IFB_projet V1.0

Généré par Doxygen 1.8.18

1 Index des structures de données		1
1.1 Structures de données	 	. 1
2 Index des fichiers		3
2.1 Liste des fichiers	 	. 3
3 Documentation des structures de données		5
3.1 Référence de la structure Carte	 	. 5
3.1.1 Description détaillée	 	. 5
3.1.2 Documentation des champs	 	. 5
3.1.2.1 couleur	 	. 5
3.1.2.2 valeur	 	. 6
3.2 Référence de la structure Contrat	 	. 6
3.2.1 Description détaillée	 	. 6
3.2.2 Documentation des champs	 	. 6
3.2.2.1 atout	 	. 6
3.2.2.2 coinche	 	. 7
3.2.2.3 nbPoint	 	. 7
3.2.2.4 preneur	 	. 7
4 Documentation des fichiers		9
4.1 Référence du fichier acquisition.c	 	. 9
4.2 Référence du fichier acquisition.h	 	. 9
4.2.1 Description détaillée		
4.2.2 Documentation des fonctions	 	. 10
4.2.2.1 acquisitionEntierAvecMessage()	 	. 10
4.2.2.2 acquisitionEntierSansMessage()		
4.2.2.3 acquisitionEntierSansMessageAvecConsigne()		
4.2.2.4 acquisitionEntierSecurise()		
4.2.2.5 acquisitionOuiNonSansMessage()		
4.2.2.6 acquisitionPseudoAvecMessage()		
4.3 Référence du fichier affichage.c		
4.3.1 Description détaillée		
4.3.2 Documentation des fonctions		
4.3.2.1 afficheContrat()		
4.3.2.2 afficheInterfacePli()		
4.3.2.3 afficheMain()		
4.3.2.4 afficheMenuPrincipal()		
4.3.2.5 afficheMenuSelection()		
4.3.2.6 afficheSousMenus()		
4.3.2.7 modifieTailleFenetre()		
4.3.2.8 proposeContratUtilisateur()		
4.4 Référence du fichier affichage.h	 	. 17

4.4.1 Description detaillee	1/
4.4.2 Documentation des fonctions	18
4.4.2.1 afficheContrat()	18
4.4.2.2 afficheInterfacePli()	18
4.4.2.3 afficheMain()	19
4.4.2.4 afficheMenuPrincipal()	20
4.4.2.5 afficheMenuSelection()	20
4.4.2.6 afficheSousMenus()	20
4.4.2.7 modifieTailleFenetre()	21
4.4.2.8 proposeContratUtilisateur()	21
4.5 Référence du fichier autre.c	22
4.5.1 Description détaillée	22
4.5.2 Documentation des fonctions	23
4.5.2.1 ajusteEchelle()	23
4.5.2.2 joue1000Partie()	23
4.5.2.3 joueurSuivant()	24
4.5.2.4 nbAleatoire()	24
4.5.2.5 pointPli()	24
4.5.2.6 setContrat()	25
4.6 Référence du fichier autre.h	25
4.6.1 Description détaillée	26
4.6.2 Documentation des fonctions	26
4.6.2.1 ajusteEchelle()	26
4.6.2.2 joue1000Partie()	27
4.6.2.3 joueurSuivant()	27
4.6.2.4 nbAleatoire()	27
4.6.2.5 pointPli()	28
4.6.2.6 setContrat()	28
4.7 Référence du fichier formatageChaine.c	29
4.7.1 Description détaillée	29
4.7.2 Documentation des fonctions	30
4.7.2.1 aligneModifieChaine()	30
4.7.2.2 centreChaine()	30
4.7.2.3 centreModifieChaine()	31
4.7.2.4 decoupeChaine()	31
4.7.2.5 formateCarte()	31
4.7.2.6 formateContrat()	32
4.7.2.7 formatePseudo()	33
4.7.2.8 genereMessage()	33
4.7.2.9 rempliEspace()	34
4.7.2.10 stockeInfoCarte()	34
4.8 Référence du fichier formatageChaine.h	35

4.8.1 Description détaillée	. 35
4.8.2 Documentation des fonctions	. 35
4.8.2.1 aligneModifieChaine()	. 35
4.8.2.2 centreChaine()	. 36
4.8.2.3 centreModifieChaine()	. 36
4.8.2.4 decoupeChaine()	. 37
4.8.2.5 formateCarte()	. 37
4.8.2.6 formateContrat()	. 38
4.8.2.7 formatePseudo()	. 38
4.8.2.8 genereMessage()	. 39
4.8.2.9 rempliEspace()	. 39
4.8.2.10 stockeInfoCarte()	. 40
4.9 Référence du fichier general.c	. 40
4.9.1 Description détaillée	. 41
4.9.2 Documentation des fonctions	. 41
4.9.2.1 annonceContrat()	. 41
4.9.2.2 calculPointManche()	. 42
4.9.2.3 initialisation()	. 43
4.9.2.4 manche()	. 43
4.9.2.5 menuPrincipal()	. 44
4.9.2.6 nouvellePartie()	. 44
4.9.2.7 pli()	. 45
4.9.2.8 poseCarte()	. 46
4.9.2.9 proposeContrat()	. 46
4.10 Référence du fichier general.h	. 47
4.10.1 Description détaillée	. 47
4.10.2 Documentation des fonctions	. 48
4.10.2.1 annonceContrat()	. 48
4.10.2.2 calculPointManche()	. 48
4.10.2.3 initialisation()	. 49
4.10.2.4 manche()	. 49
4.10.2.5 menuPrincipal()	. 50
4.10.2.6 nouvellePartie()	. 50
4.10.2.7 pli()	. 51
4.10.2.8 poseCarte()	. 52
4.10.2.9 proposeContrat()	. 52
4.11 Référence du fichier gestionCarte.c	. 53
4.11.1 Description détaillée	. 54
4.11.2 Documentation des fonctions	. 54
4.11.2.1 cartePlaceAvant()	. 54
4.11.2.2 carteValide()	. 55
4.11.2.3 distribueCarte()	. 55

4.11.2.4 forceCarte()	56
4.11.2.5 rechercheAnnonce()	56
4.11.2.6 recherCarte()	57
4.11.2.7 recherCarteSuperieur()	57
4.11.2.8 setCarte()	59
4.11.2.9 sommeForceCarte()	59
4.11.2.10 supprimeCarte()	60
4.11.2.11 trieCarte()	60
4.11.2.12 vainqueurPli()	61
4.12 Référence du fichier gestionCarte.h	61
4.12.1 Description détaillée	62
4.12.2 Documentation des fonctions	62
4.12.2.1 cartePlaceAvant()	62
4.12.2.2 carteValide()	63
4.12.2.3 distribueCarte()	64
4.12.2.4 forceCarte()	64
4.12.2.5 rechercheAnnonce()	65
4.12.2.6 recherCarte()	65
4.12.2.7 recherCarteSuperieur()	66
4.12.2.8 setCarte()	66
4.12.2.9 sommeForceCarte()	67
4.12.2.10 supprimeCarte()	67
4.12.2.11 trieCarte()	68
4.12.2.12 vainqueurPli()	68
4.13 Référence du fichier gestionFichier.c	69
4.13.1 Description détaillée	69
4.13.2 Documentation des fonctions	69
4.13.2.1 ecrireLeaderboard()	69
4.13.2.2 ecrireStatistique()	70
4.13.2.3 ecriturePseudo()	70
4.14 Référence du fichier gestionFichier.h	71
4.14.1 Description détaillée	71
4.14.2 Documentation des fonctions	71
4.14.2.1 ecrireLeaderboard()	71
4.14.2.2 ecrireStatistique()	72
4.14.2.3 ecriturePseudo()	72
4.15 Référence du fichier ia.c	73
4.15.1 Description détaillée	73
4.15.2 Documentation des fonctions	73
4.15.2.1 choixCartelA()	73
4.15.2.2 proposeContratla()	74
4.16 Référence du fichier la h	74

4.16.1 Description détaillée	75
4.16.2 Documentation des fonctions	75
4.16.2.1 choixCarteIA()	75
4.16.2.2 proposeContratla()	76
4.17 Référence du fichier main.c	76
4.17.1 Description détaillée	77
4.17.2 Documentation des fonctions	77
4.17.2.1 main()	77
4.18 Référence du fichier main.h	78
4.18.1 Description détaillée	79
4.18.2 Documentation des macros	79
4.18.2.1 DEBUG_MODE	79
4.18.2.2 MODE_1_MANCHE	80
4.18.2.3 NB_CARRACTERE_LEADERBOARD	80
4.18.2.4 NB_CARRACTERE_SCORE	80
4.18.2.5 NB_JOUEUR	80
4.18.2.6 NB_TOATAL_CARTE	80
4.18.2.7 NIVEAU_IA	80
4.18.2.8 POSITION_NB_MANCHES_POUR_GAGNER	81
4.18.2.9 POSITION_NB_VICTOIRE	81
4.18.2.10 POSITION_RECORD_VICTOIRE	81
4.18.2.11 POSITION_SCORE_MAX	81
4.18.2.12 TAILLE_MAXI_COULEUR	81
4.18.2.13 TAILLE_MAXI_MESSAGE	81
4.18.2.14 TAILLE_MAXI_PESEUDO	82
4.18.3 Documentation des définitions de type	82
4.18.3.1 Carte	82
4.18.3.2 Coinche	82
4.18.3.3 Contrat	82
4.18.3.4 Couleur	82
4.18.3.5 Joueur	82
4.18.3.6 NbPoint	82
4.18.3.7 TypeMessage	83
4.18.3.8 Valeur	83
4.18.4 Documentation du type de l'énumération	83
4.18.4.1 Coinche	83
4.18.4.2 Couleur	83
4.18.4.3 Joueur	84
4.18.4.4 NbPoint	84
4.18.4.5 TypeMessage	85
4.18.4.6 Valeur	85
4.19 Référence du fichier sous-menus.c	85

Index

4.19.1 Documentation des fonctions	. 86
4.19.1.1 leaderboard()	. 86
4.19.1.2 parametre()	. 86
4.19.1.3 statistiqueJoueur()	. 87
4.20 Référence du fichier sous-menus.h	. 87
4.20.1 Documentation des fonctions	. 87
4.20.1.1 leaderboard()	. 87
4.20.1.2 parametre()	. 88
4.20.1.3 statistiqueJoueur()	. 88
4.21 Référence du fichier tableau.c	. 89
4.21.1 Description détaillée	. 89
4.21.2 Documentation des fonctions	. 89
4.21.2.1 acquiertTableau()	. 89
4.21.2.2 afficheTableau()	. 90
4.21.2.3 aleatoireTableau()	. 90
4.21.2.4 constanteTableau()	. 91
4.21.2.5 copie()	. 91
4.21.2.6 maxi()	. 92
4.21.2.7 mini()	. 92
4.21.2.8 moyenneTableau()	. 92
4.21.2.9 ordonnerTableau()	. 93
4.21.2.10 rechercheDichotomie()	. 93
4.21.2.11 sommeTableau()	. 94
4.21.2.12 zeroSiSuperieur()	. 94
4.22 Référence du fichier tableau.h	. 95
4.22.1 Description détaillée	. 95
4.22.2 Documentation des fonctions	. 95
4.22.2.1 acquiertTableau()	. 95
4.22.2.2 afficheTableau()	. 96
4.22.2.3 aleatoireTableau()	. 96
4.22.2.4 constanteTableau()	. 97
4.22.2.5 copie()	. 97
4.22.2.6 maxi()	. 98
4.22.2.7 mini()	. 98
4.22.2.8 moyenneTableau()	. 98
4.22.2.9 ordonnerTableau()	. 99
4.22.2.10 rechercheDichotomie()	. 99
4.22.2.11 sommeTableau()	. 100
4.22.2.12 zeroSiSuperieur()	. 100

101

Chapitre 1

Index des structures de données

1.1 Structures de données

Liste des structures de données avec une brève description :

Carte		
	Represente une carte d'un jeu de 32 cartes	
Contrat		
	Represente un contrat A la belote coichA©e	

Chapitre 2

Index des fichiers

2.1 Liste des fichiers

Liste de tous les fichiers avec une brève description :

acquisition.c	
Ficher source contenant les fonctions relative a l'acquisirion	ç
acquisition.h	
Ficher header contenant les prototype des fonctions relative a l'acquisirion	9
affichage.c	
Ficher source contenant les fonctions relative à l'affichage	12
affichage.h	
Ficher header contenant les prototypes des fonctions relative à l'affichage	17
autre.c	
Ficher source contenant les fonctions qui n'allait dans aucun autre fichier	22
autre.h	
Ficher header contenant les prototype des fonctions qui n'allait dans aucun autre fichier	25
formatageChaine.c	
Ficher source contenant les fonctions relative au formatage	29
formatageChaine.h	0.
Ficher header contenant les prototypes des fonctions relative au formatage	35
general.c	40
Ficher source contenant les fonctions les plus impotantes pour le jeu de la belote	40
general.h Ficher header contenant les prototypes des fonctions les plus impotantes pour le jeu de la belote	47
gestionCarte.c	47
Ficher contenant les fonctions relative a la gestion des cartes	53
gestionCarte.h	50
Ficher header contenant les protypes des fonctions relative a la gestion des cartes	61
gestionFichier.c	
Ficher source contenant les fonctions relative à la gestion des fichiers	69
gestionFichier.h	
Ficher header contenant les prototypes des fonctions relative à la gestion des fichiers	71
ia.c	
Ficher source contenant les fonctions relative au inteligences artificielles	73
ia.h	
Ficher header contenant les prototypes des fonctions relative au inteligences artificielles	74
main.c	
Ficher contenant la fonction main	70

Index des fichiers

main.h		
	er la déclaration des contrante, les énumérations, les structe et l'inclusion des haeder	
	ojet	78
sous-me		35
sous-me		37
tableau.d		
	er source contenant les fonctions relative à la gestion des tableau	39
tableau.h		
	er header contenant les prototypes des fonctions relative à la gestion des tableau	3 5

Chapitre 3

Documentation des structures de données

3.1 Référence de la structure Carte

represente une carte d'un jeu de 32 cartes

#include <main.h>

Graphe de collaboration de Carte:

Champs de données

Couleur couleurValeur valeur

3.1.1 Description détaillée

represente une carte d'un jeu de 32 cartes

Définition à la ligne 133 du fichier main.h.

3.1.2 Documentation des champs

3.1.2.1 couleur

Couleur couleur

Définition à la ligne 135 du fichier main.h.

3.1.2.2 valeur

Valeur valeur

type Couleur, donne la famille A laquel la carte apprartient

Définition à la ligne 136 du fichier main.h.

La documentation de cette structure a été générée à partir du fichier suivant :

- main.h

3.2 Référence de la structure Contrat

represente un contrat à la belote coichée

```
#include <main.h>
```

Graphe de collaboration de Contrat:

Champs de données

- Joueur preneurNbPoint nbPointCouleur atoutCoinche coinche

3.2.1 Description détaillée

represente un contrat à la belote coichée

Permet d'avoir toutes les information concernant un contrat stocké dans une seuil et même variable

Définition à la ligne 148 du fichier main.h.

3.2.2 Documentation des champs

3.2.2.1 atout

Couleur atout

type NbPoint, nombre de point du contrat

Définition à la ligne 152 du fichier main.h.

3.2.2.2 coinche

Coinche coinche

type Couleur, couleur de l'atout qui à été prise pour le contrat

Définition à la ligne 153 du fichier main.h.

3.2.2.3 nbPoint

NbPoint nbPoint

type Joueur, joueur qui a pris le contrat

Définition à la ligne 151 du fichier main.h.

3.2.2.4 preneur

Joueur preneur

Définition à la ligne 150 du fichier main.h.

