

**―――――――――――――――――――――――――――――――――――――――――**

PROGRAMMATION PYTHON

**1A**  **2023-2024**

**―――――――――――――――――――――――――――――――――――――――――**

Désignation du projet : Gomoku (puissance 5)

**Groupe** FB

**Equipe Projet**

Pilote : Bryan THIRIMANNA

Participants :

Louis LUCAS – Raphaël VEY

Table des matières

[Introduction 3](#_Toc166280957)

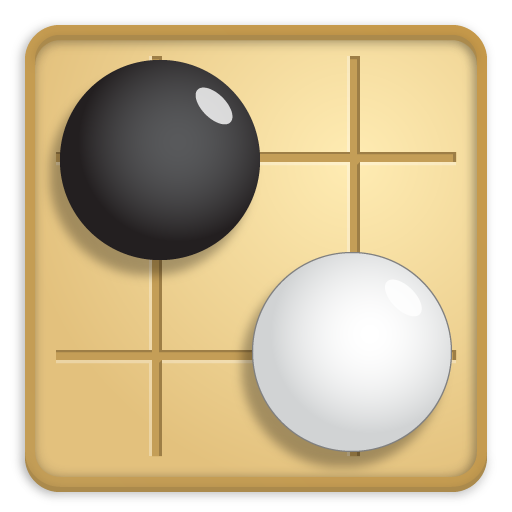
[Conception du travail 3](#_Toc166280958)

[Variables importantes 5](#_Toc166280959)

[Fonctions clés 6](#_Toc166280960)

[Fonctionnalitées implementées 8](#_Toc166280961)

[Conclusion 9](#_Toc166280962)



## Introduction

Ce rapport présente en détail le projet de développement d'un jeu de Gomoku en Python, réalisé dans le cadre du cours de programmation Python à l'EPF.

Nous avons travaillé en équipe pour concevoir et implémenter l’adaptation du jeu.

Nous avons accordé une grande attention aux détails pour susciter un minimum d'envie de découvrir le jeu, même avec une adaptation classique et simpliste. En termes de code, cependant, nous avons veillé à maintenir une propreté irréprochable.

## Conception du travail

Une image contenant texte, capture d’écran, Police

Description générée automatiquementLe programme est structuré en plusieurs parties distinctes, chacune étant responsable d'une fonctionnalité spécifique du jeu. Nous allons vous présenter la structure de ce programme :

Initialisation du jeu : L'utilisateur est accueilli par un menu principal affichant cinq options :

* Commencer une nouvelle partie
* Continuer une partie sauvegardée
* Consulter les règles du jeu
* Afficher les crédits (informations sur les auteurs du projet)
* Quitter le programme (bien qu'une commande "Stop" soit également disponible, nous avons inclus une fonctionnalité de sortie plus intuitive).

Boucle de jeu : Une fois la partie débutée, les joueurs sont invités à entrer leurs pseudonymes. Nous avons envisagé d'intégrer un mode de jeu contre une intelligence artificielle ainsi qu'un mode joueur contre joueur, mais avons préféré simplifier le jeu pour des raisons de temps (perte de temps avec l’interface graphique *Tkinter*) et de complexité. Les joueurs alternent pour placer leurs pions sur une grille, suivant une boucle où la grille est affichée à chaque tour, indiquant à qui le tour de jouer. La grille elle-même est inspirée du modèle utilisé dans un projet antérieur de bataille navale. Pour mettre fin à la partie, un joueur doit remporter la victoire et choisir de ne pas rejouer lorsqu'on lui demande.

Gestion des sauvegardes : Le programme permet de sauvegarder et de charger une partie en cours. Cependant, nous avons rencontré des difficultés avec la fonction de sauvegarde. Initialement, nous avions configuré le programme pour activer la sauvegarde dès qu'au moins un pion était placé sur la grille. Les problèmes semblent être liés à la boucle de la fonction "***Coords"***, ainsi qu'aux fonctions ***"Continuer\_partie"*** et ***"main\_partie"***. Nous avons également rencontré des difficultés à comprendre le principe de la sauvegarde de données au format JSON.

