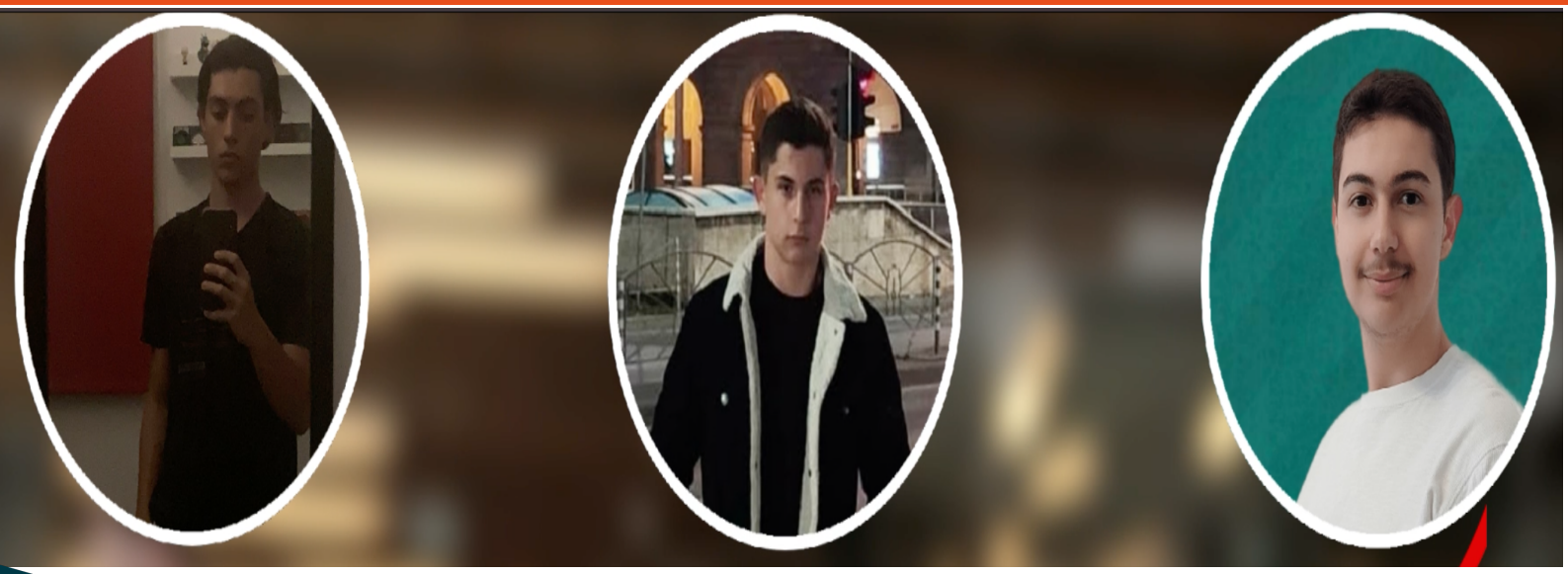
LES TROPHÉES NSI

Édition 2023

DOSSIER DE CANDIDATURE
PRÉSENTATION DU PROJET
HadjiChess



NOM DU PROJET : HadjiChess

Plan :

➢ **Membres du groupe :** Alexandre LEROUX, Naïm HARIB et SIMEOV Siméon

➢ **Professeur :** Monsieur HADJI

➢ PRÉSENTATION GÉNÉRALE :

