NSI

Carte : Class

attribut :

-numero (1-14 pour carte normal ; 0-21 pour les atouts)

-type (‘co’,’ca’,’pi’,’tr’,’at’)

Methode \_\_str\_\_ :

Retourne le type suivi d’une lettre de l’alphabet de A-V selon le numéro (A :0 ;B :1 ;… V :21)

(utilisé comme cela pour passer de str à objet et de objet à str le plus facilement)

Interface client serv

Tas : Class

Attribut :

-Cartes (list de Carte)

Methode :

-toList :>list[str]

-addCarte(Carte)-> None

-addCartestr(str)->None

-removeCarte(Carte)->None

-removeCartestr(str)->None

Joueur : Class (extend Tas)

Attribut :

Nom : str

Carte : list[Carte]

Pris : bool

4 client web

Un serv python

Interaction TCP/IP (pas UDP) Transfert d’obj json type :

Serv->Joueur

Pour le jeu t=game :

Op 1

{‘Carte’ : [‘atA’,’coC’,…(les cartes du joueur)],’Jeu’ :[‘piD’,’piH’ (Carte disposé et mise en jeu sur la table)]}

Pour la distribution t=dsip :

Op 2

{‘Carte’ :[…], Joueurs :{str :’g’, :str : ‘h’,str : ‘d’,str :’b’} }

Pour l’enchere t=ench:

Op 3

{Joueur :str,niveau: int (niveau d’enchere ; 0 s’il n’y prend pas)}

Fin d’enchere :

Op 4

{gagnant : str, ‘chien’ :[str]}

Pour les tour :

Op 5

{‘joueur’ :str, ‘carte’ :carte}

Pour la fin de tour :

Op 6

{‘Jeu’ : […],gagnant : str}

Pour la fin de jeu :

Op 7

{gagnant : [Joueur],gagnantPoints :int,perdantPoint : int}

Erreur Op 10

{‘err’ :str,action :str}

Joueur-> Serv :

Op 1

Pour premier msg

{‘nom’ = nom}

Pour enchere t=penc

Op 2

{‘niveau’ :int}

Pour quand pris t=ppri

Op 3

{‘chien’ :[str]}

Pour quand tu joues t=pplay

Op 4

{‘Carte’ :str}