La documentation de cette structure a été générée à partir du fichier suivant :

— main.h

Chapitre 4

Documentation des fichiers

4.1 Référence du fichier acquisition.c

ficher source contenant les fonctions relative a l'acquisirion

#include "main.h"

Graphe des dépendances par inclusion de acquisition.c:

4.2 Référence du fichier acquisition.h

ficher header contenant les prototype des fonctions relative a l'acquisirion

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Fonctions

- int acquisitionEntierAvecMessage (int min, int max)
- renvois au progammme un entier saisi par l'utilisateur avec message d'erreur entre 2 entiers
- int acquisition EntierSansMessage (int min, int max)
- accisition d'un entier saisi par l'utilisateur sans message d'erreur entre 2 entiers
- char acquisitionOuiNonSansMessage ()
 - accisition O/N 'oui ou non' saisi par l'utilisateur sans message d'erreur
- int acquisitionEntierSansMessageAvecConsigne (int min, int max, char consigne[])
- accisition d'un entier saisi par l'utilisateur sans message d'erreur entre 2 entiers avec une consigne paramétrable
- void acquisitionPseudoAvecMessage (char *pointeurPseudo, char instruction[], int type)
- acquisition d'un pseudo saisi par l'utilisateur avec message d'erreur
- int acquisitionEntierSecurise ()

fonction qui fait l'acquisition securisée d'un nombre

4.2.1 Description détaillée

ficher header contenant les prototype des fonctions relative a l'acquisirion

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.2.2 Documentation des fonctions

4.2.2.1 acquisitionEntierAvecMessage()

```
int acquisitionEntierAvecMessage (
                int min,
                int max )
```

renvois au progammme un entier saisi par l'utilisateur avec message d'erreur entre 2 entiers

Paramètres

min	: borne inférieur de l'ensemble dans lequel la valeur saisi doit ce trouver
max	: borne superieur de l'ensemble dans lequel la valeur saisi doit ce trouver

Renvoie

la valeur saise par l'utilisateur et vérifier

Définition à la ligne 11 du fichier acquisition.c.

Voici le graphe d'appel pour cette fonction :

4.2.2.2 acquisitionEntierSansMessage()

```
int acquisitionEntierSansMessage (
                int min,
                int max )
```

accisition d'un entier saisi par l'utilisateur sans message d'erreur entre 2 entiers

Paramètres

min	: borne inférieur de l'ensemble dans lequel la valeur saisi doit ce trouver
max	: borne superieur de l'ensemble dans lequel la valeur saisi doit ce trouver

Renvoie

la valeure entiere saise par l'utilisateur et vérifier

Définition à la ligne 24 du fichier acquisition.c.

Voici le graphe d'appel pour cette fonction :

4.2.2.3 acquisitionEntierSansMessageAvecConsigne()

```
int acquisitionEntierSansMessageAvecConsigne (  \qquad \qquad \text{int } \textit{min,}
```

```
int max,
char consigne[])
```

accisition d'un entier saisi par l'utilisateur sans message d'erreur entre 2 entiers avec une consigne paramétrable

Paramètres

min	: borne inférieur de l'ensemble dans lequel la valeur saisi doit ce trouver	
max	: borne superieur de l'ensemble dans lequel la valeur saisi doit ce trouver	
consigne	: message affiché avant a l'utilisateur pour lui indiquer ce qu'il doit saisir	

Renvoie

la valeure entiere saise par l'utilisateur et vérifier

Définition à la ligne 42 du fichier acquisition.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.2.2.4 acquisitionEntierSecurise()

```
int acquisitionEntierSecurise ( )
```

fonction qui fait l'acquisition securisée d'un nombre

Renvoie

renvoie le nombre saisi par l'utilisateur et -1 si aucun nombre a été saisi

Définition à la ligne 94 du fichier acquisition.c.

Voici le graphe des appelants de cette fonction :

4.2.2.5 acquisitionOuiNonSansMessage()

```
char acquisitionOuiNonSansMessage ( )
```

accisition O/N 'oui ou non' saisi par l'utilisateur sans message d'erreur

Renvoie

le caratere saisi par l'utilisateur, 'O' ou 'N' qui a ete verifie

Définition à la ligne 33 du fichier acquisition.c.

4.2.2.6 acquisitionPseudoAvecMessage()

acquisition d'un pseudo saisi par l'utilisateur avec message d'erreur

PPseudo	pointeur renvoyant vers le premier carractère du pseudo à modifier	
instruction	phrase à affichier pour donner l'instruction à l'utilisateur	
type	: vaut 1 si la fonction est appelée dans la fonction paramètre	

Renvoie

void

Définition à la ligne 52 du fichier acquisition.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.3 Référence du fichier affichage.c

ficher source contenant les fonctions relative à l'affichage

```
#include "main.h"
```

Graphe des dépendances par inclusion de affichage.c:

Fonctions

- int afficheMenuPrincipal (int type)
 - affiche le logo du jeu puis le menu principal
- int afficheInterfacePli (Carte dernierPli[], Carte pli[], char *pseudo[], Carte cartesEnMain[], Contrat contratActuel, char message[], Joueur utilisteur, Joueur dernierVainqueur, int score[], int pointManche[], int type)
 - affiche l'interface de jeu durant un pli
- void modifieTailleFenetre (int nbLigneFenetre, int nbColloneFentre)
 - modifie la taille de la fenetre dans laquel le programme s'execute
- void afficheSousMenus (char phrase[], char intitule[])
- affiche les différents sous menus
- void afficheMain (Carte carte[])
- affiche les carte qu'un joueur a en main
- void afficheContrat (Contrat contrat, char *pseudo[], int version)
 - affiche un contrat
- void afficheMenuSelection (char intitule[], char phrase[], int sautDeLigne)
 - affiche dans un cadre plusieurs chaines de carractères avec un certain nombre de lignes sautée entre chacunes d'elles
- Contrat proposeContratUtilisateur (Contrat dernierContrat, Joueur parle, Carte *pCarteMain)
 gère le choix du contrat par l'utilisateur

4.3.1 Description détaillée

ficher source contenant les fonctions relative à l'affichage

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.3.2 Documentation des fonctions

4.3.2.1 afficheContrat()

affiche un contrat

Paramètres

Contrat	contrat : contrat a afficher
char	*pseudo[]: tableau de pointeur contenant les pseudo des différents joueurs
int	version : 1 pour la version belle et de grande taille et 2 pour la version courte

Renvoie

void

Définition à la ligne 247 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.3.2.2 afficheInterfacePli()

affiche l'interface de jeu durant un pli

Paramètres

Carte	dernierPli[] : tableau contenant les 4 cartes du dernier plis
Carte	pli[] : tableau contenant les cartes du pli en cours, mettre 0, 0 si il n'y a pas de carte
char	*pseudo[] : tableau contenant les pseudo des 4 joueur
Carte	cartesEnMain[] : tableau contenant les carte dans la main de l'utilisateur
Contrat	contratActuel : contrat qui est en cour dans cette manche

char	message[500]
Joueur	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur
Joueur	dernierVainqueur : joueur aylant gagné le dernier pli
int	score[] : tableau qui contient les score de la partie
int	pointManche[] : tableau qui contient les points de la manche
int	type : version de la fonction : 0 pour un affichage et une acquisition et 1 pour l'acquisition seulle

Renvoie

int : valeur choisie par l'utilisateur

- < Formatage des chaine de caracère relative aux pseudo
- < Formatage des chaine de caractère relative au message
- < Formatage des chaine de caractère relative aux cartes
- < Formatage des chaine de caractère relative au contrat

Définition à la ligne 59 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.3.2.3 afficheMain()

affiche les carte qu'un joueur a en main

Paramètres

Carte	carte[] : tableau contanant les carte
-------	---------------------------------------

Renvoie

void

Définition à la ligne 227 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.3.2.4 afficheMenuPrincipal()

affiche le logo du jeu puis le menu principal

type quel type	e de menu est souhaité
----------------	------------------------

Renvoie

la valeur de l'action que l'utilisateur decide de faire

Définition à la ligne 11 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.3.2.5 afficheMenuSelection()

affiche dans un cadre plusieurs chaines de carractères avec un certain nombre de lignes sautée entre chacunes d'elles

Paramètres

char	intitule []: titre du du cadre
char	phrase[] : les chaines de carractère séparés par des points virgules
int	sautDeLigne : nombre de lignes à sauter entre chaque chaine de carractères

Renvoie

void

- < contrôle le centrage du titre du sous-menu
- < effacement de la chaine de carrateres
- <efface l'écran et affiche le sous-menu avec le texte correspondant

Définition à la ligne 269 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.3.2.6 afficheSousMenus()

affiche les différents sous menus

Renvoie

void

- < contrôle le centrage du titre du sous-menu
- < découpe la phrase pour pouvoir l'afficher
- <efface l'écran et affiche le sous-menu avec le texte correspondant

Définition à la ligne 186 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.3.2.7 modifieTailleFenetre()

modifie la taille de la fenetre dans laquel le programme s'execute

Paramètres

nbLingeFenetre	: nombre de ligne de la fenetre
nbColloneFenetre	: nombre de collone de la fenetre

Renvoie

void

Définition à la ligne 177 du fichier affichage.c.

Voici le graphe des appelants de cette fonction :

4.3.2.8 proposeContratUtilisateur()

gère le choix du contrat par l'utilisateur

Contrat	dernierContrat : dernier contrat proposé	
Joueur	parle : joueur qui parle	
Carte	*pCarteMain : pointeur sur le tableau qui stocke les carte dans la main du joueur	

Renvoie

Contrat : nouveau contrat proposé par le joueur

Paramètres

Définition à la ligne 348 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.4 Référence du fichier affichage.h

ficher header contenant les prototypes des fonctions relative à l'affichage

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Fonctions

- int afficheMenuPrincipal (int type)
 - affiche le logo du jeu puis le menu principal
- int afficheInterfacePli (Carte dernierPli[], Carte pli[], char *pseudo[], Carte cartesEnMain[], Contrat contratActuel, char message[], Joueur utilisteur, Joueur dernierVainqueur, int score[], int pointManche[], int type)
 - affiche l'interface de jeu durant un pli
- void modifieTailleFenetre (int nbLigneFenetre, int nbColloneFentre)
 - modifie la taille de la fenetre dans laquel le programme s'execute
- void afficheSousMenus (char phrase[], char intitule[])
- affiche les différents sous menus
- void afficheMain (Carte carte[])
 - affiche les carte qu'un joueur a en main
- void afficheContrat (Contrat contrat, char *pseudo[], int version)
 - affiche un contrat
- void afficheMenuSelection (char intitule[], char phrase[], int sautDeLigne)
 - affiche dans un cadre plusieurs chaines de carractères avec un certain nombre de lignes sautée entre chacunes d'elles
- Contrat proposeContratUtilisateur (Contrat dernierContrat, Joueur parle, Carte *pCarteMain)
 gère le choix du contrat par l'utilisateur

4.4.1 Description détaillée

ficher header contenant les prototypes des fonctions relative à l'affichage

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.4.2 Documentation des fonctions

4.4.2.1 afficheContrat()

affiche un contrat

Paramètres

Contrat	contrat : contrat a afficher
char	*pseudo[]: tableau de pointeur contenant les pseudo des différents joueurs
int	version: 1 pour la version belle et de grande taille et 2 pour la version courte

Renvoie

void

Définition à la ligne 247 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.4.2.2 afficheInterfacePli()

affiche l'interface de jeu durant un pli

Carte	dernierPli[] : tableau contenant les 4 cartes du dernier plis	
Carte	pli[] : tableau contenant les cartes du pli en cours, mettre 0, 0 si il n'y a pas de carte	
char	*pseudo[] : tableau contenant les pseudo des 4 joueur	
Carte	cartesEnMain[] : tableau contenant les carte dans la main de l'utilisateur	
Contrat	contratActuel : contrat qui est en cour dans cette manche	
char	nar message[500]	
Joueur	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec	
	uniquement des ordinateur	
Joueur	dernierVainqueur : joueur aylant gagné le dernier pli	
int	score[] : tableau qui contient les score de la partie	
int	pointManche[] : tableau qui contient les points de la manche	
int	type : version de la fonction : 0 pour un affichage et une acquisition et 1 pour l'acquisition seulle	

Renvoie

int : valeur choisie par l'utilisateur

- < Formatage des chaine de caracère relative aux pseudo
- < Formatage des chaine de caractère relative au message
- < Formatage des chaine de caractère relative aux cartes
- < Formatage des chaine de caractère relative au contrat

Définition à la ligne 59 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.4.2.3 afficheMain()

affiche les carte qu'un joueur a en main

Paramètres

Carte	carte[] : tableau contanant les carte

Renvoie

void

Définition à la ligne 227 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.4.2.4 afficheMenuPrincipal()

```
int afficheMenuPrincipal ( int \ type \ )
```

affiche le logo du jeu puis le menu principal

Paramètres

type	quel type de menu est souhaité
------	--------------------------------

Renvoie

la valeur de l'action que l'utilisateur decide de faire

Définition à la ligne 11 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.4.2.5 afficheMenuSelection()

affiche dans un cadre plusieurs chaines de carractères avec un certain nombre de lignes sautée entre chacunes d'elles

Paramètres

char	intitule []: titre du du cadre
char	phrase[] : les chaines de carractère séparés par des points virgules
int	sautDeLigne : nombre de lignes à sauter entre chaque chaine de carractères

Renvoie

void

- < contrôle le centrage du titre du sous-menu
- < effacement de la chaine de carrateres
- <efface l'écran et affiche le sous-menu avec le texte correspondant

Définition à la ligne 269 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.4.2.6 afficheSousMenus()

affiche les différents sous menus

phrase	phrase correspondant au sous menu choisi
1-11-01-0	principle control principle distribution control contr

Renvoie

void

- < contrôle le centrage du titre du sous-menu
- < découpe la phrase pour pouvoir l'afficher
- <efface l'écran et affiche le sous-menu avec le texte correspondant

Définition à la ligne 186 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.4.2.7 modifieTailleFenetre()

modifie la taille de la fenetre dans laquel le programme s'execute

Paramètres

nbLingeFenetre	: nombre de ligne de la fenetre
nbColloneFenetre	: nombre de collone de la fenetre

Renvoie

void

Définition à la ligne 177 du fichier affichage.c.

Voici le graphe des appelants de cette fonction :

4.4.2.8 proposeContratUtilisateur()

gère le choix du contrat par l'utilisateur

Contrat	dernierContrat : dernier contrat proposé
Joueur	parle : joueur qui parle
Carte	*pCarteMain : pointeur sur le tableau qui stocke les carte dans la main du joueur

Renvoie

Contrat : nouveau contrat proposé par le joueur

Paramètres

Définition à la ligne 348 du fichier affichage.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.5 Référence du fichier autre.c

ficher source contenant les fonctions qui n'allait dans aucun autre fichier

#include "main.h"

Graphe des dépendances par inclusion de autre.c:

Fonctions

- Joueur joueurSuivant (Joueur joueur)
 - passe au joueur suivant
- int nbAleatoire (int mini, int maxi)
 - renvoit in nombre aléatoire entre deux bornes
- void setContrat (Contrat *contrat, Joueur preneur, int nbPoint, Couleur atout, Coinche coinche)
 définit un contrat
- int pointPli (Carte pli[], Couleur atout, int nbCarte)
 - compte le nombre de points d'un pli
- float ajusteEchelle (float valeur, float entreMin, float entreMax, float sortieMin, float sortieMax)
 - met a l'echelle la valeur d'une variable comprise entre deux bornes pour que la variable de retour soit compris entre deux autres bornes
- void joue1000Partie (int nbPartie)

joue n partie aves uniquement des ia pour controler si tout ce passe bien et reuceillir des statistique

4.5.1 Description détaillée

ficher source contenant les fonctions qui n'allait dans aucun autre fichier

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.5.2 Documentation des fonctions

4.5.2.1 ajusteEchelle()

met a l'echelle la valeur d'une variable comprise entre deux bornes pour que la variable de retour soit compris entre deux autres bornes

Paramètres

float	valeur : valeur a mettre à l'echelle
float	entreMin : valeur minimum que peut prendre l'entrée
float	entreMax : valeur maximum que peut prendre l'entrée
float	sortieMin : valeur minimum que peut prendre la sortie
float	sortieMax : valeur maximum que peut prendre la sortie

Renvoie

float : la valeur de valeur une fois mise a l'echelle

Définition à la ligne 147 du fichier autre.c.