Le projet consiste en la création d'un jeu d'échecs en Python orienté objet utilisant le module *Pygame*. L'objectif est de créer un jeu fonctionnel pour deux joueurs (locale) en utilisant les règles basiques des échecs : Le déplacement, manger des pièces, mettre en échec…L'ambition à plus long terme est de développer une intelligence artificielle qui jouera contre les joueurs et sera capable de s'entraîner en analysant les parties précédemment jouées, mais cette idée est décrite plus tard.
Le projet a été initié dans le cadre de nos cours de NSI : En première nous avons eu des projets à faire et étant chacun passionné, nous avons vite terminer les nôtres. Alors nous nous sommes lancés dans un autre projet qui est celui que nous présentons à ce concours. Bien que l'idée nous soit venue en fin d'année de première, ce jeu d'échecs a été réalisé tout au long de l'année de Terminale et finalisé il y a peu afin de pouvoir l'envoyer. Nous avons choisi le jeu d'échecs car en plus de la spécialité NSI, nous avons la spécialité Mathématique et que nous sommes férus de jeu de stratégie. Parmi la liste des jeux que nous avions choisis à la base (Tictactoe, Puissance 4, Reversi…), le jeu d'échecs était le plus avancé mais aussi le plus compliqué à réaliser (pour nous). L'objectif de la réalisation de ce jeu emblématique de la stratégie et de réflexion était de nous permettre de développer des compétences en logique mais avant tout de nous faire approfondir la manipulation de la programmation orienté objet. Au-delà de nos objectifs personnels, nous gardions en tête que ce jeu nous pourrons ensuite le présenter et que des réelles parties se passent dessus.

➢ ORGANISATION DU TRAVAIL :

Nous sommes trois élèves de terminale ayant choisi la spécialité NSI au lycée Sasserno de NICE. Notre équipe se compose d'Alexandre LEROUX, Naïm HARIB et SIMEOV Siméon, chacun motivé à continuer dans le domaine de l'informatique avec des ambitions définies.
Les tâches ont été réparties entre les membres de l'équipe par rapport aux compétences de chacun mais aussi à nos difficultés. En effet, nous avons réparti les taches en nous basant sur les difficultés de chacun et le manque de connaissance que nous avions dans certains domaines. Ainsi nous avons pu affronter nos obstacles et élargir nos connaissances tout en solidifiant noss compétences.
Alexandre s'est principalement occupé de la conception et du développement des pièces et de leurs représentations. Il a également contribué à la programmation des règles de prise de pièces et l'implémentation de toute la logique du jeu (tour par tour, début et fin de partie, mise à jour des données…).
Siméon s'est notamment chargé de la conception et du développement du plateau de jeu et de la représentation et coordination de ce dernier ; il s'est aussi occupé de la programmation des règles de mouvement des pièces et de la programmation des règles de déplacement des pièces.

Naïm s'est globalement occupé de la conception et du développement de l'interface graphique, de la mise en place et de la gestion des différents évènements (utilisation de la souris pour le déplacement des pièces, manger d'autres pièces et situations d'échec) et à la bonne inertie du jeu pour la prise en charge des différents fichiers.

Il est important de souligner que malgré la repartie des taches, un travail d'équipe sans relâche fut réalisé pour la réussite de ce projet. Aucun membre du groupe n'a réalisé tout seul les tâches

qui lui ont été destinées. Chacun d'entre nous a apporté une contribution sur le code d'autrui et les parties difficiles ont fussent pertinemment solutionnées en équipe. Le projet nous a donné l'occasion d'organiser des réunions et d'utiliser des outils et logiciel, en ligne (Replt.it, Notion, Discord…) ou non (VisualStudio Live Share, Slack, Google Workspace…) afin de coder et d'interagir ensemble, les règles de l'art !

➢ LES ÉTAPES DU PROJET :

