner

Ensuite la personne qui devine rentre un chiffre

Et ensuite la personne qui fait deviner le chiffre a le choix de prendre "Plus grand", "Plus Petit" et "C'est bon", mais elle peut mentir sur le vrai stade du chiffre donné, si il dit que le chiffre est plu grand alors qu'il est plus petit (ou inversement) il prendra un strike au bout de trois celui qui devine gagne sans avoir forcément trouvé le chiffre.

Si il ne dit pas que le chiffre est juste alors qui l'est, celui qui fait deviner perd directement.

Menu Allumettes :

(Même début que devinette pour choisir qui commence)

```
Il reste -- allumettes
Combien a retirer
      1
      2
      3
```

Ici j'affiche le nombre restant d'allumettes et demande combien il faut en retirer. Si il y a disons 2 allumettes restantes le joueur ne pourra plus sélectionner 3.

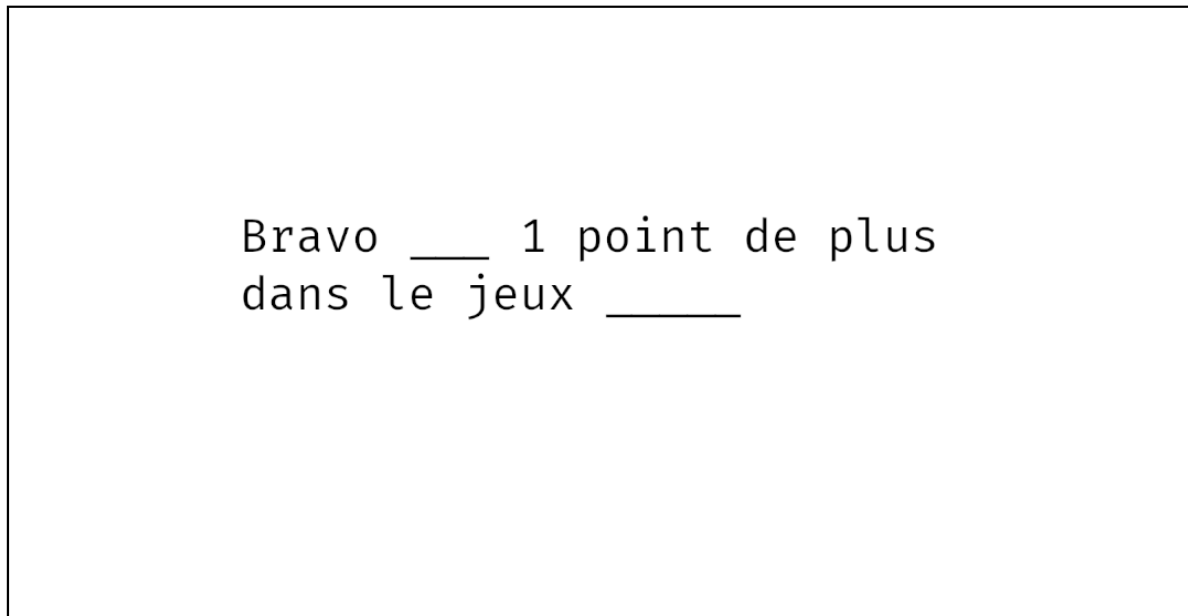
Menu Morpion :

(Même début que devinette pour choisir qui commence)

```
A ton tour ____
  |
--|
  |
  |
--|
  |
  |
```

J'affiche la grille morpion puis laisse le joueur se déplacer avec les flèches de clavier.

Écran de victoire :



J'affiche cet écran après la victoire d'un des joueurs pendant 1s.

Code :

Bibliothèques externes à mon projet :

Termios et Tty:

Termios est un utilitaire de terminal, qui a pour but de récupérer les "options" du terminal actuel et de les modifier. Tty de l'autre est un add-on de Termios qui rajoute des utilitaires pour être plus précis sur le changement d'option.

Os :

Je n'importe que 3 commandes de os:

- system : Me permet d'exécuter des commandes comme si j'étais dans un terminal

- `get_terminal_size` : Me permet de récupérer le nombre de lignes et de colonnes disponible sur le terminal actuel.
- `path` : Je l'utilise seulement pour vérifier si un fichier existe

Sys :

Je n'utilise qu'une seule commande de sys et c'est la commande `stdin.read(1)` qui me permet de lire sur l'entrée standard 1 caractère.

Fichier :

main.py :

C'est ici que le programme commence, donc il est gérer l'affichage du menu principale et de l'affichage de l'enregistrement des joueurs.

Dans un premier temps je demande le pseudo des 2 joueurs puis ensuite je vérifie si les 2 ne sont pas les mêmes et s' il existe dans le fichier. Puis si les joueurs n'existent pas alors je les créerai.

Après ça je peux directement afficher le menu et commencer à enregistrer les touches pressées par l'utilisateur. Ensuite j'agis en fonction de l'option choisie.

rules.py :

C'est ici que l'affichage du menu des règles et des règles est géré. J'affiche le menu puis ensuite enregistré les touches pressées puis agis en fonction.

scoreboard.py :

C'est ici que l'affichage du menu des scoreboards et des scoreboards est géré.