Voici le graphe des appelants de cette fonction :

4.5.2.2 joue1000Partie()

joue n partie aves uniquement des ia pour controler si tout ce passe bien et reuceillir des statistique

Paramètres

int	nbPartie nombre de partie que l'on veut jouer
-----	---

Renvoie

voir

Définition à la ligne 153 du fichier autre.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.5.2.3 joueurSuivant()

```
Joueur joueurSuivant (

Joueur joueur )
```

passe au joueur suivant

Paramètres

```
Joueur joueur : joueur actuel
```

Renvoie

Joueur : le joueur suivant

Définition à la ligne 11 du fichier autre.c.

Voici le graphe des appelants de cette fonction :

4.5.2.4 nbAleatoire()

```
int nbAleatoire (
          int mini,
          int maxi )
```

renvoit in nombre aléatoire entre deux bornes

Paramètres

int	mini : borne inférieur
int	maxi : borne surpérieur

Renvoie

int : le nombre aléatoire

Définition à la ligne 21 du fichier autre.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.5.2.5 pointPli()

compte le nombre de points d'un pli

Carte	pli []: tableau des cartes possées dont on doit faire la somme des points
Couleur	atout: couleur de l'atout joué dans la manche
nbCarte	nombre de carte à compter

Renvoie

points

Définition à la ligne 34 du fichier autre.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.5.2.6 setContrat()

définit un contrat

Paramètres

Contat	*contrat : pointeur sur le contrat a modifier
Joueur	preneur : preneur du contrat
int	nbPoint : nombre de point
Couleur	atout : couleur de l'aout
Coinche	coinche : normal, coinché , surcoinché

Renvoie

void

Définition à la ligne 26 du fichier autre.c.

Voici le graphe des appelants de cette fonction :

4.6 Référence du fichier autre.h

ficher header contenant les prototype des fonctions qui n'allait dans aucun autre fichier

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Fonctions

```
    Joueur joueur Suivant (Joueur joueur)
        passe au joueur suivant
        int nbAleatoire (int mini, int maxi)
        renvoit in nombre aléatoire entre deux bornes
        void setContrat (Contrat *contrat, Joueur preneur, int nbPoint, Couleur atout, Coinche coinche)
        définit un contrat
        int pointPli (Carte pli[], Couleur atout, int nbCarte)
        compte le nombre de points d'un pli
        float ajusteEchelle (float valeur, float entreMin, float entreMax, float sortieMin, float sortieMax)
        met a l'echelle la valeur d'une variable comprise entre deux bornes pour que la variable de retour soit compris entre
        deux autres bornes
        void joue1000Partie (int nbPartie)
        joue n partie aves uniquement des ia pour controler si tout ce passe bien et reuceillir des statistique
```

4.6.1 Description détaillée

ficher header contenant les prototype des fonctions qui n'allait dans aucun autre fichier

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.6.2 Documentation des fonctions

4.6.2.1 ajusteEchelle()

met a l'echelle la valeur d'une variable comprise entre deux bornes pour que la variable de retour soit compris entre deux autres bornes

Paramètres

float	valeur : valeur a mettre à l'echelle
float	entreMin : valeur minimum que peut prendre l'entrée
float	entreMax : valeur maximum que peut prendre l'entrée
float	sortieMin : valeur minimum que peut prendre la sortie
float	sortieMax : valeur maximum que peut prendre la sortie

Renvoie

float : la valeur de valeur une fois mise a l'echelle

Définition à la ligne 147 du fichier autre.c.

Voici le graphe des appelants de cette fonction :

4.6.2.2 joue1000Partie()

joue n partie aves uniquement des ia pour controler si tout ce passe bien et reuceillir des statistique

Paramètres

int | nbPartie nombre de partie que l'on veut jouer

Renvoie

voir

Définition à la ligne 153 du fichier autre.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.6.2.3 joueurSuivant()

```
Joueur joueurSuivant (

Joueur joueur )
```

passe au joueur suivant

Paramètres

Joueur	joueur : joueur actuel

Renvoie

Joueur : le joueur suivant

Définition à la ligne 11 du fichier autre.c.

Voici le graphe des appelants de cette fonction :

4.6.2.4 nbAleatoire()

```
int nbAleatoire (
          int mini,
          int maxi )
```

renvoit in nombre aléatoire entre deux bornes

int	mini : borne inférieur
int	maxi : borne surpérieur

Renvoie

int : le nombre aléatoire

Définition à la ligne 21 du fichier autre.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.6.2.5 pointPli()

compte le nombre de points d'un pli

Paramètres

Carte	pli []: tableau des cartes possées dont on doit faire la somme des points
Couleur	atout: couleur de l'atout joué dans la manche
nbCarte	nombre de carte à compter

Renvoie

points

Définition à la ligne 34 du fichier autre.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.6.2.6 setContrat()

définit un contrat

Paramètres

Contat	*contrat : pointeur sur le contrat a modifier
Joueur	preneur : preneur du contrat
int	nbPoint : nombre de point
Couleur	atout : couleur de l'aout
Coinche	coinche : normal, coinché , surcoinché

void

Définition à la ligne 26 du fichier autre.c.

Voici le graphe des appelants de cette fonction :

Référence du fichier formatageChaine.c

ficher source contenant les fonctions relative au formatage

```
#include "main.h"
```

Graphe des dépendances par inclusion de formatageChaine.c:

Fonctions

- int centreChaine (char chaineInitial[], char chaineFinal[], int longeurChaine)
 - transphorme une chaine de caractère de longeur inconue en une chaine de caratère fixé de tel sorte que le texte soit centré
- int centreModifieChaine (char chaine[], int longeurChaine)
 - transforme une chaine de cararactère en sa vertion centrée
- int decoupeChaine (char chaineInitiale[], char ∗chaineFinale, int tailleLigne, int nbLigne)
- fontion qui permet de découper une chaine de caracère en plusieur chaine de taille fixée en coupant sur un espace
- void rempliEspace (char *chaine, int nbEspace)
 - rempli une chaine de caractère de nbEspace caractère espace '
- void formateCarte (Carte carte[], char ∗chaineFinale, int nbCarte, int tailleChaine, int version)
 - formate les chaine de caracère qui affiche la valeur et la couleur des cartes
- void stockeInfoCarte (Carte carte, char *valeur, char *couleur, int version, int tailleChaine)
 - met dans une chaine de caractère la valeur et la couleur d'une carte
- void formatePseudo (Joueur joueur, char *pseudo[], int tailleChaine, char chaineFinal[], int version)
 - stocke dans une chaine de caractère le pseudo d'un des joueur
- void formateContrat (Contrat contrat, char *chaineFinal, int tailleLigne, char *pseudo[])
 - stocke dans des chaine de caractère les info du contrat
- int aligneModifieChaine (char chaine[], int longeurChaine)
 - transforme une chaine de cararactère en sa vertion aligné a gauche
- void genereMessage (char message[], Joueur parle, char *pseudo[], Carte carteJoue, int score, TypeMessage typeMessage)

génere un message et l'enregistre dans une chaine de caractère

Description détaillée

ficher source contenant les fonctions relative au formatage

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.7.2 Documentation des fonctions

4.7.2.1 aligneModifieChaine()

transforme une chaine de cararactère en sa vertion aligné a gauche

Paramètres

char	chaine : chaine a modifier
int	longeurChaine

Renvoie

int: 0 si tout c'est bien passé

Définition à la ligne 299 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.7.2.2 centreChaine()

transphorme une chaine de caractère de longeur inconue en une chaine de caratère fixé de tel sorte que le texte soit centré

Paramètres

char	chaineInitial : chaine a modifier
char	chaineFinal : pointeur vers la variable qui stocke la chainne final
int	longeurChaine : longeur de la chaine final

Renvoie

int : 0 si tout va bien, 1 si la chaine est Initiale test plus longue que la longeur désirée

Définition à la ligne 10 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.7.2.3 centreModifieChaine()

transforme une chaine de cararactère en sa vertion centrée

Paramètres

char	chaine : chaine a modifier
int	longeurChaine

Renvoie

int: 0 si tout c'est bien passé

Définition à la ligne 35 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.7.2.4 decoupeChaine()

fontion qui permet de découper une chaine de caracère en plusieur chaine de taille fixée en coupant sur un espace

Paramètres

chaineInitiale[]	: chaine a decouper
chaineFinale[][]	: tableau contenant les chaine un fois découpé
onamor marol jij	. tablead contenant lee chame an lee deceape
taillal iana	unabra da aaraaàta mayimun nar liana
tailleLigne	: nobre de caracète maximun par ligne
nbLinge	: nombre maximun de ligne
	The state of the s

Renvoie

0 si tout va bien 1 si la chaine est trop longue

Définition à la ligne 45 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.7.2.5 formateCarte()

```
char * chaineFinale,
int nbCarte,
int tailleChaine,
int version )
```

formate les chaine de caracère qui affiche la valeur et la couleur des cartes

Paramètres

chaineFinale[][][]	tableau a trois dimention qui stocke les chaine de caracère de la valeur et la couleurs des cartes
carte[]	: tableau de typpe Carte contenant les carte a afficher
nbCarte	: nombre de carte dans le tableau carte[]
tailleChaine	: taille des chaineFinale
version	: 0 si c'est la verion courte et 1 pour la verision longue

Renvoie

void

Définition à la ligne 101 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.7.2.6 formateContrat()

stocke dans des chaine de caractère les info du contrat

Paramètres

Contrat	contrat : contrat a afficher
char	*chainefinal : chaine final ou on enregistre les info du contrat
int	tailleLigne : taille d'une ligne
char	*pseudo[] : tableau qui contient les pseudo des joueur

Renvoie

void

Définition à la ligne 251 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.7.2.7 formatePseudo()

```
void formatePseudo (
    Joueur joueur,
    char * pseudo[],
    int tailleChaine,
    char chaineFinal[],
    int version )
```

stocke dans une chaine de caractère le pseudo d'un des joueur

Paramètres

Joueur	joueur : joueur dont on veut afficher le pseudo
char	*pseudo[] : tableau contennant les pseudo des joueur
int	tailleChaine : taille de la chaine final
char	chaineFinal[] : chaine final ou est stocké le pseudo
int	varsion : 1 pour la version ou la chaine finale est centrée 0 pour l'aligné a gauche

Renvoie

void

Définition à la ligne 220 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.7.2.8 genereMessage()

génere un message et l'enregistre dans une chaine de caractère

Paramètres

char	message[] : tableau de char ou on enregistre le message
Joueur	parle : joueur qui pose une carte
char	*pseudo[] : tableau qui contient les pseudo des joueur
Carte	carteJoue : carte qui vine d'être jouée
int	score : score a afficher
TypeMessage	typeMessage : permet de savoir quelle message on veux afficher

Renvoie

void

Définition à la ligne 314 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.7.2.9 rempliEspace()

rempli une chaine de caractère de nbEspace caractère espace ' '

Paramètres

chaine[]	: chaine a remplir;
nbEspace[]	: nombre d'espaca a placer dans la chaine

Renvoie

void

Définition à la ligne 93 du fichier formatageChaine.c.

Voici le graphe des appelants de cette fonction :

4.7.2.10 stockelnfoCarte()

met dans une chaine de caractère la valeur et la couleur d'une carte

Paramètres

carte	: variable de type carte qui contient la carte a afficher
valeur	: pointeur vers la chaine qui stocke la valeur (ou la valeur et la couleur si mode court)
couleur	: pointeur vers la chaine qui stocke la couleur
vesion	: 0 si c'est la version courte et 1 pour la version longue
tailleChaine	: taille de la chaine de caratère dans laquel on écrit

Renvoie

void

Définition à la ligne 108 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.8 Référence du fichier formatageChaine.h

ficher header contenant les prototypes des fonctions relative au formatage

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Fonctions

— int centreChaine (char chaineInitial[], char chaineFinal[], int longeurChaine) transphorme une chaine de caractère de longeur inconue en une chaine de caratère fixé de tel sorte que le texte soit centré — int centreModifieChaine (char chaine[], int longeurChaine) transforme une chaine de cararactère en sa vertion centrée — int decoupeChaine (char chaineInitiale[], char *chaineFinale, int tailleLigne, int nbLigne) fontion qui permet de découper une chaine de caracère en plusieur chaine de taille fixée en coupant sur un espace void rempliEspace (char *chaine, int nbEspace) rempli une chaine de caractère de nbEspace caractère espace ' ' void formateCarte (Carte carte[], char *chaineFinale, int nbCarte, int tailleChaine, int version) formate les chaine de caracère qui affiche la valeur et la couleur des cartes void stockeInfoCarte (Carte carte, char *valeur, char *couleur, int version, int tailleChaine) met dans une chaine de caractère la valeur et la couleur d'une carte void formatePseudo (Joueur joueur, char *pseudo[], int tailleChaine, char chaineFinal[], int version) stocke dans une chaine de caractère le pseudo d'un des joueur void formateContrat (Contrat contrat, char *chaineFinal, int tailleLigne, char *pseudo[]) stocke dans des chaine de caractère les info du contrat int aligneModifieChaine (char chaine[], int longeurChaine) transforme une chaine de cararactère en sa vertion aligné a gauche
— void genereMessage (char message[], Joueur parle, char *pseudo[], Carte carteJoue, int score, TypeMessage typeMessage) génere un message et l'enregistre dans une chaine de caractère

4.8.1 Description détaillée

ficher header contenant les prototypes des fonctions relative au formatage

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.8.2 Documentation des fonctions

4.8.2.1 aligneModifieChaine()

transforme une chaine de cararactère en sa vertion aligné a gauche

Paramètres

char	chaine : chaine a modifier
int	longeurChaine

Renvoie

int: 0 si tout c'est bien passé

Définition à la ligne 299 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.8.2.2 centreChaine()

transphorme une chaine de caractère de longeur inconue en une chaine de caratère fixé de tel sorte que le texte soit centré

Paramètres

char	chaineInitial : chaine a modifier
char	chaineFinal : pointeur vers la variable qui stocke la chainne final
int	longeurChaine : longeur de la chaine final

Renvoie

int : 0 si tout va bien, 1 si la chaine est Initiale test plus longue que la longeur désirée

Définition à la ligne 10 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.8.2.3 centreModifieChaine()

transforme une chaine de cararactère en sa vertion centrée

char chaine:		chaine : chaine a modifier
int		IongeurChaine

int: 0 si tout c'est bien passé

Définition à la ligne 35 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.8.2.4 decoupeChaine()

fontion qui permet de découper une chaine de caracère en plusieur chaine de taille fixée en coupant sur un espace

Paramètres

chaineInitiale[]	: chaine a decouper
chaineFinale[][]	: tableau contenant les chaine un fois découpé
tailleLigne	: nobre de caracète maximun par ligne
nbLinge	: nombre maximun de ligne

Renvoie

0 si tout va bien 1 si la chaine est trop longue

Définition à la ligne 45 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.8.2.5 formateCarte()

formate les chaine de caracère qui affiche la valeur et la couleur des cartes

chaineFinale[][][]	tableau a trois dimention qui stocke les chaine de caracère de la valeur et la couleurs des cartes
carte[]	: tableau de typpe Carte contenant les carte a afficher
nbCarte	: nombre de carte dans le tableau carte[]
tailleChaine : taille des chaineFinale	
version	: 0 si c'est la verion courte et 1 pour la verision longue

void

Définition à la ligne 101 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.8.2.6 formateContrat()

stocke dans des chaine de caractère les info du contrat

Paramètres

Contrat	contrat : contrat a afficher	
char	*chainefinal : chaine final ou on enregistre les info du contrat	
int	tailleLigne : taille d'une ligne	
char *pseudo[] : tableau qui contient les pseudo des joueur		

Renvoie

void

Définition à la ligne 251 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.8.2.7 formatePseudo()

```
void formatePseudo (
          Joueur joueur,
          char * pseudo[],
          int tailleChaine,
          char chaineFinal[],
          int version )
```

stocke dans une chaine de caractère le pseudo d'un des joueur

Joueur	joueur : joueur dont on veut afficher le pseudo	
char *pseudo[] : tableau contennant les pseudo des joueur		
int	tailleChaine : taille de la chaine final	
char	chaineFinal[] : chaine final ou est stocké le pseudo	
int	varsion : 1 pour la version ou la chaine finale est centrée 0 pour l'aligné a gauche	

void

Définition à la ligne 220 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.8.2.8 genereMessage()

génere un message et l'enregistre dans une chaine de caractère

Paramètres

char message[] : tableau de char ou on enregistre le message		
Joueur	parle : joueur qui pose une carte	
char	*pseudo[] : tableau qui contient les pseudo des joueur	
Carte carteJoue : carte qui vine d'être jouée		
int score : score a afficher		
TypeMessage	typeMessage : permet de savoir quelle message on veux afficher	

Renvoie

void

Définition à la ligne 314 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.8.2.9 rempliEspace()

rempli une chaine de caractère de nbEspace caractère espace ' '

chaine[]	: chaine a remplir;
nbEspace[]	: nombre d'espaca a placer dans la chaine

void

Définition à la ligne 93 du fichier formatageChaine.c.