Le projet a commencé par l'idée de créer un jeu d'échecs en Python orienté objet après avoir fini nos projets scolaires comme expliqué précédemment. Avant de se lancer dans la programmation et le code pur et dur, nous avons dû mettre en place une organisation : Nonobstant la distribution des tâches, nous avons, avec l'aide notre professeur, effectué un réel cahier des charges, indispensable pour chaque projet. Objectifs, délais, fonctionnalités à mettre en œuvre, contraintes techniques, tests, planning, maintenance, évolution entre autres. Tout y est passé ! Nous avons d'abord choisi de programmer chacun un fonctionnement de plateau de jeu afin d'ensuite comparer et prendre le meilleur. Mais voyant que nous n'arrivions pas à décider, nous sommes plutôt partis sur le choix de la répartition des tâches décrites dans la partie précédente. Après avoir mis en place de cahier des charges et rédigé nos « TODO List » personnels (Mouvements des pièces, règles du jeu, interface graphique, gestion des erreurs…). Nous nous sommes mis au travail en suivant les indications susmentionnées. D'abord nous avions l'idée de faire jouer contre un CPU qui aurait déplacé les pièces au hasard mais nous avons voté pour abandonner cette idée non ambitieuse. Nous voulions un jeu d'échec fonctionnel, alors nous avons préféré faire un jeu 1v1 en premier temps afin de s'assurer d'avoir la capacité de le faire. Nous avons plus-tard développer l'idée du joueur adverse non humain.

➢ FONCTIONNEMENT ET OPÉRATIONNALITÉ :

Actuellement, le projet est fonctionnel pour une partie d'échecs pour deux joueurs en local (tour par tour). Deux petits bug notoires ont été retrouvés en fin de projet et nous avons travaillé dessus afin de déployer un patch. Lorsque l'un des rois est attaqué, mis en position d'échec, et que l'on mange son assaillant ou une pièce adverse, la pièce essayant de défendre ne bouge pas alors que l'assaillant est tué et mis sur le côté. Alors le coup n'est pas compté mais cela vient de la fonctionnalité donnant l'impossibilité de bouger d'autres pièces lorsque le roi est en échec afin que la partie reprenne continue normalement. Dans certaines situations, lorsque le roi se met en échec, la partie prend fin (nous restons dans l'optique que dans une partie réelle, le roi ne doit et ne peut pas se mettre en échec).
Pour valider l'opérationnalité du projet et son fonctionnement, des tests ont été effectués à chaque modification et en boucle. On lançait simplement notre programme et on faisait des mouvements au hasard. Ensuite, nous est venu l'idée d'effectuer des *asserts* pour la vérification des fonctions et méthodes mais cela n'a pas finalisé. Nous avons finalement décidé de mettre en place des macros afin de jouer automatiquement des mouvements au démarrage du programme pour vérifier que tout était opérationnel et que chaque code des membres de l'équipe concordait.
Nous avons tous rencontré des difficultés liées à la complexité du jeu d'échecs et à la programmation orientée objet en Python. Des recherches et des expérimentations ont permis de surmonter ces difficultés. De plus, nous avons rencontré pas mal de problème au niveau de la gestion de notre temps et les tests. En effet, nous avions notre bac à préparer et obtenir et les tests ont pris une partie majeure dans notre temps avant la mise en place d'autres méthodes plus simples (macros & assertions). Afin de pallier notre problème de temps, nous avons simplement obtenu notre bac de NSI (avec briaud et les notes 19, 20 et 20 🏆), après quoi nous avons pu nous consacrer à plein temps au développement et au programme de NSI.

➢ OUVERTURE :

Actuellement c'est un jeu d'échecs 1v1 en local, donc sur le même PC. Le but est de trouver un moyen de faire un CPU simple qui jouera à la place d'un autre joueur. A terme, cela serait idéale de faire jouer ChatGPT grâce aux requêtes sur leurs API et au module python open-ai et request afin d'entraîner notre propre intelligence artificielle (min-max). Pour ça on pourrait héberger le script python sur un site prévu à cet effet afin que chacune des parties qui se déroule soit enregistrée et analysée par l'IA. Une IA qui aurait un modèle du « deep learning » lui donnant un pourcentage de réussite qu'elle modifiera au fur et à mesure des partie sur des caractères bien précis (la prise de pion, défense du roi, nombre d'échec sur le roi adverse …) lui permettant de réagir efficacement en fonction d'une situation. Nous pouvons aussi, encore une fois grâce au modules *request* et *socket*, mettre en place une partie qui se déroulerait à distance. Il reste beaucoup de chemin à faire mais c'est totalement envisageable surtout que nous aimerions prendre avec nous des plus jeunes élèves qui s'intéresse à la NSI et l'informatique en général afin qu'ils puissent contribuer au projet pour l'améliorer.

Pour finir et « analyser », nous sommes tous les trois unanime : Le projet est un franc succès grâce à notre entente et notre motivation. Le jeu est fluide et pour python nous trouvons le résultat de notre dur labeur. Si c'était à refaire nous le referons et si nous en avons l'occasion, on continuera à développer des petits projets !

DOCUMENTATION

• Spécifications fonctionnelles (guide d'utilisation, déroulé des étapes d'exécution, description des fonctionnalités et des paramètres)
• Spécifications techniques (architecture, langages et bibliothèques utilisés, matériel, choix techniques, format de stockage des données, etc)
• Illustrations, captures d'écran, etc

L'utilisation de notre programme ne présente rien de compliqué :
Si votre ordinateur dispose d'un environnement python et de quelques modules à installer, vous pourrez lancer et jouer avec facilité le jeu.

Afin que tout soit opérationnel vous pouvez utiliser l'installation rapide des prérequis grâce au fichier requirements.txt et la commande :
pip install-r requirements.txt

Si vous n'avez pas pip installé commencez d'abord par utiliser la commande suivante :
python get-pip.py

Maintenant que tout est près, vous pouvez lancer le fichier principal pour jouer !

Nous avons utilisé la POO, Programmation Orienté Objet, ce qui signifie que nous n'avons pas simplement ajouter des photos que nous déplaçons :
Grâce à ce type de programmation orienté objet, chaque pièce est une entité abstraite qui décrit les propriétés et le comportement de la pièce en général après avec instancier sa classe. C'est cela qui nous permet de gérer différemment chacune des pièces par rapport à son mouvement, son rôle et sa place sur l'échiquier.
Le jeu se lance dans une fenêtre de 640*640, un carré parfait pour une partie parfaite 😁
Nous avons adapté les couleurs afin que le jeu soit plus agréable et avons rajouté une espace de jeu réaliste : le plateau d'échec ne prend pas toute la place sur l'écran, si une pièce est mangée alors elle sera rangée sur le côté, comme sur une table lors du jeu en réalité et des indications spéciales ont été rajouté (chiffres et lettres pour labelliser les cases).

Comme le jeu se passe sur un GUI (Graphical User Interface) et non pas dans un CLI (Command Line Interpreter), nous avons consacré le terminal afin de récupérer des informations utiles au développement : déplacement, coordonnées de cases, utilitaires…
Afin d'afficher et faire fonctionner notre jeu et comme on le voit dans l'installation des modules et bibliothèques, nous utilisons le *pygame*. Dans nos phases de recherche pour le développement de ce jeu, nous cherchions un moteur de jeu puissant… Mais au final pourquoi faire ? Notre jeu d'échec sera peu coûteux en ressource et performant, alors nous nous sommes tournés vers pygame : Le module propose un certain nombre d'outils et de paramètre pour le développement de jeu 2D et peut être adapté à la POO, donc nous avons fait le choix de ne pas utiliser de moteur de jeu mais simplement *pygame* pour que notre jeu soit indépendant. Nous avons aussi utilisé le module *pickle* qui permet d'implémenter des protocoles binaires pour la sérialisation et désérialisation d'un objet (ici nos pièces) en python, « convertir un objet en flux d'octet ». Pour faire simple et mettre dans le contexte de notre projet, ce module est utilisé pour sauvegarder les pièces mangées pour convertir l'objet instancié en séquence de caractère (binaire).

Les assets utilisés pour les pièces du jeu sont sous licence CC0 (Creative Commons Zero)
La police d'écriture est sous license OFL (Open Font License)
Fenêtre de dénouement de partie (won.png) sous licence CC BY 4.0 et tout crédit apporté à greenpixels → https://greenpixels.itch.io/ avec modification apportés par différence à l'image trouvée ci-contre : https://harib-naim.com/won.png