# Compte Rendu - Programme Puissance 4

## Sujet

Implémentation d'un jeu Puissance 4 en Python avec une interface graphique utilisant Tkinter.

## Description générale

Le Puissance 4 est un jeu de stratégie pour deux joueurs dans lequel chaque joueur place à tour de rôle des pions dans une grille de 7 colonnes et 6 lignes. L'objectif est d'aligner quatre pions de la même couleur, que ce soit horizontalement, verticalement ou en diagonale. Le programme gère la gravité pour placer les pions dans la première case vide d'une colonne, vérifie en temps réel les conditions de victoire et affiche un message indiquant le gagnant dès qu'un alignement est détecté.

## Cahier des charges

Le programme doit :  
• Afficher une interface graphique avec une grille de 7 colonnes et 6 lignes.  
• Permettre à deux joueurs de s'affronter en déterminant aléatoirement qui commence la partie.  
• Gérer la gravité pour que les pions se placent dans la première case libre de la colonne choisie.  
• Vérifier les conditions de victoire (alignement horizontal, vertical et diagonal) à chaque coup.  
• Empêcher le placement d'un pion dans une colonne déjà pleine.  
• Afficher dynamiquement le joueur actif et, en cas de victoire, un message annonçant le gagnant.

## Choix des outils

• Langage : Python 3  
• Bibliothèque graphique : Tkinter (Canvas, Label, Button)

## Structure du programme

Le programme est organisé autour de plusieurs fonctions clés :  
1. **pile\_ou\_face()** : Détermine aléatoirement le joueur qui commence.  
2. **init\_ui()** : Initialise l'interface graphique, crée la fenêtre principale, le canvas et le label pour afficher le gagnant.  
3. **init\_game\_grid()** : Dessine la grille de jeu et initialise le label du joueur actif.  
4. **update\_game\_grid()** : Met à jour l'affichage des pions sur la grille après chaque coup.  
5. **column\_select()** : Crée les boutons permettant de sélectionner la colonne où déposer un pion et met à jour le joueur actif.  
6. **gravity(column)** : Calcule la première ligne vide dans la colonne sélectionnée, simulant l'effet de la gravité.  
7. **select\_column(col)** : Gère le placement du pion, la mise à jour de la grille, la vérification du gagnant et le changement de joueur.  
8. **check\_winner(board)** : Vérifie la présence d'un alignement de 4 pions (horizontal, vertical ou diagonal) pour déterminer le gagnant.

## Répartition des tâches

• **Affichage et interface graphique** : Nathan  
• **Choix du premier joueur** : Samuel  
• **Gestion des tours de jeu** : Samuel & Bastian  
• **Détection des alignements** : Samuel  
• **Blocage des colonnes pleines** : Bastian  
**• Gestion de la gravité (placement des pions)** : Bastian  
• **Sélection des colonnes pour le placement d'un pion** : Nathan & Bastian

## Difficultés rencontrées

• Mise en œuvre de la gravité pour placer correctement les pions.  
• Développement d'un algorithme efficace pour vérifier les conditions de victoire sur la grille.  
• Coordination entre la logique de jeu et l'interface graphique pour assurer une mise à jour en temps réel.  
• Gérer de manière optimisé la logique des colonnes.

## Ouvertures possibles

• Améliorer l'interface graphique avec des animations pour le placement des pions.  
• Intégrer un système de score et un historique des parties.  
• Permettre une rejouabilité sans avoir à redémarrer l'application et envisager une version en ligne.