Voici le graphe des appelants de cette fonction :

4.8.2.10 stockelnfoCarte()

met dans une chaine de caractère la valeur et la couleur d'une carte

Paramètres

carte	: variable de type carte qui contient la carte a afficher	
valeur	: pointeur vers la chaine qui stocke la valeur (ou la valeur et la couleur si mode court)	
couleur	: pointeur vers la chaine qui stocke la couleur	
vesion	: 0 si c'est la version courte et 1 pour la version longue	
tailleChaine : taille de la chaine de caratère dans laquel on écrit		

Renvoie

void

Définition à la ligne 108 du fichier formatageChaine.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.9 Référence du fichier general.c

ficher source contenant les fonctions les plus impotantes pour le jeu de la belote

```
#include "main.h"
```

Graphe des dépendances par inclusion de general.c:

Fonctions

- void initialisation (int nbLigneFenetre, int nbColloneFenetre)
 - fonction qui gà re l'initialitation et le lancement du programme
- void menuPrincipal ()
 - fonction qui gà "re le debut de la partie
- int nouvelle Partie (char *pseudo[], Joueur utilisateur, int *pStatistique, int infoEcritureFicher[])
 - lancement d'une nouvelle partie
- char manche (char *pseudo[], int score[], Joueur dealer, Joueur utilisateur, int *pStatistique)
 lancement d'une manche

- Contrat annonceContrat (char *pseudo[], Joueur dealer, Carte *pCarteMain, Joueur utilisateur)
 gà "re l'annonce et la surenchà "re des contats par les joueurs
- Contrat proposeContrat (Contrat dernierContrat, Joueur parle, char *pseudo[], Carte *pCarteMain, Joueur utilisateur)

proposition d'un contrat par un joueur

— Joueur pli (Contrat contrat, Joueur premierAJouer, char *pseudo[], Carte *pCarteMain, int pointManche[], int pointAnonce[], char belote[], Carte cartePli[], Carte carteAncienPli[], int score[], Joueur utilisateur, int numPli)

fonction qui gà "re un pli

- int poseCarte (Joueur joueur, int numCarte, Carte *pMainJoueurs, Carte pli[], int carteRestante)
 fonction qui verifie la validitÃ@ d'une carte est la pose
- char calculPointManche (Contrat contrat, int pointManche[], int pointAnonce[], int pointBelote[], int score[], char *pseudo[], Joueur utilisateur)

compte les point gagné par chaque joueur au cour de la manche e fonction du contrat et ajoute ces point au tableau qui contient tout les score de la partie

4.9.1 Description détaillée

ficher source contenant les fonctions les plus impotantes pour le jeu de la belote

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.9.2 Documentation des fonctions

4.9.2.1 annonceContrat()

gà re l'annonce et la surenchà re des contats par les joueurs

char	*pseudo[]: tableau de pointeurs contenant les pseudos des différents joueurs	
Joueur	dealer : joueur qui distibue les cartes	
Carte	*pCarteMain : pointeur sur le tableau qui stocke les cartes dans la main de chaque joueur	
Joueuer utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur		

Contrat: contrat final qui a ete choisi pour la partie

< Affichage du choix du joueur

Définition à la ligne 287 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.9.2.2 calculPointManche()

compte les point gagné par chaque joueur au cour de la manche e fonction du contrat et ajoute ces point au tableau qui contient tout les score de la partie

Paramètres

Contrat	contrat : contrat qui a été choisi durant la manche
int	pointManche[] : tableau qui contient les points de la manche
int	pointAnonce[] : tableau qui contient les points d'anonce de la manche
int	pointBelote[] : tableau contenant les point de belote rebelote de chaque joueur
int	score[] : tableau qui contient les score de la partie
char	*pseudo[]: tableau de pointeur contenant les pseudo des diffÃ@rents joueur
Joueur	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur

Renvoie

char : 1 si le contrtat a été réussi et 0 si le contrat a échoué

- < On reche si le contrat est rampli
- < cas ou un des joueur a pris un capot
- < cas ou un des joueur a pris une gÃ@nÃ@rale
- < Le contrat est rempli si le les point de l'équipe est suppérieur a ce qui anoncé et si il on 82 point sans les anonces
- < on affiche ce qui c'est passé

Définition à la ligne 476 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.9.2.3 initialisation()

fonction qui gà re l'initialitation et le lancement du programme

Paramètres

nbLigneFenetre	: taille verticale de la fenêtre
nbColloneFenetre	: taille horizonotale de la fenêtre

Renvoie

void

Définition à la ligne 11 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.9.2.4 manche()

lancement d'une manche

Paramètres

char	*pseudo[] : tableau de pointeurs contenant les pseudos des diffÃ@rents joueurs	
int	score[] : tableau contenant les scores des joueur	
Joueur	dealer : joueur qui distibue les cartes	
Joueuer	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur	
int	*pStatistique : pointeur sur un tableau 4*4 qui contient des statistique sur chaque joueur	

Renvoie

char: 0 si tout le monde passe, 1 si la manche a bien lieu

- < tableau qui stocke les anonce de chaque joueur, on y ajoute 1 pour le roi d'atout puis 1 pour la dame d'atout
- < distribution des cartes
- < On cherche si les joueurs on des anonce a faire
- < On passe a la phase suivante uniquement si un contrat a ete pris sinon on relance une manche

< si un joueur a fait une belote rebelote il gagne 20 point

Définition à la ligne 188 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.9.2.5 menuPrincipal()

```
void menuPrincipal ( )
```

fonction qui gà re le debut de la partie

Renvoie

void

- < contrà 'le d'acquisition avec rÃ@affichage de l'interfface
- <executer la fonction nouvelle partie

Définition à la ligne 16 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.9.2.6 nouvellePartie()

lancement d'une nouvelle partie

Paramètres

char	*pseudo[] : tableau de pointeurs contenant les pseudos des diffÃ@rents joueurs
Joueuer	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur
int	*pStatistique : pointeur sur un tableau 4*4 qui contient des statistique sur chaque joueur
int	infoEcritureFichier[] : tableau contenant les info qui devront etre écrite dans le fichier

Renvoie

int : nombre de manche jusqu'a la victoir d'une équipe

- < On change l'affichage pour avoir un affichage plus simple a lire lors d'une partie entre 4 ordinateur
- < a suprimer une fois de debug fini
- < Fin de partie

- < affichage des resultat
- < L'utilisatuer et nord gagnent
- < est et touest gagnent

Définition à la ligne 99 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.9.2.7 pli()

```
Joueur pli (

Contrat contrat,

Joueur premierAJouer,

char * pseudo[],

Carte * pCarteMain,

int pointManche[],

int pointAnonce[],

char belote[],

Carte cartePli[],

Carte carteAncienPli[],

int score[],

Joueur utilisateur,

int numPli)
```

fonction qui gà re un pli

Paramètres

Contrat	contrat : contrat qui a été choisipour cette manche
Joueur	premierAjouer : premier joueur a jouer dans le pli
char	*pseudo[]: tableau de pointeur contenant les pseudo des diffÃ@rents joueur
Carte	*pCarteMain : pointeur sur le tableau qui stocke les carte dans la main de chaque joueur
int	pointManche[] : tableau contenant les point de chaque joueur dans la manche
int	pointAnonce[] : tableau contenant les point d'anonce de chaque joueur
char	belote[] : tableau contenant les ancone belote rebelote de chaque joueur
Carte	cartePli[] : tableau contenant les 4 carte du pli
Carte	carteAncienPli[] : tableau contenant les carte du pli précédent
int	score[] : tableau qui contient les score de la partie
Joueur	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur

Renvoie

Joueur : le vainceur du pli

- < interface de pli Utilisateur
- < interface de pli ordinateur
- < on regarde si une dame ou un roi d'atout vien d'etre jouer pour ajouter au tableau belote[]

- < affichage de la carte qui vien d'etre jouée
- < 10 de der, on ajoute 10 point au vainceur si on est dans le dernier pli

Définition à la ligne 390 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.9.2.8 poseCarte()

fonction qui verifie la validité d'une carte est la pose

Paramètres

joueur	variable qui defini la position du joueur qui effectue l'action
numCarte	variable qui defini la carte choisie par le joueur
pMainJoueurs	: pointeur qui renvoie vers le tableau de la main de tous les joueurs
pPli	: pointeur qui renvoie vers le tableau des carte jouées dans le pli

Renvoie

Définition à la ligne 463 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.9.2.9 proposeContrat()

proposition d'un contrat par un joueur

Contrat	dernierContrat : dernier contrat proposé	
Joueur	parle : joueur qui parle	
char	*pseudo[]: tableau de pointeurs contenant les pseudos des différents joueurs	
Carte	*pCarteMain : pointeur sur le tableau qui stocke les cartes dans la main de chaque joueurs	
Joueur	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec	
	uniquement des ordinateur Généré par Doxygen	

Contrat : nouveau contrat proposé par le joueur

- < acquisition par l'utilisateur
- < choix par l'ia d'un contrat

Définition à la ligne 371 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.10 Référence du fichier general.h

ficher header contenant les prototypes des fonctions les plus impotantes pour le jeu de la belote

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Fonctions

- void initialisation (int nbLigneFenetre, int nbColloneFenetre)
 - fonction qui g $ilde{A}$ re l'initialitation et le lancement du programme
- void menuPrincipal ()
 - fonction qui gà re le debut de la partie
- int nouvelle Partie (char *pseudo[], Joueur utilisateur, int *pStatistique, int infoEcritureFicher[])
 - lancement d'une nouvelle partie
- char manche (char *pseudo[], int score[], Joueur dealer, Joueur utilisateur, int *pStatistique)
 - lancement d'une manche
- Contrat annonceContrat (char *pseudo[], Joueur dealer, Carte *pCarteMain, Joueur utilisateur)
- gà re l'annonce et la surenchà re des contats par les joueurs
- Contrat proposeContrat (Contrat dernierContrat, Joueur parle, char *pseudo[], Carte *pCarteMain, Joueur utilisateur)
 - proposition d'un contrat par un joueur
- Joueur pli (Contrat contrat, Joueur premierAJouer, char *pseudo[], Carte *pCarteMain, int pointManche[], int pointAnonce[], char belote[], Carte cartePli[], Carte carteAncienPli[], int score[], Joueur utilisateur, int numPli)
- fonction qui gà re un pli
- int poseCarte (Joueur joueur, int numCarte, Carte *pMainJoueurs, Carte pli[], int carteRestante)
 - fonction qui verifie la validité d'une carte est la pose
- char calculPointManche (Contrat contrat, int pointManche[], int pointAnonce[], int pointBelote[], int score[], char *pseudo[], Joueur utilisateur)

compte les point gagn \tilde{A} © par chaque joueur au cour de la manche e fonction du contrat et ajoute ces point au tableau qui contient tout les score de la partie

4.10.1 Description détaillée

ficher header contenant les prototypes des fonctions les plus impotantes pour le jeu de la belote

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.10.2 Documentation des fonctions

4.10.2.1 annonceContrat()

gà re l'annonce et la surenchà re des contats par les joueurs

Paramètres

char	*pseudo[]: tableau de pointeurs contenant les pseudos des différents joueurs
Joueur	dealer : joueur qui distibue les cartes
Carte	*pCarteMain : pointeur sur le tableau qui stocke les cartes dans la main de chaque joueur
Joueuer	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur

Renvoie

Contrat : contrat final qui a ete choisi pour la partie

< Affichage du choix du joueur

Définition à la ligne 287 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.10.2.2 calculPointManche()

compte les point gagné par chaque joueur au cour de la manche e fonction du contrat et ajoute ces point au tableau qui contient tout les score de la partie

Contrat	contrat : contrat qui a été choisi durant la manche	
int	pointManche[] : tableau qui contient les points de la manche	
int	pointAnonce[] : tableau qui contient les points d'anonce de la manche	
int	pointBelote[] : tableau contenant les point de belote rebelote de chaque joueur	
int	score[] : tableau qui contient les score de la partie Généré par Doxyge	
char	*pseudo[]: tableau de pointeur contenant les pseudo des différents joueur	
Joueur	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur	

char : 1 si le contrtat a été réussi et 0 si le contrat a échoué

- < On reche si le contrat est rampli
- < cas ou un des joueur a pris un capot
- < cas ou un des joueur a pris une gÃ@nÃ@rale
- < Le contrat est rempli si le les point de l' \tilde{A} ©quipe est supp \tilde{A} ©rieur a ce qui anonc \tilde{A} © et si il on 82 point sans les anonces
- < on affiche ce qui c'est passé

Définition à la ligne 476 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.10.2.3 initialisation()

```
void initialisation (
                int nbLigneFenetre,
                int nbColloneFenetre )
```

fonction qui gà re l'initialitation et le lancement du programme

Paramètres

nbLigneFenetre	: taille verticale de la fenêtre
nbColloneFenetre	: taille horizonotale de la fenêtre

Renvoie

void

Définition à la ligne 11 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.10.2.4 manche()

lancement d'une manche

Paramètres

char	*pseudo[] : tableau de pointeurs contenant les pseudos des diffÃ@rents joueurs
int	score[] : tableau contenant les scores des joueur
Joueur	dealer : joueur qui distibue les cartes
Joueuer	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur
int	*pStatistique : pointeur sur un tableau 4*4 qui contient des statistique sur chaque joueur

Renvoie

char: 0 si tout le monde passe, 1 si la manche a bien lieu

- < tableau qui stocke les anonce de chaque joueur, on y ajoute 1 pour le roi d'atout puis 1 pour la dame d'atout
- < distribution des cartes
- < On cherche si les joueurs on des anonce a faire
- < On passe a la phase suivante uniquement si un contrat a ete pris sinon on relance une manche
- < si un joueur a fait une belote rebelote il gagne 20 point

Définition à la ligne 188 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.10.2.5 menuPrincipal()

```
void menuPrincipal ( )
```

fonction qui gà re le debut de la partie

Renvoie

void

- < contr $\tilde{\mathsf{A}}$ 'le d'acquisition avec r $\tilde{\mathsf{A}}$ ©affichage de l'interfface
- <executer la fonction nouvelle partie

