Nom du projet :

**Jeu de carte Ronda**

Description du jeu Ronda:

La **Ronda** est un jeu de cartes qui se joue avec des [cartes espagnoles](https://fr.wikipedia.org/wiki/Jeu_de_cartes_espagnol): les suites sont coupes, épées, bâtons, deniers en utilisant quarante cartes. On trouve des cartes numérotées de 1 à 7, et trois autres cartes : le Valet (10), le Cavalier (11) et le Roi (12)). On remarquera qu'il « manque » les 8 et 9.

Le jeu se joue à deux, trois ou quatre joueurs en constituant dans le cas de 4 joueurs 2 équipes qui jouent en commun et dont les joueurs sont assis l'un en face de l'autre avec des jetons.

Pour notre programme est implémenté à la façon de jouer avec l’ordinateur.

On commence par distribuer 4 différentes cartes pour la table, 3 pour joueur, et 3 pour ordi.

Joueur et ordi commencent par vérifier leurs 3 cartes s’il a deux mêmes numéros on dit il a une Ronda si les 3 cartes ont mêmes numéros on dit qu’il a tringa

Qui a Ronda gagne 1 point, tringa 5 points

Cas ou les deux ont Ronda qui a le plus grand numéro gagne 2 points (même numéro 1 point chacun)

Cas ou les deux ont tringa qui a le plus grand numéro gagne 10 points (même numéro 5 point chacun)

Cas Ronda avec Tringa : qui a Tringa gagne 6 points peux import les numéros.

Joueur ou ordi vérifier les cartes sur la table s’il a un même numéro avec ses cartes il ramasse les deux cartes qui ont les mêmes numéros et tous les numéros qui le suit dans la table. Après cette opération il ne reste aucune carte sur la table c’est Missa il gagne un point. Si adversaire a posé une carte sur la table et tu as le même numéro tu la prends avec suivie s’il en a et tu gagnes 1point c’est Essti.

Si ses cartes n’ont pas de même numéro sur la table il jette une carte sur la table et donne le tour de jouer au suivant.

Après passer les tours on distribuer 3 nouvelles cartes chacun.

Ainsi de suite jusqu’à la fin du paquet de 40. Fin d’une partie de jeu.

On compte les cartes ramassées de chacun qui a plus que 20 cartes gagnent la différence en point

Example joueur a 26 carte et ordi 14, le joueur gagne 6 points.

On continue les parties l’équipe qui totalise la somme de 41 points sera la gagnante.

Description du programme:

**Les classes**

**class** **Card:**

Constructeur \_\_init\_\_(suit, Rank, label) reçoit 3 attributs :

Suit pour suite

Rank pour numéro

Label pour savoir le propriétaire de la carte utilisé pour Essti et Missa

Méthode \_\_str\_\_() pour affichage de carte sous forme Numero\_Suit

Méthode \_\_lt\_\_(autre carte) reçoit une autre carte comme paramètre retourne le plus petit Rank entre les deux

Méthode \_\_gt\_\_(autre carte) reçoit une autre carte comme paramètre retourne le plus grand Rank entre les deux

Méthode \_\_eq\_\_(autre carte) reçoit une autre carte comme paramètre retourne s’elles sont égaux.

**Class** **Paquet**:

Constructeur \_\_init\_\_() pour production du paquet de 40 cartes avec les listes suivantes :

suit\_names = ["Coupe", "Epee", "Baton", "Denier"]

rank\_names = [1, 2, 3, 4, 5, 6, 7, 10, 11, 12]

Retourne un paquet sous forme d’une liste de 40 objets cartes

Methode \_\_str\_\_() retourne la liste avec des lignes de 10 cartes pour affichage.

Methode brasser () utilise fonction shuffle () importée de module Random pour brasser aléatoirement la liste du paquet.

Methode add\_card(carte) recoit un objet carte en paramètre et l’ajoute à la liste.

Methode remove\_card(carte) reçoit un objet carte en paramètre et l’enlevé de la liste.

Methode pop\_card(index) reçoit index de la liste et enlevé objet carte de la liste à cet index s’il n’y a pas d’index enlevé la première.

Methode leng\_cards () retourne la longueur de la liste.

Methode carte(index) retourne la carte dans la liste à cet index

Methode move\_one\_card (carte, autre) transfère une carte d’un paquet à une autre

Methode move\_cards (autre, label, n=1) transfère un nombre de cartes d’un paquet à une autre.