Affichage des règles et des crédits : Des fonctionnalités sont incluses pour afficher les règles du jeu et les crédits des créateurs du projet. Ces fonctionnalités sont relativement simples, se contentant d'afficher du texte.

Nous avons laissé des parties de codes concernant la sauvegarde des profils / état du jeu / score. Nous avons tout fait pour que les erreurs par le système (la console) ne s’affichent pas.

Notons aussi le fait que nous avions pris le temps de chercher à optimiser le code. Car la taille d’un programme est aujourd’hui un élément important dans les vrais jeux (ressources, etc..). Nous voulions montrer qu’un code peut être optimal lorsqu’il est léger.

Une image contenant capture d’écran, texte, carré, ligne

Description générée automatiquement

## Variables importantes

Au cours de notre programme, nous avons défini 20 variables nous permettant de simplifier le code ;

1. ***jr\_1 et jr\_2 :*** Dictionnaires représentant les joueurs avec leurs pseudonymes, numéro et score.
2. ***max\_tour :*** Variable représentant le nombre maximum de tours dans une partie.
3. ***Alphabet :*** Chaîne de caractères représentant les lettres de l'alphabet utilisées pour les coordonnées de la grille.
4. ***grille :*** Liste de listes représentant la grille de jeu.
5. ***s :*** Chaîne de caractères utilisée pour l'affichage de la grille.
6. ***line :*** Chaîne de caractères utilisée pour l'affichage de la grille.
7. ***nligne :*** Variable utilisée pour l'affichage de la grille.
8. ***saisie :*** Chaîne de caractères représentant l'entrée de l'utilisateur.
9. ***var :*** Variable utilisée pour le tirage aléatoire du premier joueur.
10. ***premier\_jr :*** Variable représentant le premier joueur.
11. ***der\_jr :*** Variable représentant le dernier joueur à avoir joué.
12. ***msg :*** Chaîne de caractères utilisée pour afficher un message.
13. ***Coordonees\_saisie :*** Chaîne de caractères représentant les coordonnées saisies par le joueur.
14. ***coords\_pos\_x et coords\_pos\_y :*** Variables représentant les coordonnées converties en nombres entiers.
15. ***pion\_posable :*** Variable indiquant si un pion peut être placé à une certaine position.
16. ***gagnant :*** Variable indiquant le numéro du joueur gagnant.
17. ***reponse :*** Chaîne de caractères représentant la réponse de l'utilisateur pour rejouer.
18. ***scores :*** Liste contenant les scores des deux joueurs.
19. ***rejouer :*** Variable indiquant si les joueurs veulent rejouer ou non.
20. ***player1 et player2 :*** Chaînes de caractères représentant les pseudonymes des joueurs lors de la création d'une nouvelle partie.
21. ***dossier\_gomoku :*** Chaîne de caractères représentant le nom du dossier pour la sauvegarde.
22. ***data :*** Liste contenant les données à sauvegarder dans le fichier JSON.
23. ***fichier\_path :*** Chaîne de caractères représentant le chemin du fichier de sauvegarde JSON.
24. ***fichier*** : Variable représentant le fichier de sauvegarde JSON.

## Fonctions clés

Voici la liste de nos 20 fonctions du programme Gomoku. Il n’y a pas d’auteurs spécifique à ces fonctions, car nous étions 3 à travailler dessus, via appel vidéo ;