Définition à la ligne 16 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.10.2.6 nouvellePartie()

lancement d'une nouvelle partie

Paramètres

char	*pseudo[] : tableau de pointeurs contenant les pseudos des diffÃ@rents joueurs
Joueuer	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur
int	*pStatistique : pointeur sur un tableau 4*4 qui contient des statistique sur chaque joueur
int	infoEcritureFichier[] : tableau contenant les info qui devront etre écrite dans le fichier

Renvoie

int : nombre de manche jusqu'a la victoir d'une équipe

- < On change l'affichage pour avoir un affichage plus simple a lire lors d'une partie entre 4 ordinateur
- < a suprimer une fois de debug fini
- < Fin de partie
- < affichage des resultat
- < L'utilisatuer et nord gagnent
- < est et touest gagnent

Définition à la ligne 99 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.10.2.7 pli()

fonction qui gà re un pli

Contrat	contrat : contrat qui a été choisipour cette manche	
Joueur	premierAjouer : premier joueur a jouer dans le pli	
char	*pseudo[]: tableau de pointeur contenant les pseudo des différents joueur	
Carte	*pCarteMain : pointeur sur le tableau qui stocke les carte dans la main de chaque joueur	
int	pointManche[] : tableau contenant les point de chaque joueur dans la manche	
int	pointAnonce[] : tableau contenant les point d'anonce de chaque joueur	
Généré par Do	"ୱିଣାଠte[] : tableau contenant les ancone belote rebelote de chaque joueur	
Carte	cartePli[] : tableau contenant les 4 carte du pli	
Carte	carteAncienPli[] : tableau contenant les carte du pli précédent	
int	aggraff : tablegu qui contient les georg de la partie	

Joueur : le vainceur du pli

- < interface de pli Utilisateur
- < interface de pli ordinateur
- < on regarde si une dame ou un roi d'atout vien d'etre jouer pour ajouter au tableau belote[]
- < affichage de la carte qui vien d'etre jouée
- < 10 de der, on ajoute 10 point au vainceur si on est dans le dernier pli

Définition à la ligne 390 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.10.2.8 poseCarte()

fonction qui verifie la validité d'une carte est la pose

Paramètres

joueur	variable qui defini la position du joueur qui effectue l'action
numCarte	variable qui defini la carte choisie par le joueur
pMainJoueurs	: pointeur qui renvoie vers le tableau de la main de tous les joueurs
pPli	: pointeur qui renvoie vers le tableau des carte jouées dans le pli

Renvoie

Définition à la ligne 463 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.10.2.9 proposeContrat()

proposition d'un contrat par un joueur

Paramètres

Contrat	dernierContrat : dernier contrat proposé
Joueur	parle : joueur qui parle
char	*pseudo[]: tableau de pointeurs contenant les pseudos des différents joueurs
Carte	*pCarteMain : pointeur sur le tableau qui stocke les cartes dans la main de chaque joueurs
Joueur	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur

Renvoie

Contrat : nouveau contrat proposé par le joueur

- < acquisition par l'utilisateur
- < choix par l'ia d'un contrat

Définition à la ligne 371 du fichier general.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.11 Référence du fichier gestionCarte.c

ficher contenant les fonctions relative a la gestion des cartes

```
#include "main.h"
```

Graphe des dépendances par inclusion de gestionCarte.c:

Fonctions

- void distribueCarte (Carte *pCarteMain)
 - distribue les cartes entre les différent joueur
- void setCarte (Carte *carte, Valeur valeurCarte, Couleur couleurCarte)
 - donne une valeur et une couleur a une carte
- void supprimeCarte (Carte carte[], int nbCarte, int carteASupprimer)
 - supprime une carte d'un tableau et réduit la taille de ce tableu par le bas
- Joueur vainqueurPli (Carte pli[], Couleur atout, Joueur premierAJouer)
 - donne le vainquer d'un pli
- float forceCarte (Carte carteACalculer, Couleur atout, Couleur entame)
 - calcul la force d'une carte, en faisant la probabilitée qu'elle a de gagné contre toutes les autres cartes
- char carteValide (Carte cartePose, Carte pli[], Couleur atout, Carte *pCarteMainJoueur, Joueur premierA←
 Jouer, Joueur parle)
 - ddétermine si une carte peut etre posé par un joueur
- char rechercherCarte (Carte *pCarte, int nbCarte, Couleur couleurCherche, Valeur valeurCherche)
 - cherche dans un tableau de carte si il y a une certaine couleur ou une certaine valeur ou les deux
- char recherCarteSuperieur (Carté *pCarte, int nbCarte, Carte carteCherche, Couleur atout, Couleur entame)
- cherche dans un tableau de carte si il y a une carte d'une force suppérieur a carteCherche
- float sommeForceCarte (Carte *tableáuCarte, int nbCarte, Coúléur atout)
- calucule la somme force des carte dans un tableau

 void trieCarte (Carte tableauCarte[], int nbCarte, Couleur atout)
 - trie un tableau de cartes
- char cartePlaceAvant (Carte carteRefference, Carte carteCompare, Couleur atout)
 - permet de savoir si une carte doit etre placée avant lors du trie des cartes
- void rechercheAnnonce (Carte *pCarteMain, int pointAnonce[], Joueur utilisateur, char *pseudo[])

rechere dans un tableau de cartes qui correspond a la main d'un joureur si il peut faire des anonc, et si oui ajoute les points assacier au tableau dans le tableau des anonce

4.11.1 Description détaillée

ficher contenant les fonctions relative a la gestion des cartes

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.11.2 Documentation des fonctions

4.11.2.1 cartePlaceAvant()

permet de savoir si une carte doit etre placée avant lors du trie des cartes

Paramètres

Carte	carteRefference : carte par raport a laquelle on compare
Carte	carteCompare : carte que l'on compare
Couleur	atout : couleur de l'atout dans la manche

Renvoie

char : renvoie 1 si la catre doit etre placée avant 0 sinon

- < Les deux carte sont elle de la meme famille
- < si carteCompate est un atout
- < si la couleur de la carteCompare est devant celle de carteRefference dans l'énumération(ordre arbitraire)

Définition à la ligne 404 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.11.2.2 carteValide()

ddétermine si une carte peut etre posé par un joueur

Paramètres

Carte	cartPose : carte que l'on veut poser
Carte	pli[] : tableau contenet les carte du pli
Couleur	atout : couleur de l'atout
Carte	*pCarteMainJoueur pointeur sur un tableau de 8 carte contenent les cartes dans la main du joueur
Joueur premierAJouer : premier joueur a jouer dans le pli	
Joueur	parle : joueur en train de jouer

Renvoie

char 1 si la carte est valide 0 sinon

- < Fonction faite a partir de l'oranigrame qui montre comment d©terminer si une carte est valide a partir des r¨gle
- < On verifier que ce n'est pas une carte vide
- < premià re carte du plis?
- < Le joueur posà de il la couleur demandée?
- < l'entame est en atout
- < Le joueur a il un ajout de valeur supérieur au meilleur ajout posé
- < si la carte est un ajout le valeur supérieur au meilleur atout posé
- < Si la carte est un atout
- < Si la carte est dans la couleur demandée
- < Le partemenaire est maitre?
- < Le joueur a il un atout?
- < il y a deja un ajout de posé
- < Le joueur a il un ajout de valeur supérieur au meilleur ajout posé
- < si la carte est un ajout le valeur supérieur au meilleur atout posé
- < Si la carte est un atout alors elle est valide

Définition à la ligne 261 du fichier gestionCarte.c.

 $\label{thm:continuous} \mbox{Voici le graphe des appelants de cette fonction}: \mbox{Voici le graphe des appelants}: \mbox{Voici le graphe}: \mbox{Voici le$

4.11.2.3 distribueCarte()

distribue les cartes entre les différent joueur

Paramètres

Carte	*pCarteMain : pointeur sur le tableau qui stocke les carte dans la main de chaque joueur

Définition à la ligne 12 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.11.2.4 forceCarte()

calcul la force d'une carte, en faisant la probabilitée qu'elle a de gagné contre toutes les autres cartes

Paramètres

Carte	carteACalculer : carte dont on veut connaître la force
Couleur	atout : couleur de l'atout durant la partie
Couleur	entame : couleur de l'entame durant la partie

Renvoie

float : probabilitée de victoir de la carte

- < la carte est un atout ou que la manche se joue en tout atout
- < si on est en tout atout mais que la couleur de la carte n'est pa la couleur de l'entame alors la carte sera plus faible que toutes les cartees dans la bonne couleur
- < la manche est en sans atout et la carte est dans la couleur de l'entame
- < on est en atout d'une certaine couleur ou en sans atoutavec une couleur diffÃ@rente de l'entame
- < quand l'entame est différent de la couleur de la carte et que ce n'est pas un atout alor on pert a chaque fois

Définition à la ligne 80 du fichier gestionCarte.c.

Voici le graphe des appelants de cette fonction :

4.11.2.5 rechercheAnnonce()

rechere dans un tableau de cartes qui correspond a la main d'un joureur si il peut faire des anonc, et si oui ajoute les points assacier au tableau dans le tableau des anonce

Paramètres

Carte	*pCarteMain : pointeur sur le premier élément du tableau qui cintient les cartes de tout les joueur
int	pointAnonce[] : tableau contenant les point d'anonce
Joueur	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur
char	*pseudo[] : tableau contenant les pseudo des 4 joueur

Renvoie

void

< on cherche si le joueur a un carré

Définition à la ligne 435 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.11.2.6 rechercherCarte()

cherche dans un tableau de carte si il y a une certaine couleur ou une certaine valeur ou les deux

Paramètres

Carte	*pCarte : pointeur sur un tableau de carte
int	nbCarte : nombre de carte dans le tableau
Couleur	couleurCherche : couleur que l'on veut chercher dans le tableau (mettre SANS_COULEUR pour rechercher toutes les couleur)
Valeur	valeurCherche : valeur que l'on cherche dans le tableau (mettre SANS_VALEUR pour chercher
	toutes les valeur

Renvoie

char 1 si on a trouvé une carte qui correspond au critère 0 sinon

Définition à la ligne 343 du fichier gestionCarte.c.

Voici le graphe des appelants de cette fonction :

4.11.2.7 rechercherCarteSuperieur()

```
Carte carteCherche,
Couleur atout,
Couleur entame )
```

cherche dans un tableau de carte si il y a une carte d'une force suppérieur a carteCherche

Paramètres

Carte	*pCarte : pointeur sur un tableau de carte
int	nbCarte : nombre de carte dans le tableau
Carte	carteCherche : carte a laquele on veut que la carte cherchée soit supérieur
Couleur	atout : couleur de l'atout dans la manche
Couleur	entame : couleur de l'entame dans la manche

Renvoie

char 1 si on a trouvé une carte qui correspond au critère 0 sinon

Définition à la ligne 355 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.11.2.8 setCarte()

donne une valeur et une couleur a une carte

Paramètres

Carte	*carte : pointeur sur la carte a modifier
Valeur	valeurCarte : valeur a donner a la carte
Couleur	couleurCarte : couleur a donner a la carte

Renvoie

void

Définition à la ligne 50 du fichier gestionCarte.c.

Voici le graphe des appelants de cette fonction :

4.11.2.9 sommeForceCarte()

calucule la somme force des carte dans un tableau

Carte	tableauCarte : tableau de carte sur lequel on veut travailler
int	nbCarte : nombre de carte dans le tableau
Gé©égé/þar/Þo	vatout : couleur de l'atout sur la manche

float : la valeur de lla somme des force des carte

Définition à la ligne 369 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.11.2.10 supprimeCarte()

supprime une carte d'un tableau et réduit la taille de ce tableu par le bas

Paramètres

Carte	carte[] : tableau conteneant les carte a modifier
int	nbCarte : nombre total de carte
int	carteASupprimer : indice dans le tableau de la carte a enlever

Renvoie

void

Définition à la ligne 56 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.11.2.11 trieCarte()

trie un tableau de cartes

Paramètres

	Carte	tableauCarte[] : tableau contenant les cartes a trier
ſ	int	nbCarte nombre de carte dans le tableau
	Couleur	atout : couleur de l'atout dans la manche

Renvoie

void

< On r \tilde{A} \otimes alise un tri a bulle pour mettre atout du plus fort au moin fort au d \tilde{A} \otimes but de la main puis les carte par famille

< Si la carte dois aller devant dans le tableau

Définition à la ligne 386 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.11.2.12 vainqueurPli()

```
Joueur vainqueurPli (

Carte pli[],

Couleur atout,

Joueur premierAJouer )
```

donne le vainquer d'un pli

Paramètres

Carte	pli[] : tableau contenant les carte du pli
Couleur	atout : couleur de l'atout durant cette manche
Joueur	premierAJouer : joueur qui pose la première carte du pli

Renvoie

Joueur : le jouer qui a gagné le pli

< La carte du joueur que l'on test est plus forte que la meilleur actuelle alors ca devient la meilleure

Définition à la ligne 64 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.12 Référence du fichier gestionCarte.h

ficher header contenant les protypes des fonctions relative a la gestion des cartes

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Fonctions

- void distribueCarte (Carte *pCarteMain)
 - distribue les cartes entre les différent joueur
- void setCarte (Carte *carte, Valeur valeurCarte, Couleur couleurCarte)
 - donne une valeur et une couleur a une carte
- void supprimeCarte (Carte carte[], int nbCarte, int carteASupprimer)
- supprime une carte d'un tableau et réduit la taille de ce tableu par le bas
- Joueur vainqueurPli (Carte pli[], Couleur atout, Joueur premierAJouer)
 - donne le vainquer d'un pli
- float forceCarte (Carte carteACalculer, Couleur atout, Couleur entame)
 - calcul la force d'une carte, en faisant la probabilitée qu'elle a de gagné contre toutes les autres cartes
- char carteValide (Carte cartePose, Carte pli[], Couleur atout, Carte *pCarteMainJoueur, Joueur premierA←
 Jouer, Joueur parle)
 - ddétermine si une carte peut etre posé par un joueur
- char rechercherCarte (Carte *pCarte, int nbCarte, Couleur couleurCherche, Valeur valeurCherche)

- cherche dans un tableau de carte si il y a une certaine couleur ou une certaine valeur ou les deux
 char rechercherCarteSuperieur (Carte *pCarte, int nbCarte, Carte carteCherche, Couleur atout, Couleur entame)
 - cherche dans un tableau de carte si il y a une carte d'une force suppérieur a carteCherche
- float sommeForceCarte (Carte *tableauCarte, int nbCarte, Couleur atout)
 - calucule la somme force des carte dans un tableau
- void trieCarte (Carte tableauCarte[], int nbCarte, Couleur atout)
 - trie un tableau de cartes
- char cartePlaceAvant (Carte carteRefference, Carte carteCompare, Couleur atout)
 - permet de savoir si une carte doit etre placée avant lors du trie des cartes
- void rechercheAnnonce (Carte *pCarteMain, int pointAnonce[], Joueur utilisateur, char *pseudo[])

rechere dans un tableau de cartes qui correspond a la main d'un joureur si il peut faire des anonc, et si oui ajoute les points assacier au tableau dans le tableau des anonce

4.12.1 Description détaillée

ficher header contenant les protypes des fonctions relative a la gestion des cartes

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.12.2 Documentation des fonctions

4.12.2.1 cartePlaceAvant()

permet de savoir si une carte doit etre placée avant lors du trie des cartes

Paramètres

Carte	carteRefference : carte par raport a laquelle on compare
Carte	carteCompare : carte que l'on compare
Couleur	atout : couleur de l'atout dans la manche

Renvoie

char : renvoie 1 si la catre doit etre placée avant 0 sinon

- < Les deux carte sont elle de la meme famille
- < si carteCompate est un atout
- < si la couleur de la carteCompare est devant celle de carteRefference dans l'Ã@numÃ@ration(ordre arbitraire)

Définition à la ligne 404 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.12.2.2 carteValide()

ddétermine si une carte peut etre posé par un joueur

Paramètres

Carte	cartPose : carte que l'on veut poser
Carte	pli[] : tableau contenet les carte du pli
Couleur	atout : couleur de l'atout
Carte	*pCarteMainJoueur pointeur sur un tableau de 8 carte contenent les cartes dans la main du joueur
Joueur	premierAJouer : premier joueur a jouer dans le pli
Joueur	parle : joueur en train de jouer

Renvoie

char 1 si la carte est valide 0 sinon

- < Fonction faite a partir de l'oranigrame qui montre comment d\(\tilde{\text{Q}}\) terminer si une carte est valide a partir des r\(\tilde{\text{C}}\) "gle
- < On verifier que ce n'est pas une carte vide
- < premià "re carte du plis?
- < Le joueur posà de il la couleur demandée?
- < l'entame est en atout
- < Le joueur a il un ajout de valeur supérieur au meilleur ajout posé
- < si la carte est un ajout le valeur supÃ@rieur au meilleur atout posÃ@
- < Si la carte est un atout
- < Si la carte est dans la couleur demandée
- < Le partemenaire est maitre?

- < Le joueur a il un atout?
- < il y a deja un ajout de posé
- < Le joueur a il un ajout de valeur supérieur au meilleur ajout posé
- < si la carte est un ajout le valeur supérieur au meilleur atout posé
- < Si la carte est un atout alors elle est valide

Définition à la ligne 261 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.12.2.3 distribueCarte()

distribue les cartes entre les différent joueur

Paramètres

Carte	*pCarteMain : pointeur sur le tableau qui stocke les carte dans la main de chaque joueur

Définition à la ligne 12 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.12.2.4 forceCarte()

calcul la force d'une carte, en faisant la probabilitée qu'elle a de gagné contre toutes les autres cartes

Paramètres

Carte	carteACalculer : carte dont on veut connaitre la force
Couleur	atout : couleur de l'atout durant la partie
Couleur	entame : couleur de l'entame durant la partie

Renvoie

float : probabilitée de victoir de la carte

< la carte est un atout ou que la manche se joue en tout atout

- < si on est en tout atout mais que la couleur de la carte n'est pa la couleur de l'entame alors la carte sera plus faible que toutes les cartees dans la bonne couleur
- < la manche est en sans atout et la carte est dans la couleur de l'entame
- < on est en atout d'une certaine couleur ou en sans atoutavec une couleur diff©rente de l'entame
- < quand l'entame est diffĩrent de la couleur de la carte et que ce n'est pas un atout alor on pert a chaque fois

Définition à la ligne 80 du fichier gestionCarte.c.

Voici le graphe des appelants de cette fonction :

4.12.2.5 rechercheAnnonce()

rechere dans un tableau de cartes qui correspond a la main d'un joureur si il peut faire des anonc, et si oui ajoute les points assacier au tableau dans le tableau des anonce

Paramètres

Carte	*pCarteMain : pointeur sur le premier élément du tableau qui cintient les cartes de tout les joueur
int	pointAnonce[] : tableau contenant les point d'anonce
Joueur	utilisateur : donne la position de l'utilisateur, mettre SANS_Joueur pour faire une partie avec uniquement des ordinateur
char	*pseudo[] : tableau contenant les pseudo des 4 joueur

Renvoie

void

< on cherche si le joueur a un carré

Définition à la ligne 435 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.12.2.6 rechercherCarte()

cherche dans un tableau de carte si il y a une certaine couleur ou une certaine valeur ou les deux

Carte	*pCarte : pointeur sur un tableau de carte
int	nbCarte : nombre de carte dans le tableau
Couleur	couleurCherche : couleur que l'on veut chercher dans le tableau (mettre SANS_COULEUR pour rechercher toutes les couleur)
Valeur	valeurCherche : valeur que l'on cherche dans le tableau (mettre SANS_VALEUR pour chercher toutes les valeur

Renvoie

char 1 si on a trouvé une carte qui correspond au critère 0 sinon

Définition à la ligne 343 du fichier gestionCarte.c.

Voici le graphe des appelants de cette fonction :

4.12.2.7 rechercherCarteSuperieur()

cherche dans un tableau de carte si il y a une carte d'une force suppérieur a carteCherche

Paramètres

Carte	*pCarte : pointeur sur un tableau de carte
int	nbCarte : nombre de carte dans le tableau
Carte	carteCherche : carte a laquele on veut que la carte cherchée soit supérieur
Couleur	atout : couleur de l'atout dans la manche
Couleur	entame : couleur de l'entame dans la manche

Renvoie

char 1 si on a trouvé une carte qui correspond au critère 0 sinon

Définition à la ligne 355 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.12.2.8 setCarte()

donne une valeur et une couleur a une carte

Carte	*carte : pointeur sur la carte a modifier
Valeur	valeurCarte : valeur a donner a la carte
Couleur	couleurCarte : couleur a donner a la carte

Renvoie

void

Définition à la ligne 50 du fichier gestionCarte.c.

Voici le graphe des appelants de cette fonction :

4.12.2.9 sommeForceCarte()

calucule la somme force des carte dans un tableau

Paramètres

Carte	tableauCarte: tableau de carte sur lequel on veut travailler
int	nbCarte : nombre de carte dans le tableau
Couleur	atout : couleur de l'atout sur la manche

Renvoie

float : la valeur de lla somme des force des carte

Définition à la ligne 369 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.12.2.10 supprimeCarte()

supprime une carte d'un tableau et réduit la taille de ce tableu par le bas

Paramètres

Carte	carte[] : tableau conteneant les carte a modifier
int	nbCarte : nombre total de carte
int	carteASupprimer : indice dans le tableau de la carte a enlever

Renvoie

void

Définition à la ligne 56 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.12.2.11 trieCarte()

trie un tableau de cartes

Paramètres

Carte	tableauCarte[] : tableau contenant les cartes a trier
int	nbCarte nombre de carte dans le tableau
Couleur	atout : couleur de l'atout dans la manche

Renvoie

void

- < On r \tilde{A} ©alise un tri a bulle pour mettre atout du plus fort au moin fort au d \tilde{A} ©but de la main puis les carte par famille
- < Si la carte dois aller devant dans le tableau

Définition à la ligne 386 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.12.2.12 vainqueurPli()

donne le vainquer d'un pli

Paramètres

Carte	pli[] : tableau contenant les carte du pli
Couleur	atout : couleur de l'atout durant cette manche
Joueur	premierAJouer : joueur qui pose la première carte du pli

Renvoie

Joueur : le jouer qui a gagné le pli

< La carte du joueur que l'on test est plus forte que la meilleur actuelle alors ca devient la meilleure

Définition à la ligne 64 du fichier gestionCarte.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.13 Référence du fichier gestionFichier.c

ficher source contenant les fonctions relative à la gestion des fichiers

```
#include "main.h"
```

Graphe des dépendances par inclusion de gestionFichier.c:

Fonctions

```
— int ecriturePseudo (char *pPseudo, FILE *pFichier)
```

fonction qui recherche dans un fichier si le joueur est deja enregistré au le rajoute à la fin si ce n'est pas le cas

int ecrireStatistique (FILE *fichier, int ligne, int statAModifier, int type)

modifie la valeur d'une statistique dans le fichier de sauvegarde des scores — int ecrireLeaderboard (FILE *fichier, char *pseudo, int scoreJ)

vérifie si un nouveau record à été battu et l'ajoute si c'est le cas

4.13.1 Description détaillée

ficher source contenant les fonctions relative à la gestion des fichiers

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.13.2 Documentation des fonctions

4.13.2.1 ecrireLeaderboard()

vérifie si un nouveau record à été battu et l'ajoute si c'est le cas

FILE	*fichier : pointeur vers fichier de meilleurs score
char	*pPseudo : pointeur vers le pseudo du joueur
int	score : score a éventuellement enregistrer

Renvoie

0

Définition à la ligne 97 du fichier gestionFichier.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.13.2.2 ecrireStatistique()

modifie la valeur d'une statistique dans le fichier de sauvegarde des scores

Paramètres

FILE	*fichier : pointeur vers le fichier
int	ligne : ligne du fichier à laquelle sont sauvegardée les statistiques du joueur
int	statAModifier : la nouvelle statistique à enregistrer
int	type : vaut 1 pour mdifier le nombre de victoires, vaut 2 pour changer le score max, vaut 3 pour changer le nombre de manche min pour une victoire

Renvoie

valeur enregistrée

Définition à la ligne 44 du fichier gestionFichier.c.

Voici le graphe des appelants de cette fonction :

4.13.2.3 ecriturePseudo()

fonction qui recherche dans un fichier si le joueur est deja enregistr \tilde{A} au le rajoute \tilde{A} la fin si ce n'est pas le cas

char	*pPseudo: pointeur vers le nom du joueur
FILE	*pFichier : pointeur vers le fichier contenant les statistiques

Renvoie

le numero de la ligne dans lequel est noté le pseudo

Définition à la ligne 11 du fichier gestionFichier.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.14 Référence du fichier gestionFichier.h

ficher header contenant les prototypes des fonctions relative à la gestion des fichiers

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Fonctions

- int ecriturePseudo (char *pPseudo, FILE *pFichier)
- fonction qui recherche dans un fichier si le joueur est deja enregistr \tilde{A} © au le rajoute \tilde{A} la fin si ce n'est pas le cas int ecrireStatistique (FILE *fichier, int ligne, int statAModifier, int type)
- modifie la valeur d'une statistique dans le fichier de sauvegarde des scores
- int ecrireLeaderboard (FILE *fichier, char *pseudo, int score)

vÃ@rifie si un nouveau record à Ã@tÃ@ battu et l'ajoute si c'est le cas

4.14.1 Description détaillée

ficher header contenant les prototypes des fonctions relative à la gestion des fichiers

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.14.2 Documentation des fonctions

4.14.2.1 ecrireLeaderboard()

vérifie si un nouveau record à été battu et l'ajoute si c'est le cas

FILE	*fichier : pointeur vers fichier de meilleurs score
char	*pPseudo : pointeur vers le pseudo du joueur
int	score : score a éventuellement enregistrer

Renvoie

0

Définition à la ligne 97 du fichier gestionFichier.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.14.2.2 ecrireStatistique()

modifie la valeur d'une statistique dans le fichier de sauvegarde des scores

Paramètres

FILE	*fichier : pointeur vers le fichier	
int	ligne : ligne du fichier à laquelle sont sauvegardée les statistiques du joueur	
int	statAModifier : la nouvelle statistique à enregistrer	
int	type : vaut 1 pour mdifier le nombre de victoires, vaut 2 pour changer le score max, vaut 3 pour changer le nombre de manche min pour une victoire	

Renvoie

valeur enregistrée

Définition à la ligne 44 du fichier gestionFichier.c.

Voici le graphe des appelants de cette fonction :

4.14.2.3 ecriturePseudo()

fonction qui recherche dans un fichier si le joueur est deja enregistr \tilde{A} au le rajoute \tilde{A} la fin si ce n'est pas le cas

char	*pPseudo: pointeur vers le nom du joueur	
FILE	*pFichier : pointeur vers le fichier contenant les statistiques	

Renvoie

le numero de la ligne dans lequel est noté le pseudo

Définition à la ligne 11 du fichier gestionFichier.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.15 Référence du fichier ia.c

ficher source contenant les fonctions relative au inteligences artificielles

```
#include "main.h"
```

Graphe des dépendances par inclusion de ia.c:

Fonctions

- Contrat proposeContratla (Joueur parle, Carte *pCarteMain, Contrat dernierContrat)
 proposition par l'ia d'un contrat
- int choixCartelA (Joueur joueur, Carte *pMainJoueur, Carte pli[], Joueur premierJoueur, Couleur atout, int carteRestante)

proposition de la pose d'une carte par l'IA

4.15.1 Description détaillée

ficher source contenant les fonctions relative au inteligences artificielles

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.15.2 Documentation des fonctions

4.15.2.1 choixCartelA()

proposition de la pose d'une carte par l'IA

Joueur	joueur : determine qu'elle IA doit jouer	
Carte	*pMainJoueur : pointeur vers le tableau qui enregistre la main de l'IA	
Carte	pli[] : tableau qui enregistre les cartes jouées pendant le pli	
Joueur dernierVainqueur : defini quel joueur à posé la première carte		
Couleur	ouleur atout : defini quelle couleur est en atout	
int	carteRestante : nombre de cartes resantes en main	

Renvoie

debug for(int i=0;i<nbCarteValide;i++){ printf(" $|%d| \n",numCarteValide[i]$); } afficheMain(pCarteValidee); debug for(int i=0;i<offset;i++){ printf(" $|%d| \n",numCarteGagnante[i]$); } afficheMain(pCarteGagante); Définition à la ligne 60 du fichier ia.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.15.2.2 proposeContratla()

proposition par l'ia d'un contrat

Paramètres

Joueur	parle : joueur qui parle
*pCarteMain pointeur sur le tableau qui contient les cartes du ju	
Contrat	dernierContrat : dernier contrat qui a été proposé

Renvoie

Contrat

- < Permet de déterminer de manière euristique la meilleur valeur de seuilMinPrise et seuilMaxPrise
- < ON coinche!

Définition à la ligne 11 du fichier ia.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.16 Référence du fichier ia.h

ficher header contenant les prototypes des fonctions relative au inteligences artificielles Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Fonctions

- Contrat proposeContratla (Joueur parle, Carte *pCarteMain, Contrat dernierContrat)
- proposition par l'ia d'un contrat
 int choixCartelA (Joueur joueur, Carte *pMainJoueur, Carte pli[], Joueur dernierVainqueur, Couleur atout, int carteRestante)

proposition de la pose d'une carte par l'IA

4.16.1 Description détaillée

ficher header contenant les prototypes des fonctions relative au inteligences artificielles

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.16.2 Documentation des fonctions

4.16.2.1 choixCarteIA()

proposition de la pose d'une carte par l'IA

Paramètres

Joueur	joueur : determine qu'elle IA doit jouer	
Carte	*pMainJoueur : pointeur vers le tableau qui enregistre la main de l'IA	
Carte	pli[] : tableau qui enregistre les cartes jouées pendant le pli	
Joueur	dernierVainqueur : defini quel joueur à posé la première carte	
Couleur	atout : defini quelle couleur est en atout	
int	carteRestante : nombre de cartes resantes en main	

Renvoie

```
debug for(int i=0;i<nbCarteValide;i++){ printf("|\%d| \n",numCarteValide[i]); } afficheMain(pCarteValidee); debug for(int i=0;i<offset;i++){ printf("|\%d|| \n",numCarteGagnante[i]); } afficheMain(pCarteGagante); Définition à la ligne 60 du fichier ia.c.
```

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.16.2.2 proposeContratla()

proposition par l'ia d'un contrat

Paramètres

Joueur	parle : joueur qui parle	
Carte	Carte *pCarteMain pointeur sur le tableau qui contient les cartes du jou	
Contrat	dernierContrat : dernier contrat qui a été proposé	

Renvoie

Contrat

- < Permet de déterminer de manià re euristique la meilleur valeur de seuilMinPrise et seuilMaxPrise
- < ON coinche!

Définition à la ligne 11 du fichier ia.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.17 Référence du fichier main.c

ficher contenant la fonction main

```
#include "main.h"
Graphe des dépendances par inclusion de main.c:
```

Fonctions

```
— int main (int argc, char *argv[])
Entrée du programme.
```

4.17.1 Description détaillée

ficher contenant la fonction main

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.17.2 Documentation des fonctions

4.17.2.1 main()

```
int main (
          int argc,
          char * argv[] )
```

Entrée du programme.

Auteur

Florian.C

Renvoie

EXIT_SUCCESS - Arrêt normal du programme

< CODE FINAL

< definit la taille de la fennetre a 50 lignes et 91 colones

< FIN CODE FINAL

Définition à la ligne 19 du fichier main.c.

Voici le graphe d'appel pour cette fonction :

4.18 Référence du fichier main.h

ficher la dÃ@claration des contrante, les Ã@numÃ@rations, les structe et l'inclusion des haeder du projet

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>
#include <time.h>
#include <string.h>
#include "tableau.h"
#include "acquisition.h"
#include "affichage.h"
#include "gestionCarte.h"
#include "general.h"
#include "formatageChaine.h"
#include "autre.h"
#include "ia.h"
#include "sous-menus.h"
#include "gestionFichier.h"
```

Graphe des dépendances par inclusion de main.h: Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Structures de données

```
    struct Carte
        represente une carte d'un jeu de 32 cartes
    struct Contrat
        represente un contrat à la belote coichée
```

Macros

```
#define TAILLE_MAXI_PESEUDO 20
#define TAILLE_MAXI_COULEUR 8
#define NB_JOUEUR 4
#define TAILLE_MAXI_MESSAGE 500
#define NB_TOĀTAL_CARTE 32
#define NIVĒAU_IA 2
#define NB_CARRACTERE_SCORE 33
#define POSITION_NB_VICTOIRE 21
#define POSITION_SCORE_MAX 25
#define POSITION_NB_MANCHES_POUR_GAGNER 30
#define NB_CARRACTĒRE_LEADĒRBOARD 26
#define POSITION_RECORD_VICTOIRE 21
#define DEBUG_MODE 0
#define MODE_1_MANCHE 0
```

Définitions de type

```
typedef enum Couleur Couleur
typedef enum Valeur Valeur
typedef enum Joueur Joueur
typedef enum Coinche Coinche
typedef enum NbPoint NbPoint
typedef struct Carte Carte
typedef struct Contrat Contrat
typedef enum TypeMessage TypeMessage
```

Énumérations

```
— enum Couleur {
   SANS COULEUR = 0, COEUR = 1, PIQUE = 2, CARREAU = 3,
   TREFLE = 4, TOUT ATOUT = 5, SANS ATOUT = 6 }
       Les diffÃ@rentes couleur d'atout possible a la belote coinchÃ@e.
 – enum Valeur {
   SANS_VALEUR = 0, AS = 1, VALET = 11, DAME = 12,
   ROI = 13, SEPT = 7, HUIT = 8, NEUF = 9,
   DIX = 10
       Les diffÃ@rentes valeur possible des cartes d'un jeu de 32 cartes.
— enum Joueur {
   SANS JOUEUR = 0, NORD = 1, EST = 2, SUD = 3,
   OUEST = 4 }
       Les diffÃ@rente joueur lors d'une partie.
 – enum Coinche { NORMAL = 0, COINCHE = 1, SURCOINCHE = 2 }
       indique il un contrat est coinché ou surcoinché
— enum NbPoint {
   ZERO = 0, QUATRE_VINGT = 80, QUATRE_VINGT_DIX = 90, CENT = 100,
   CENT DIX = 110, CENT VINGT = 120, CENT TRENTE = 130, CENT QUARANTE = 140,
CENT_CINQUANTE = 150, CENT_SOIXANTE = 160, CAPOT = 170, GENERALE = 180 }
— enum TypeMessage { SANS_MESSAGE, POSE_CARTE, RESULTAT_PLI, RESULTAT_MANCHE }
       contante à passer en paramà "tre à la fonction génà "reMessage() pour donner le type de message a génà "reMessage()
```

4.18.1 Description détaillée

ficher la dÃ@claration des contrante, les Ã@numÃ@rations, les structe et l'inclusion des haeder du projet

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.18.2 Documentation des macros

4.18.2.1 **DEBUG_MODE**

```
#define DEBUG_MODE 0
```

0 pour le mode normal, 1 pour le mode d©bug

Définition à la ligne 26 du fichier main.h.

4.18.2.2 MODE_1_MANCHE

```
#define MODE_1_MANCHE 0
```

0 pour le mode normal, 1 pour le mode ou une partie est constitu \tilde{A} d'une uique manche Structure et \tilde{A} num \tilde{A} rations

Définition à la ligne 29 du fichier main.h.

4.18.2.3 NB_CARRACTERE_LEADERBOARD

```
#define NB_CARRACTERE_LEADERBOARD 26
```

Définition à la ligne 24 du fichier main.h.

4.18.2.4 NB_CARRACTERE_SCORE

#define NB_CARRACTERE_SCORE 33

Définition à la ligne 20 du fichier main.h.

4.18.2.5 NB_JOUEUR

#define NB_JOUEUR 4

Définition à la ligne 16 du fichier main.h.

4.18.2.6 NB_TOATAL_CARTE

#define NB_TOATAL_CARTE 32

Définition à la ligne 18 du fichier main.h.

4.18.2.7 NIVEAU_IA

#define NIVEAU_IA 2

Définition à la ligne 19 du fichier main.h.

4.18.2.8 POSITION_NB_MANCHES_POUR_GAGNER

#define POSITION_NB_MANCHES_POUR_GAGNER 30

Définition à la ligne 23 du fichier main.h.

4.18.2.9 POSITION_NB_VICTOIRE

#define POSITION_NB_VICTOIRE 21

Définition à la ligne 21 du fichier main.h.

4.18.2.10 POSITION_RECORD_VICTOIRE

#define POSITION_RECORD_VICTOIRE 21

Définition à la ligne 25 du fichier main.h.

4.18.2.11 POSITION_SCORE_MAX

#define POSITION_SCORE_MAX 25

Définition à la ligne 22 du fichier main.h.

4.18.2.12 TAILLE_MAXI_COULEUR

#define TAILLE_MAXI_COULEUR 8

Définition à la ligne 15 du fichier main.h.

4.18.2.13 TAILLE_MAXI_MESSAGE

#define TAILLE_MAXI_MESSAGE 500

Définition à la ligne 17 du fichier main.h.

4.18.2.14 TAILLE_MAXI_PESEUDO

#define TAILLE_MAXI_PESEUDO 20

< Constantes

Définition à la ligne 14 du fichier main.h.

4.18.3 Documentation des définitions de type

4.18.3.1 Carte

typedef struct Carte Carte

4.18.3.2 Coinche

typedef enum Coinche Coinche

4.18.3.3 Contrat

typedef struct Contrat Contrat

4.18.3.4 Couleur

typedef enum Couleur Couleur

4.18.3.5 Joueur

typedef enum Joueur Joueur

4.18.3.6 NbPoint

typedef enum NbPoint NbPoint

4.18.3.7 TypeMessage

typedef enum TypeMessage TypeMessage

Fontions standard Fontions créée dans le cardre du projet

4.18.3.8 Valeur

typedef enum Valeur Valeur

4.18.4 Documentation du type de l'énumération

4.18.4.1 Coinche

enum Coinche

indique il un contrat est coinché ou surcoinché

Valeurs énumérées

NORMAL	
COINCHE	Le contrat n'est pas coiché, par défaut
SURCOINCHE	Le contrat est Coinché

Définition à la ligne 97 du fichier main.h.

4.18.4.2 Couleur

enum Couleur

Les diffÃ@rentes couleur d'atout possible a la belote coinchÃ@e.

toute les couleur disponible ainsi que tout atouts et sans atout sert a la fois pour le type Carte et le type Contrat

Valeurs énumérées

SANS_COULEUR	
COEUR	Couleur par défaut, pour une carte vide
PIQUE	Couleur coeur
CARREAU	Couleur pique
TREFLE	Couleur carreau
TOUT_ATOUT	Couleur trefle
SANS_ATOUT	Tout atout

Définition à la ligne 39 du fichier main.h.

4.18.4.3 Joueur

enum Joueur

Les différente joueur lors d'une partie.

Les joueur sont nomé par les different point cardinaut correspondant à leur placement sur la table

Valeurs énumérées

SANS_JOUEUR	
NORD	Joueur par défaut, quand il n'y a pas de joueur ou pour l'initialitation
EST	Joueur en haut
SUD	Joueur à gauche
OUEST	Joueur en bas

Définition à la ligne 81 du fichier main.h.

4.18.4.4 NbPoint

enum NbPoint

Valeurs énumérées

ZERO	
QUATRE_VINGT	Le contrat vaut zéro point, par défaut correspont a un joueur qui passe ou pour l'initiatilation
QUATRE_VINGT_DIX	Le contrat vaut 80 points
CENT	Le contrat vaut 90 points
CENT_DIX	Le contrat vaut 100 points
CENT_VINGT	Le contrat vaut 110 points
CENT_TRENTE	Le contrat vaut 120 points
CENT_QUARANTE	Le contrat vaut 130 points
CENT_CINQUANTE	Le contrat vaut 140 points
CENT_SOIXANTE	Le contrat vaut 150 points
CAPOT	Le contrat vaut 160 points
GENERALE	Le contrat est un capot

Définition à la ligne 112 du fichier main.h.

4.18.4.5 TypeMessage

enum TypeMessage

contante à passer en paramà "tre à la fonction gÃ@nà "reMessage() pour donner le type de message a gÃ@nÃ@rer

Valeurs énumérées

SANS MESSAGE	
	non de manage e efficiency non defect
POSE_CARTE	pas de message a afficher, par defaut
RESULTAT_PLI	lorsqu'un joueur vien de poser une carte
RESULTAT_MANCHE	anonce le vainceur d'un plis

Définition à la ligne 164 du fichier main.h.

4.18.4.6 Valeur

enum Valeur

Les diffÃ@rentes valeur possible des cartes d'un jeu de 32 cartes.

Les valeurs associée on été placée de manière arbitraire et n'ont pas d'importance sur le programme Pour simplifier, le tout atout et le sans atout sont considéré comme des couleurs

Valeurs énumérées

SANS_VALEUR	
AS	Valeur par défaut, pour une carte vide
VALET	Valeur As
DAME	Valeur Valet
ROI	Valeur Dame
SEPT	Valeur Roi
HUIT	Valeur 7
NEUF	Valeur 8
DIX	Valeur 9

Définition à la ligne 60 du fichier main.h.

4.19 Référence du fichier sous-menus.c

#include "main.h"

Graphe des dépendances par inclusion de sous-menus.c:

Fonctions

- void parametre (char *pseudo[])
- affiche le sous menu des paramètre, permet de changer le psedo de tout les ordinateur
- int leaderboard (FILE *fichier)
- fonction qui recupère les meilleurs score dans un ficher et les affiche dans la console
- int statistiqueJoueur (FILE *fichier, int ligne)

fonction qui recupère les statistiques du joueur et les affiches dans la console

4.19.1 Documentation des fonctions

4.19.1.1 leaderboard()

```
int leaderboard (  {\tt FILE} \, * \, fichier \, ) \\
```

fonction qui recupère les meilleurs score dans un ficher et les affiche dans la console

Paramètres

pointeur	vers le fichier à lire
----------	------------------------

Renvoie

0

Définition à la ligne 32 du fichier sous-menus.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.19.1.2 parametre()

affiche le sous menu des paramètre, permet de changer le psedo de tout les ordinateur

Paramètres

char *pseudo[] : tableau de pointeurs contenant les pseudos des différents joueurs

Renvoie

void

Définition à la ligne 11 du fichier sous-menus.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.19.1.3 statistiqueJoueur()

```
int statistiqueJoueur (  {\tt FILE} \, * \, fichier, \\ {\tt int} \, \, ligne \, )
```

fonction qui recupère les statistiques du joueur et les affiches dans la console

Paramètres

pointeur	vers le fichier de sauvegarde des scores
numero	de la ligne à laquelle sont ecrites les statistiques du joueur

Renvoie

0

Définition à la ligne 85 du fichier sous-menus.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.20 Référence du fichier sous-menus.h

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Fonctions

- void parametre (char *pseudo[])
- affiche le sous menu des paramètre, permet de changer le psedo de tout les ordinateur int leaderboard (FILE *fichier)
- fonction qui recupère les meilleurs score dans un ficher et les affiche dans la console int statistiqueJoueur (FILE *fichier, int ligne)

fonction qui recupère les statistiques du joueur et les affiches dans la console

4.20.1 Documentation des fonctions

4.20.1.1 leaderboard()

```
int leaderboard (
     FILE * fichier )
```

fonction qui recupère les meilleurs score dans un ficher et les affiche dans la console

Paramètres

pointeur	vers le fichier à lire

Renvoie

0

Définition à la ligne 32 du fichier sous-menus.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.20.1.2 parametre()

affiche le sous menu des paramètre, permet de changer le psedo de tout les ordinateur

Paramètres

```
char *pseudo[] : tableau de pointeurs contenant les pseudos des différents joueurs
```

Renvoie

void

Définition à la ligne 11 du fichier sous-menus.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.20.1.3 statistiqueJoueur()

```
int statistiqueJoueur (
     FILE * fichier,
     int ligne )
```

fonction qui recupère les statistiques du joueur et les affiches dans la console

Paramètres

pointeur	vers le fichier de sauvegarde des scores
numero	de la ligne à laquelle sont ecrites les statistiques du joueur

Renvoie

0

Définition à la ligne 85 du fichier sous-menus.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.21 Référence du fichier tableau.c

ficher source contenant les fonctions relative à la gestion des tableau

```
#include "main.h"
```

Graphe des dépendances par inclusion de tableau.c:

Fonctions

```
— int sommeTableau (int tableau[], int tailleTableau)
       fait la somme des valeur d'un tableau contenant des entiers
float moyenneTableau (int tableau[], int tailleTableau)
       fait la moyenne des valeur d'un tableau contenant des entiers
void copie (int tableauOriginal[], int tableauCopie[], int tailleTableau)
       copie un tableau dans un autre
int mini (int tableau[], int tailleTableau)
       trouve la valeur minimum d'un tableau
 int maxi (int tableau[], int tailleTableau)
       trouve la valeur maximum d'un tableau
void zeroSiSuperieur (int tableau[], int tailleTableau, int maximum)
       met a 0 toute les valeur d'un tableau qui sont superieur a un maximum
void ordonnerTableau (int tableau[], int tailleTableau)
       classe les valeur d'un tableau dans l'orde croissant

    void afficheTableau (int tableau[], int tailleTableau)

       affiche un tableau
void acquiertTableau (int tableau[], int tailleTableau)
       demande a l'utilisateurde saisir les valeurs d'un tableau
   void constanteTableau (int tableau[], int tailleTableau, int valeur)
       donne a toute les valeur d'un tableau une constante

    int rechercheDichotomie (int tableau[], int tailleTableau, int nCherche)

       chercherche par dicotomie si une valeur est au moin une fois dans le tableau

    void aleatoireTableau (int tableau[], int tailleTableau, int mini, int maxi)

       donne des nombres aléatoire au valeur d'un tableau
```

4.21.1 Description détaillée

ficher source contenant les fonctions relative à la gestion des tableau

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.21.2 Documentation des fonctions

4.21.2.1 acquiertTableau()

demande a l'utilisateurde saisir les valeurs d'un tableau

tableau[]	: nom du tableau a saisir
tailleTableau	: la taille du tableau, entier positif

Renvoie

void

Définition à la ligne 93 du fichier tableau.c.

4.21.2.2 afficheTableau()

affiche un tableau

Paramètres

tableau[]	: le tableau a afficher
tailleTableau	: la taille du tableau, entier positif

Renvoie

void

Définition à la ligne 83 du fichier tableau.c.

4.21.2.3 aleatoireTableau()

donne des nombres aléatoire au valeur d'un tableau

Paramètres

tableau[]	: tableau a modifier
tailleTableau	: taille du tableau entier positif
mini	: valeur minimum des nombres aléatoires
maxi	: valeur maximum des nombres aléatoires

Renvoie

Définition à la ligne 127 du fichier tableau.c.

Voici le graphe d'appel pour cette fonction :

4.21.2.4 constanteTableau()

donne a toute les valeur d'un tableau une constante

Paramètres

tableau[]	: tableau a modifier
tailleTableau	: taille du tableau entier positif
valeur	: valeur a donner a toutes les valeur du tableau

Renvoie

void

Définition à la ligne 101 du fichier tableau.c.

4.21.2.5 copie()