Methode move\_cards\_no\_dbl (self, autre, n=1) transfère un nombre de cartes non doublet d’une paquet à une autre

**Class Hand (Paquet)** une classe enfant de Paquet

Constructeur \_\_init\_\_() avec 3 attributs liste des cartes, total des points gagnées et liste des cartes gagnées.

Methode verification () retourne dans un Hand s’il y a des doubles ou triple Rank, va être utilisée pour Ronda et Tringa.

Methode liste\_gain () retourne la longueur de la liste cartes gagnées.

Methode pointage\_randa\_tringa(autre) compare deux Hand et retourne les points gagnés de chacun

**Les fonctions du programme**

Dans une fonction principale jeu () :

Fonction affichage () affiche la page du jeu avec les détails.

Fonction distribuer () distribue les cartes du paquet 3 carte au joueur et ordi chaque tour et 4 cartes une fois au début pour la table.

Fonction sequence (card, table) vérifier l’existence Rank d’une carte dans la table s’il existe il retourne toutes les cartes ayant les numéros suivants dans un objet hand et le reste de la table dans une autre liste

Fonction tour\_ordi (ordi, table) appelée quand le tour de l’ordi arrive

Fonction tour (carte, table) reçoit une carte et appelée quand le tour du joueur arrive

Fonction parti (ordi, joueur, table) qui retourne les points gagnés de chacun

Dans cette fonction le programme déclenche une grande boucle :

On vérifier si la liste du paquet est vide sinon on entre dans une autre boucle en vérifiant si toutes les cartes du joueur et ordi sont jouées sinon :

qui affiche avec Affichage ()

* Les cartes de la table,
* Les cartes d’ordi (normalement ils faut qu’ils soient cachés mais seulement pour la démo)
* Les 3 cartes du joueur chacune a cote d’un choix.

Demande au joueur de donner son choix on appelle les fonctions tour () et tour\_ordi () qui vont faire la vérification sur les cartes de la table pour ramasser les mêmes cartes et leur suivies, les missa et les essti.

Après le sorite de cette boucle on vérifier les points si quelqu’un à dépasser 41 points le jeu est terminer ou toutes les cartes du joueur et ordi sont jouées sinon on fait une autre distribution et appelle aux fonctions tour () et tour\_ordi () une autre fois.

**Programme**

Notre programme est un jeu entre joueur et ordi.

Début de programme commence par afficher un menu avec 3 choix

1. Partie simple (qui a plus du point après la fin d’une partie sera gagnant.)
2. Jeu complet (on joue plusieurs parties jusqu’à ou le joueur ou ordi a 41 points)
3. Quitter le jeu

Avec une boucle qui affiche le même menu jusqu’à choisir 3

**Partie Simple :**

On appelle la fonction parti () qui retourne les points de joueur et ordi en vérifiant le plus grand pour savoir le gagnant

On afficher avec affichage () le résultat

Dans la boucle interne :

A la fin de la partie on compte les points qui a plus du point gagne.

**Jeu complet :** une autre boucle qui répète la fonction parti () jusqu’à l’un des points dépasse 41. Entre la répétition de parti () on affiche le score de chaque partie.

**Cas de graphique :**

on ajoute

**Class Affichage:**

Constricteur \_\_init\_\_() avec les attributs :

* paquet, ordi, joueur,
* carte1, carte2, carte3, table
* table\_image, ordi\_image, image1 image3, dernier
* frame\_table, frame\_ordi, frame\_joeur1, frame\_table, frame\_ordi, lbl\_points

Methode change() pour changer les images

Methode clear\_frame() pour vider les cadres

Methode clique\_choix() appelle tour() et tour\_ordi()

Methode affiche\_initial() pour afficher la fenetre avec les images et les details

Installation :

Ce projet ne nécessite aucune installation spéciale et il est simple à utiliser.

Utilisation :

L’utilisation de ce programme est facile, en l’exécutant il y a un menu affiché qui vous guide à choisir une partie simple ou jeu complet.

Contribution :

Puisque ce programme utilise des nombres il a besoin d’une vérification au niveau des entrées pour les positifs et les négatives et au niveau des sorties pour la division par zéro qui déclenche une erreur. Et je suis ouvert pour toute contribution.

Auteurs et reconnaissances :

Auteur Mr : Monir Rafi

État du projet :

Projet Terminé.