1. ***afficher\_grille(grille)*** *:* Fonction pour afficher la grille de jeu.
2. ***premier\_tour(grille) :*** Fonction pour déterminer aléatoirement le premier joueur.
3. ***ordre(der\_jr, jr\_1, jr\_2) :*** Fonction pour définir l'ordre des joueurs.
4. ***Afficher\_msg(msg) :*** Fonction pour afficher un message.
5. ***Coords(Joue, grille) :*** Fonction pour vérifier les coordonnées saisies par le joueur et gérer les commandes spéciales (save, stop, help).
6. ***afficher\_aide() :*** Fonction pour afficher les règles du jeu.
7. ***Coords2Nums(pos)*** *:* Fonction pour convertir les coordonnées en lettres et chiffres en indices de la grille.
8. ***verif(grille, coordonee\_x, coordonee\_y) :*** Fonction pour vérifier si une case est vide.
9. ***tour(grille, der\_jr, Joue) :*** Fonction pour permettre à un joueur de jouer un tour.
10. ***condition\_pour\_gagner(grille, jr\_1, jr\_2, scores) :*** Fonction pour vérifier s'il y a un gagnant ou une égalité.
11. ***partie(grille, max\_tour, premier\_jr, jr\_1, jr\_2, joueur\_gagnant, scores, Joue) :*** Fonction principale pour exécuter une partie complète.
12. ***restart(joueur\_gagnant, scores)*** *:* Fonction pour demander aux joueurs s'ils veulent rejouer.
13. ***main() :*** Fonction principale du programme pour gérer le menu principal et les choix des joueurs.
14. ***main\_nouvelle\_partie() :*** Fonction pour gérer une nouvelle partie**.**
15. ***main\_continuer\_partie() :*** Fonction pour gérer la reprise d'une partie sauvegardée.
16. ***afficher\_credits() :*** Fonction pour afficher les crédits du jeu.
17. ***main\_partie() :*** Fonction principale pour gérer chaque partie et mettre à jour les scores.
18. ***init\_grille()*** *:* Fonction pour initialiser une grille vide.
19. ***sauvegarde(jr\_1, jr\_2, grille, Joue)* :** Fonction pour sauvegarder la partie dans unfichier JSON**.**
20. ***chargement(jr\_1, jr\_2, grille, Joue) :*** Fonction pour charger une partie à partir d'un fichier JSON.

## Fonctionnalitées implementées

|  |  |
| --- | --- |
| **Fonctionnalité** | **Etat** |
| Nouvelle partie | Fonctionnel |
| Continuer partie (à partir d’une sauvegarde) | Non fonctionnel |
| Afficher les règles du jeu | Fonctionnel |
| Afficher les crédits | Fonctionnel |
| Création des profils joueurs | Fonctionnel |
| Afficher la grille | Fonctionnel |
| Commande « Stop » pour quitter la partie | Fonctionnel |
| Commande « Save » pour sauvegarder la partie | Non fonctionnel |
| Interdiction de placer plus de 5 pions | Fonctionnel |
| Jeu, tour à tour | Fonctionnel |
| Conditions de victoire | Fonctionnel |
| Le premier pion placé au milieu | Fonctionnel |
| Hors-jeu | Fonctionnel |
| Match nul | Fonctionnel |
| Gestion des scores | Fonctionnel |
| Fonctionnalité de rejouer | Fonctionnel |
| Sauvegarde des profils joueurs | Non fonctionnel |
| Mode IA (robot intelligent) | Non implémentée |
| Interface graphique ++ (esthétique & bon fonctionnement) | Non implémentée |

## Conclusion

Nous considérons notre projet Gomoku comme une adaptation assez réussie du célèbre jeu de société en Python. Nous avons ajouté un maximum de détails, permettant une gestion efficace du jeu et assure une expérience utilisateur fluide. Nous avons réussi à implémenter avec succès à la fois les fonctionnalités de base et avancées, offrant ainsi aux joueurs une expérience complète et divertissante.

Ce projet python démontre notre capacité à concevoir, implémenter et documenter un programme fonctionnel et interactif. Cependant, nous notons qu'il nous manque encore la fonctionnalité de sauvegarde des joueurs et du score. Nous envisageons de nous concentrer sur l'interface graphique, car elle est essentielle pour susciter l'envie de jouer chez les joueurs. Une interface simple ne suffit pas à captiver l'utilisateur. De plus, nous envisageons d'explorer la création d'un mode de jeu avec une intelligence artificielle pour ajouter une dimension supplémentaire à l'expérience de jeu.

Projet disponible sur **GitHub** : <https://github.com/bryanthrmn/Gomoku/tree/main>

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement