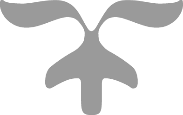


Polytech Casino

Bonjour et bienvenu au Polytech casino !



14 janvier 2018

Réalisé par :

Héloïse Faure et Théo Lemaire

Groupe 1

Sommaire :

1. Présentation du jeu

a) Le contrat (version +)

b) Les règles

2. Les stratégies

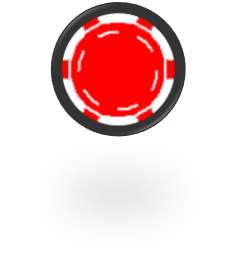
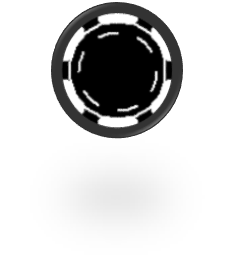
3. Structure du code

4. Le travail en Binôme

5. Synthèse du projet

a) Les difficultés Rencontrées

b) le travail qui reste à faire

**1. Présentation du jeu**

**Le contrat :**

Le contrat choisit est la version +.

**Description du projet :**

« Les allumettes seront représentées par un dessin fait avec turtle, et elles seront situées dans un décor.

Votre décor devra inclure les éléments suivants :

* une zone pour afficher ce que vient de jouer l'ordinateur,
* une zone pour afficher ce qui représente les allumettes (donc par exemple, les frites ou les arbres),
* un élément du décor devra se trouver à deux endroits différents (par exemple, 2 montagnes, ou deux chaises),
* un élément du décor (différent des allumettes) devra être répétitif (par exemple, une forêt qui contient plusieurs arbres, une tribune de foot qui contient plusieurs spectateurs ...).

En plus de la version de base, le joueur peut choisir de commencer ou non et la règle du jeu est tirée au hasard parmi une liste de règles de jeu possible. On entend par règle du jeu, ce que l'on peut retirer à chaque tour. Par exemple, la règle [1,2,3] est celle du jeu de base, [1,2,3,4] permet de retirer de 1 à 4 allumettes, la règle [1,2,6] permet de retirer 1,2 ou 6 allumettes.

Du point de vue du code, il faudra impérativement utiliser des modules. »

**Les règles :**

Prenez Place et venez jouer !

Les règles du jeu sont simples. Vous allez jouer contre l’ordinateur. Un nombre de carte tiré au hasard seront étalées sur la table du Casino, le but du jeu et de piocher la dernière.

Appuyez sur « entrer » quand vous êtes prêt à jouer !

Tout d'abord, vous pouvez choisir la difficulté de l’ordinateur, vous avez le choix entre un ordinateur faible et un ordinateur moyen. Suite à cela, le nombre de cartes autorisé à retirer vous seront affiché sur la tapis de jeu, ces règles sont tirées au hasard parmi une liste de règles prédéfinies. Les cartes s’affichent !

Choisissez ensuite si vous voulez commencer ou non.

Le jeu commence !

**Comment jouer ?**

Vous pouvez voir le nombre de carte disposé sur le tapis de jeu (sous l’inscription « Polytech casino »).

Entrez le nombre de carte que vous désirez prendre dans la fenêtre qui apparaît.

Vous allez ainsi voir le nombre de carte disparaître.

Successivement vient le tour de votre adversaire, l'ordinateur. Le nombre retirait par celui-ci se trouve en haut à droite.

Tour à tour, le nombre de carte diminue et la stratégie devient de plus en plus importante !

A vous d’élaborer une stratégie gagnante… et bonne chance !

**Les stratégies de jeux**

**2. Les stratégies**

Les règles présentes dans notre jeu sont : [1,2,3] ; [1,3,5] ; [2,4,6] ; [1,4,5]

Voici quelques stratégies afin de pouvoir faciliter votre victoire !

Procédons de manière méthodique. Pour le jeux classique les règles sont [1,2,3]. Ainsi pour assurer votre victoire vous devez procéder de la manière suivante :

Pour 4 cartes sur la table vous perdrez à coup sûr (dans le cas d’un ordi moyen)

Pour 5 cartes la table, vous devez prendre 1 carte pour en laisser 4 et gagner.

(…)

Ainsi, nous développons la stratégie suivante :

Vous devez absolument commencer !

Dans un premier temps, vous devais arrondir le nombre de carte sur la table à un multiple de 4 :

**Exemple** :

Il y a 27 cartes sur la table

Multiples de 4 : 4,8,12,16,20,24,28,32

Vous retirez donc 3 cartes pour arriver à 24, qui est un multiple de 4

Ensuite vous aller retirer :

3 cartes si l’ordinateur en a pris 1

2 cartes si l’ordinateur en a pris 2

1 carte si l’ordinateur en a pris 3

Vous prenez le complémentaire de 4, afin que, tour à tour, les cartes soient enlevées 4 par 4. A la fin il restera donc 4 cartes. Alors c’est au tour de l’ordinateur de jouer, il se trouve donc dans une impasse et vous gagnez !

**Exemple*:***

Il y a 17 cartes sur la table, vous commencez à jouer, et vous retirez donc une carte pour arriver à 16.

L’ordinateur -3 cartes et vous -1 carte, il en reste donc 12,

L’ordinateur -2 cartes et vous -2 cartes, il en reste donc 8,

L’ordinateur retire 1 carte et vous -3 cartes, il en reste 4,

L’ordinateur perd ! Et cela à tous les coups !

**Remarque :** Si le nombre de cartes sur la table est déjà un multiple de 4 et que vous commencez. Alors jouez aléatoirement et vous allez chercher à retrouver un multiple de 4 au prochain tour.

De même si vous choisissez de ne pas commencer mais que l’ordinateur choisit un nombre de cartes et que le nombre de cartes restant est un multiple de 4 .

**Exemple :**

Il y a 18 cartes sur la table et l’ordinateur commence (celui-ci n’a pas de stratégie)

L’Ordinateur prend 2 cartes, il en reste 16 (vous ne pouvez donc pas appliquer la méthode)

Vous prenez un nombre quelconque (3 par exemple) il en reste 13

L’ordinateur en prend 2 aléatoirement, il en reste donc 11

C’est à vous de retrouver le multiple de 4 en prenant 3 cartes. Il en reste 8 et vous reprenez la stratégie et vous avez gagné.

A présent à vous de l’élaborer d’autre stratégies sur le même raisonnement pour les autres règles !

**Démarrage du projet :**

**3. Structure du code**

**Début du projet**

Lors de la première séance nous avons codé très simplement (sans fonction) la première version du jeu. Puis la semaine d’après nous nous sommes retrouvés pour commencer correctement le projet.

Dans un premier temps nous avons écrit tout le code dans une même fonction, mais cela était assez illisible et difficile à comprendre. Ainsi nous avons pensé à avoir une structure de code plus rigoureuse et soigné. Pour cela, il fallait à apprendre à coder proprement. Pour cela nous avons regardé sur internet des méthodes pour coder sur python et nous avons également demandé des conseils à des Peip2.C’est alors que nous avons décidé de décomposer le code en plusieurs fonctions aillant une page de code python respective, afin de soigner la présentation, de facilité la lisibilité pour mieux coder.

Nous avons appris à faire des appels de fonctions en important la fonction dans une autre page de code (from fonction import\*). Ainsi nous nous somme répartie plusieurs fonctions à coder.

**Déroulement du projet**

Au fur et à mesure, nous avons créé des fonctions pour décomposer chaque partie de notre code. En commençant par les fonctions ‘regle\_jeu’, ‘jeu\_ordi’ et ‘jeu\_joueur’ que nous avons importé dans notre programme principal ‘allumette’.

Nous avons ensuite décidé le thème de notre interface graphique : un tapis de jeu de Casino. De là nous avons créé les fonctions ‘tapis’ et ‘carte’.

Nous avons travaillé simultanément sur les fonctions ci-dessus afin de les améliorer. Nous avons décidé de terminer complétement notre interface graphique en créant la fonction ‘jetons’ et en regroupant toutes les fonctions du décor dans une autre fonction, ‘interface\_graphique’.

Une fois l’interface graphique terminée, nous avons finis les autres fonctions du programme et importé la fonction ‘interface\_graphique’ dans le programme principal ‘allumette’.

Le jeu fonctionne à présent, notre décor est prêt, cependant nous jouons toujours dans le terminal de python. Il faut à présent relier le décor avec notre programme ce qui fut le plus long et le plus difficile.

La fonction ‘pile\_carte’ nous a permis d’afficher les cartes sur le tapis de jeu, nous avons relié cette fonction avec le nombre de cartes tiré au hasard. Puis lorsque l’on jouait depuis le terminal, on pouvait voir sur le décor que les cartes apparaissaient, ça fonctionnait !

Suite à cela nous avons créé la fonction ‘suppression’. Puis pendant que l’un faisait une nouvelle fonction ‘main’ pour le décor, l’autre s’est occupé de relier le terminal au tapis de jeu, on a donc créé les fonctions ‘résultats’, ‘main’ et ‘affichage\_ordi’, puis modifié l’ensemble du programme grâce aux fonctions ‘numinput’ et ‘textinput’, des modules de turtle.

**Les modules :**

Nous avons utilisé des fonctions principalement. Les fonctions ‘jeu\_ordi’ et ‘jeu\_joueur’ ont comme paramètre le nombre de cartes et également la règle à suivre (jeu\_nom (x,regle)), ces fonctions return à la fin le nombre de cartes qu’il reste une fois qu’ils ont joué.

Dans nos fonctions nous avons utilisé plusieurs modules : par exemple le module math ou random. Un en particulier est le module time que nous avons utilisé afin de pouvoir mettre du temps avant que l’ordinateur retire des cartes.

Nous avons utilisé une structure de données qui est la liste afin de pouvoir définir les règles du jeu. On l’utilise dans les fonctions ‘jeu\_noms’ ce qui permet de faire respecter les règles aux joueurs. Il y a aussi ne nombreuses boucles ‘while’ ainsi que des ‘if,elif,else’.

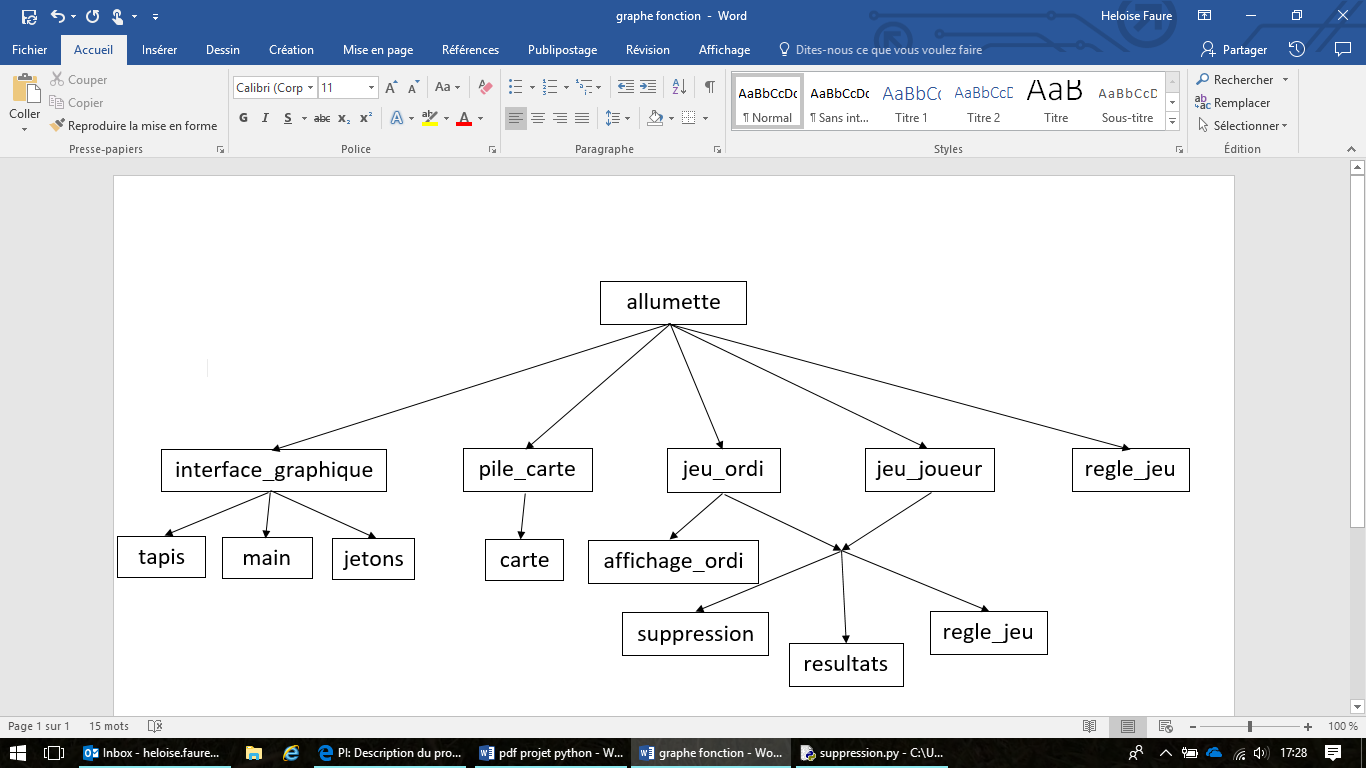
**L’interface graphique :**

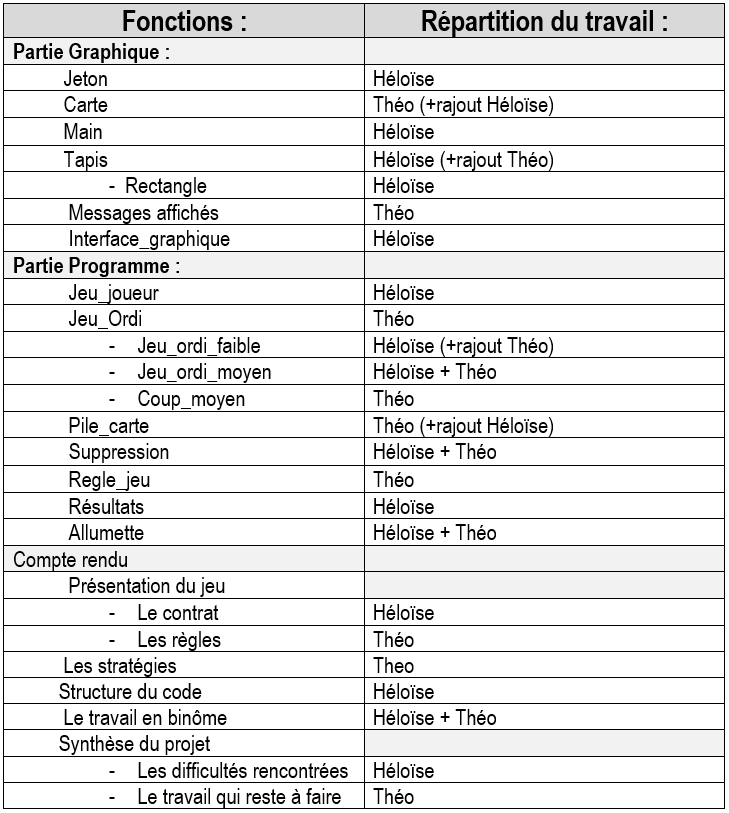
Pour pouvoir réaliser les fonctions de l’interface graphique nous avons importé le module turtle.Chaque fonction de l’interface graphique est sous la forme (x, y, taille) afin de pouvoir adapter sa taille et son placement, sur le tapis de jeu et en fonction de la taille de la fenêtre si besoin.

Certaines fonctions contiennent des paramètres en plus comme la couleur ou la circonférence du cercle.

Toutes les fonctions de l’interface graphique sont regroupées dans une fonction ‘interface\_graphique\_fixe’ ce qui permet de pouvoir tout tracer en une seule fois. Cette fonction est appelée au tout début du programme dans le programme principal ‘allumette’.

***Voilà un schéma qui montre comment toutes les fonctions sont reliées entre elles :***



****

**4. Le travail en Binôme**

Vous pouvez lire ci-dessus le travail principal effectué par les membres du binôme. Cependant, chaque partie du code et du compte rendu a été lu et remis en forme par les deux membres.

**5. Synthèse du projet**

**Les difficultés rencontrées :**

**Pour le programme :**

- La décomposition du code en plusieurs fonction

- Le codage des règles du jeu (cas d’erreur avec nombre négatif, non autorisé, etc..)

- Relation entre terminal de Python et l’interface Graphique

Au tout début, c’était difficile de savoir comment décomposer le code judicieusement en plusieurs fonctions. Nous avons donc procédé par étape en modifiant et recommençant les fonctions plusieurs fois.

Nous avons eu du mal à comprendre exactement ce que vous attendiez pour les règles du jeu et comment il faillait les coder. Nous avons donc créé plusieurs fonctions ‘règle’. De plus il était difficile de gérer tous les cas d’erreur quand un des joueurs retirer les cartes (nombres négatifs, nombres non autorisés, quand le nombre de cartes autorisé à retirer était plus grand que le nombre de cartes restant…)

La plus grosse difficulté que nous avons rencontrée était de relier le terminal de python à l’interface graphique, cela nous a pris une bonne partie des vacances. N’aillant pas trouvé comment faire avec les modules de python, on a pensé à utiliser un module tkinter, puis finalement nous avons trouvé une autre méthode sans importer d’autre module.

**Pour l’interface graphique :**

* Que les fonctions pour chaque module ne contredisent pas les autres fonctions (ex : orientation des cartes, placement décalé,…) Ainsi l’utilisation du goto() a été remplacé par des forward() et nos module se terminent par des seth(0) pour rectifier l’orientation.
* Les paramètres de tout le décors selon (x,y,taille), en posant des calcul de proportions. Ceci représente l’élément le plus complexe et important de cette partie.
* La suppression de cartes : refaire une pile de carte en enlevant le nombre de carte sélectionner ou effacer le nombre de carte sélectionné simplement
* La vitesse d’exécution : le speed(« fastest ») ne fut pas assez rapide à notre gout, nous avons finalement trouver tracer(0) après de longue recherche.

Concernant l’interface graphique nous avons eu des difficultés à créer les fonctions pour que chaque module précédent n’influence pas le module d’après. En effet, lors de la création d’un nouveau module, celui-ci était mal orienté et mal placé. On s’est vite rendu compte qu’il ne fallait pas utilisé des goto() et forward() comme tel, on a donc créé nos modules avec, au début, « up() , goto(x,y), down() » et à la fin, seth(0) pour remettre l’angle dans sa position initiale.

De même pour les modules de l’interface graphique, nous avons dû les reparamétrer avec (x,y,taille) pour pouvoir les modifier et les placer comme on le voulait. Pour la main en particulier, nous avons dû prendre un papier à part pour faire des calculs de proportionnalité car c’était la partie de l’interface graphique la plus difficile à coder, à chaque fois, les tailles des parties de la main n’allaient pas. Cela nous a pris du temps.

Pour la suppression des cartes, on s’est longuement demandé s’il fallait que l’on return la fonction ’pile\_carte’ moins le nombre de cartes enlevées par l’ordi et le joueur, ou bien si on supprimait les cartes une par une grâce à une fonction qui dessine des rectangles verts. Après avoir essayé de coder les deux, nous avons finis par choisir la deuxième option qui nous paraissait moins compliquée.

Enfin, la vitesse d’exécution des modules de l’interface graphique fut un problème constant car nous avions le module ‘speed(« fastest »)’ mais cela ne suffisait pas, nous voulions une vitesse instantanée. Après maintes recherches nous avons trouvé le module ‘tracer(0)’ mais cela enlève directement les cartes sur le tapis ce que nous ne voulons pas.

***Le travail qui reste à faire :***

* A la fin du jeu, demander si le joueur voudrait refaire une autre partie.
* Mettre un « timeur » pour qu’il y en environ 3 secondes avant que l’ordinateur joue, ce que nous avons déjà essayé de mettre mais celui-ci était déficiant.
* Au lieu de taper le nombre de cartes à retirer dans une fenêtre, insérer des boutons pour chaque règle, sur le tapis de jeu et cliquer dessus pour retirer un nombre de cartes.
* Il faudrait également qu’on fasse disparaitre les cartes une par une et non par paquet choisit. Mais cela pose un problème, une contradiction, avec le module python ‘tracer(0)’. Il faudrait le positionner au bon emplacement, nous ne l’avons pas trouvé.
* Enfin, nous voulions que les deux mains dans les coins se déplacent et ramassent elles-mêmes les cartes, mais c’était ambitieux de notre part étant donné le délai imposé.