J'affiche le menu puis ensuite enregistré les touches pressées puis agis en fonction.

termUtils.py :

C'est ici que je fais disparaître le curseur et que je change la façon dont le terminal fonctionne. Je fais aussi le check de si le fichier gérant les joueurs est créé ou pas. J'ai aussi les fonctions permettant d'écrire partout dans le terminal.

players.py :

C'est ici que les points et que le fichiers lié au joueurs est géré.
J'ai ici les fonctions permettant d'ajouter des points et de vérifier si un joueur existe etc

ANSIColors.py :

C'est ici que les couleurs et les background sont gérés.

devinette.py :

C'est ici que l'affichage et le fonctionnement du jeu de devinette.
En premier lieu j'affiche le menu puis enregistre les touches pressées par l'utilisateur, ensuite j'agis en fonction.

Au début du jeu je demande qui fait deviner un chiffre à l'autre puis ensuite je lui demande de le noter, pour éviter d'afficher le chiffre caché je n'affiche pas les touches pressées et pour éviter de faire un trop gros chiffre j'empêche l'écriture de nombre a plus de 3 chiffres. Ensuite je demande à l'autre joueur de deviner. Et pour finir la boucle de jeu, je repasse au joueur 1 et lui demande l'état de la réponse du joueur 2.

A la fin, j'affiche quel joueur a gagné et je lui donne un point dans le jeu en question.

allumettes.py :

C'est ici que l'affichage et le fonctionnement du jeu d'allumettes.
En premier lieu j'affiche le menu puis enregistre les touches pressées par l'utilisateur, ensuite j'agis en fonction.

En premier lieu je demande qui commence ensuite j'affiche le nombre d'allumettes restantes, a qui est le tour et un petit menu permettant de sélectionner un chiffre entre 1 et 3.

Si il y a par exemple 1 allumettes restante alors les boutons 2 et 3 ne sont plus validables.

Quand le nombre d'allumettes est égal a 0 alors le joueur qui vient de jouer perd et l'autre joueur gagne. J'affiche le joueur gagnant et ensuite lui donne le point correspondant.

morpion.py :

C'est ici que l'affichage et le fonctionnement du jeu de morpion.

En premier lieu j'affiche le menu puis enregistre les touches pressées par l'utilisateur, ensuite j'agis en fonction.

En premier lieu je demande qui commence ensuite j'affiche le morpion en lui même, le joueur peut se déplacer de haut en bas et de gauche à droite avec les flèches.

Si un pion est posé sur la grille alors je vérifie si il y a un pion à l'endroit présent, puis ensuite le rajoute pour ensuite vérifier si ce n'est pas un mouvement gagnant. Si ce n'est pas le cas alors je continue. Si la grille est pleine alors c'est une égalité et donc j'arrête le jeu et affiche un écran d'égalité. Quand un joueur gagne, j'affiche le joueur gagnant et ensuite lui donne le point correspondant.

puissance4.py :

C'est ici que l'affichage et le fonctionnement du jeu de puissance 4.

En premier lieu j'affiche le menu puis enregistre les touches pressées par l'utilisateur, ensuite j'agis en fonction.

En premier lieu je demande qui commence ensuite j'affiche le puissance 4 en lui même, le joueur peut se déplacer seulement de gauche à droite avec les flèches.

Si un pion est posé sur la grille alors je vérifie si il y a un pion à l'endroit ou le pion devrait atterrir, puis ensuite le rajoute pour ensuite vérifier si ce n'est pas un mouvement gagnant. Si ce n'est pas le cas alors je continue. Si la grille est pleine alors c'est une égalité et donc j'arrête le jeu et affiche un écran d'égalité. Quand un joueur gagne, j'affiche le joueur gagnant et ensuite lui donne le point correspondant.

Pour des informations plus précises il faut aller voir le code entièrement commenté.

Vidéos de présentations :

Menu maître : [Lien](#)

Menu des règles : [Lien](#)

Menu des scoreboard : [Lien](#)

Jeu de devinettes : [Lien](#)

Jeu des allumettes : [Lien](#)

Jeu du morpion : [Lien](#)

Jeu du puissance 4 : [Lien](#)

Potentiel problèmes de mon programme :

Le centrage de certaines parties reste à désirer mais je n'ai pas réussi à trouver de solution dans le temps imparti.

La gestion des points est assez basique.

Le code est répété pas mal de fois malgré le fait que j'ai essayé d'éviter ça un maximum du temps.

L'affichage, malgré le fait qu'il soit adaptable, fait des problèmes avec certaines tailles de terminaux (haut mais fin par exemple).

Mon programme **N'EST PAS** compatible avec Windows, il ne fonctionne que sur des machines Unix parce que "termios" est une librairie standard d'Unix, aussi j'utilise la commande "clear" n'existe pas sous Windows, par ailleurs j'aurais pu palier à ce problème en utilisant "clear || cls" ("cls" étant l'équivalent Windows de "clear").