```
void copie (
                int tableauOriginal[],
               int tableauCopie[],
                int tailleTableau )
```

copie un tableau dans un autre

Paramètres

tableauOriginal[]	: tableaau que l'on veut copier
tableauCopie[]	: tableau dans lequel on copie tableauOriaginal
tailleTableau	: la taille du tableau, entier positif

Renvoie

void

Définition à la ligne 25 du fichier tableau.c.

4.21.2.6 maxi()

trouve la valeur maximum d'un tableau

Paramètres

tableau	[] : tableau dont on veut travailler
tailleTableau	: la taille du tableau, entier positif

Renvoie

int valeur la plus grande du tableau

Définition à la ligne 45 du fichier tableau.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.21.2.7 mini()

trouve la valeur minimum d'un tableau

Paramètres

tableau	[] : tableau dont on veut travailler
tailleTableau	: la taille du tableau, entier positif

Renvoie

int valeur la plus petite du tableau

Définition à la ligne 33 du fichier tableau.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.21.2.8 moyenneTableau()

fait la moyenne des valeur d'un tableau contenant des entiers

tableau[]	: le nom du tableau
tailleTableau	: la taille du tableau, entier positif

Renvoie

la moyenne des valeur du tableau, réel

Définition à la ligne 20 du fichier tableau.c.

Voici le graphe d'appel pour cette fonction :

4.21.2.9 ordonnerTableau()

classe les valeur d'un tableau dans l'orde croissant

Paramètres

tableau[]	: le tableau à classer
tailleTableau	: la taille du tableau, entier positif

Renvoie

void

Définition à la ligne 66 du fichier tableau.c.

4.21.2.10 rechercheDichotomie()

```
int rechercheDichotomie (
          int tableau[],
          int tailleTableau,
          int nCherche )
```

chercherche par dicotomie si une valeur est au moin une fois dans le tableau

Paramètres

tableau[]	: tableau dans lequel on fait la recherche
tailleTableau	: taille du tableau entier positif
nCherche	nombre que l'on cherche dans le tableau

Renvoie

```
int: 1 si on a trouvé, 0 sinon
```

Définition à la ligne 108 du fichier tableau.c.

4.21.2.11 sommeTableau()

fait la somme des valeur d'un tableau contenant des entiers

Paramètres

tableau[]	: le nom du tableau
tailleTableau	: la taille du tableau, entier positif

Renvoie

la somme des valeur du table entier

Définition à la ligne 11 du fichier tableau.c.

Voici le graphe des appelants de cette fonction :

4.21.2.12 zeroSiSuperieur()

```
void zeroSiSuperieur (
          int tableau[],
          int tailleTableau,
          int maximum )
```

met a 0 toute les valeur d'un tableau qui sont superieur a un maximum

Paramètres

tableau[]	: tableau sur lequel on veut travailler
tailleTableau	: la taille du tableau, entier positif
maximum	: valeur maximum au dessus de quoi la valeur est remise a 0

Renvoie

void

Définition à la ligne 57 du fichier tableau.c.

4.22 Référence du fichier tableau.h

ficher header contenant les prototypes des fonctions relative à la gestion des tableau

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier:

Fonctions

```
— int sommeTableau (int tableau[], int tailleTableau)
       fait la somme des valeur d'un tableau contenant des entiers
— float moyenneTableau (int tableau[], int tailleTableau)
       fait la moyenne des valeur d'un tableau contenant des entiers
void copie (int tableauOriginal[], int tableauCopie[], int tailleTableau)
       copie un tableau dans un autre
— int mini (int tableau[], int tailleTableau)
       trouve la valeur minimum d'un tableau
int maxi (int tableau[], int tailleTableau)
       trouve la valeur maximum d'un tableau
void zeroSiSuperieur (int tableau[], int tailleTableau, int maximum)
       met a 0 toute les valeur d'un tableau qui sont superieur a un maximum

    void ordonnerTableau (int tableau[], int tailleTableau)

       classe les valeur d'un tableau dans l'orde croissant
void afficheTableau (int tableau[], int tailleTableau)
       affiche un tableau

    void acquiertTableau (int tableau[], int tailleTableau)

       demande a l'utilisateurde saisir les valeurs d'un tableau

    void constante Tableau (int tableau[], int taille Tableau, int valeur)

       donne a toute les valeur d'un tableau une constante

    void aleatoireTableau (int tableau[], int tailleTableau, int mini, int maxi)

       donne des nombres aléatoire au valeur d'un tableau
int rechercheDichotomie (int tableau[], int tailleTableau, int nCherche)
       chercherche par dicotomie si une valeur est au moin une fois dans le tableau
```

4.22.1 Description détaillée

ficher header contenant les prototypes des fonctions relative à la gestion des tableau

Auteur

Carlo.A & Florian.C

Version

v1.0

Date

12 juin 2020

4.22.2 Documentation des fonctions

4.22.2.1 acquiertTableau()

demande a l'utilisateurde saisir les valeurs d'un tableau

tableau[]	: nom du tableau a saisir
tailleTableau	: la taille du tableau, entier positif

Renvoie

void

Définition à la ligne 93 du fichier tableau.c.

4.22.2.2 afficheTableau()

affiche un tableau

Paramètres

tableau[]	: le tableau a afficher
tailleTableau	: la taille du tableau, entier positif

Renvoie

void

Définition à la ligne 83 du fichier tableau.c.

4.22.2.3 aleatoireTableau()

donne des nombres aléatoire au valeur d'un tableau

Paramètres

tableau[]	: tableau a modifier
tailleTableau	: taille du tableau entier positif
mini	: valeur minimum des nombres aléatoires
maxi	: valeur maximum des nombres aléatoires

Renvoie

Définition à la ligne 127 du fichier tableau.c.

Voici le graphe d'appel pour cette fonction :

4.22.2.4 constanteTableau()

donne a toute les valeur d'un tableau une constante

Paramètres

tableau[]	: tableau a modifier
tailleTableau	: taille du tableau entier positif
valeur	: valeur a donner a toutes les valeur du tableau

Renvoie

void

Définition à la ligne 101 du fichier tableau.c.

4.22.2.5 copie()

```
void copie (
                int tableauOriginal[],
               int tableauCopie[],
                int tailleTableau )
```

copie un tableau dans un autre

Paramètres

tableauOriginal[]	: tableaau que l'on veut copier
tableauCopie[]	: tableau dans lequel on copie tableauOriaginal
tailleTableau	: la taille du tableau, entier positif

Renvoie

void

Définition à la ligne 25 du fichier tableau.c.

4.22.2.6 maxi()

trouve la valeur maximum d'un tableau

Paramètres

tableau	[] : tableau dont on veut travailler
tailleTableau	: la taille du tableau, entier positif

Renvoie

int valeur la plus grande du tableau

Définition à la ligne 45 du fichier tableau.c.

Voici le graphe d'appel pour cette fonction : Voici le graphe des appelants de cette fonction :

4.22.2.7 mini()

trouve la valeur minimum d'un tableau

Paramètres

tableau	[] : tableau dont on veut travailler
tailleTableau	: la taille du tableau, entier positif

Renvoie

int valeur la plus petite du tableau

Définition à la ligne 33 du fichier tableau.c.

 $\label{thm:continuous} \mbox{Voici le graphe des appelants de cette fonction}: \mbox{Voici le graphe des appelants}: \mbox{Voici le graphe}: \mbox{Voici le$

4.22.2.8 moyenneTableau()

fait la moyenne des valeur d'un tableau contenant des entiers

tableau[]	: le nom du tableau
tailleTableau	: la taille du tableau, entier positif

Renvoie

la moyenne des valeur du tableau, réel

Définition à la ligne 20 du fichier tableau.c.

Voici le graphe d'appel pour cette fonction :

4.22.2.9 ordonnerTableau()

classe les valeur d'un tableau dans l'orde croissant

Paramètres

tableau[]	: le tableau à classer
tailleTableau	: la taille du tableau, entier positif

Renvoie

void

Définition à la ligne 66 du fichier tableau.c.

4.22.2.10 rechercheDichotomie()

```
int rechercheDichotomie (
          int tableau[],
          int tailleTableau,
          int nCherche )
```

chercherche par dicotomie si une valeur est au moin une fois dans le tableau

Paramètres

tableau[]	: tableau dans lequel on fait la recherche
tailleTableau	: taille du tableau entier positif
nCherche	nombre que l'on cherche dans le tableau

Renvoie

```
int: 1 si on a trouvé, 0 sinon
```

Définition à la ligne 108 du fichier tableau.c.

4.22.2.11 sommeTableau()

fait la somme des valeur d'un tableau contenant des entiers

Paramètres

tableau[]	: le nom du tableau
tailleTableau	: la taille du tableau, entier positif

Renvoie

la somme des valeur du table entier

Définition à la ligne 11 du fichier tableau.c.

Voici le graphe des appelants de cette fonction :

4.22.2.12 zeroSiSuperieur()

```
void zeroSiSuperieur (
          int tableau[],
          int tailleTableau,
          int maximum )
```

met a 0 toute les valeur d'un tableau qui sont superieur a un maximum

Paramètres

tableau[]	: tableau sur lequel on veut travailler
tailleTableau	: la taille du tableau, entier positif
maximum	: valeur maximum au dessus de quoi la valeur est remise a 0

Renvoie

void

Définition à la ligne 57 du fichier tableau.c.

Index

acquiertTableau	affichage.h, 19
tableau.c, 89	afficheMenuPrincipal
tableau.h, 95	affichage.c, 14
acquisition.c, 9	affichage.h, 19
acquisition.h, 9	afficheMenuSelection
acquisitionEntierAvecMessage, 10	affichage.c, 15
acquisitionEntierSansMessage, 10	affichage.h, 20
acquisitionEntierSansMessageAvecConsigne, 10	afficheSousMenus
acquisitionEntierSecurise, 11	affichage.c, 15
acquisitionOuiNonSansMessage, 11	affichage.h, 20
acquisitionPseudoAvecMessage, 11	afficheTableau
acquisitionEntierAvecMessage	tableau.c, 90
acquisition.h, 10	tableau.h, 96
acquisitionEntierSansMessage	ajusteEchelle
acquisition.h, 10	autre.c, 23
•	autre.h, 26
acquisitionEntierSansMessageAvecConsigne	aleatoireTableau
acquisition.h, 10	tableau.c, 90
acquisitionEntierSecurise	tableau.h, 96
acquisition.h, 11	aligneModifieChaine
acquisitionOuiNonSansMessage	formatageChaine.c, 30
acquisition.h, 11	formatageChaine.h, 35
acquisitionPseudoAvecMessage	annonceContrat
acquisition.h, 11	general.c, 41
affichage.c, 12	general.h, 48
afficheContrat, 13	AS
afficheInterfacePli, 13	main.h, 85
afficheMain, 14	atout
afficheMenuPrincipal, 14	Contrat, 6
afficheMenuSelection, 15	autre.c, 22
afficheSousMenus, 15	ajusteEchelle, 23
modifieTailleFenetre, 16	joue1000Partie, 23
proposeContratUtilisateur, 16	joueurSuivant, 23
affichage.h, 17	nbAleatoire, 24
afficheContrat, 18	pointPli, 24
afficheInterfacePli, 18	setContrat, 25
afficheMain, 19	autre.h, 25
afficheMenuPrincipal, 19	ajusteEchelle, 26
afficheMenuSelection, 20	joue1000Partie, 27
afficheSousMenus, 20	joueurSuivant, 27
modifieTailleFenetre, 21	nbAleatoire, 27
proposeContratUtilisateur, 21	pointPli, 28
afficheContrat	setContrat, 28
affichage.c, 13	Selcontrat, 28
affichage.h, 18	calculPointManche
afficheInterfacePli	general.c, 42
affichage.c, 13	general.h, 48
affichage.h, 18	CAPOT
afficheMain	main.h, 84
affichage.c. 14	CARREAU

main.h, 83	DAME
Carte, 5	main.h, 85
couleur, 5	DEBUG_MODE
main.h, 82	main.h, 79
valeur, 5	decoupeChaine
cartePlaceAvant	formatageChaine.c, 31
gestionCarte.c, 54	formatageChaine.h, 37
gestionCarte.h, 62	distribueCarte
carteValide	gestionCarte.c, 55
gestionCarte.c, 54	gestionCarte.h, 64
gestionCarte.h, 63	DIX
CENT	main.h, 85
main.h, 84	ecrireLeaderboard
CENT_CINQUANTE	
main.h, 84	gestionFichier.c, 69
CENT_DIX	gestionFichier.h, 71
main.h, 84	ecrireStatistique
CENT_QUARANTE	gestionFichier.c, 70
main.h, 84	gestionFichier.h, 72
CENT_SOIXANTE	ecriturePseudo
main.h, 84	gestionFichier.c, 70
CENT_TRENTE	gestionFichier.h, 72
main.h, 84	EST
CENT VINGT	main.h, 84
main.h, 84	faura O auta
centreChaine	forceCarte
formatageChaine.c, 30	gestionCarte.c, 56
formatageChaine.h, 36	gestionCarte.h, 64
centreModifieChaine	formatageChaine.c, 29
formatageChaine.c, 30	aligneModifieChaine, 30
formatageChaine.h, 36	centreChaine, 30
choixCartelA	centreModifieChaine, 30
ia.c, 73	decoupeChaine, 31
ia.h, 75	formateCarte, 31
COEUR	formateContrat, 32
	formatePseudo, 32
main.h, 83 COINCHE	genereMessage, 33
	rempliEspace, 34
main.h, 83	stockeInfoCarte, 34
Coinche	formatageChaine.h, 35
main.h, 82, 83	aligneModifieChaine, 35
coinche	centreChaine, 36
Contrat, 6	centreModifieChaine, 36
constanteTableau	decoupeChaine, 37
tableau.c, 91	formateCarte, 37
tableau.h, 97	formateContrat, 38
Contrat, 6	formatePseudo, 38
atout, 6	genereMessage, 39
coinche, 6	rempliEspace, 39
main.h, 82	stockeInfoCarte, 40
nbPoint, 7	formateCarte
preneur, 7	formatageChaine.c, 31
copie	formatageChaine.h, 37
tableau.c, 91	formateContrat
tableau.h, 97	formatageChaine.c, 32
Couleur	formatageChaine.h, 38
main.h, 82, 83	formatePseudo
couleur	formatageChaine.c, 32
Carte, 5	formatageChaine.h, 38
,	

annonceContrat, 41 calculPointManche, 42 manche, 43 menuPrincipal, 44 nouvellePartie, 44 pil, 45 poseCarte, 46 poroposeContrat, 47 annonceContrat, 48 calculPointManche, 48 initialisation generalh, 47 annonceContrat, 48 calculPointManche, 48 initialisation generalh, 49 manche, 50 nouvellePartie, 50 pil, 51 poseCarte, 52 proposeContrat, 52 GENERALE main, 84 distribueCarte, 55 foreCarte, 56 rechercheCarte, 57 rechercherCarte, 59 sommeForeCarte, 59 sommeForeCarte, 59 sommeForeCarte, 60 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forceCarte, 66 setCarte, 68 setCarte, 69 seriesCarte, 67 supprimeCarte, 67 supp	managed at 40	acritura Dagueda - 70
calculPointManche, 42 initialisation, 42 manche, 43 menuPrincipal, 44 nouvellePartie, 44 pli, 45 poseCarte, 46 proposeContrat, 46 proposeContrat, 46 general, h, 47 annonceContrat, 48 calculPointManche, 48 initialisation, 49 manche, 49 menuPrincipal, 50 nouvellePartie, 50 pli, 51 poseCarte, 52 proposeContrat, 52 generalh, 87 generalh, 87 generalh, 87 main, 84 genereMessage formatageChaine, c, 33 formatageChaine, 33 gestionCarte, 6, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 rechercherCarte, 57 rechercherCarte, 59 supprimeCarte, 60 vainqueurPli, 61 gestionCarte, 64 foreceCarte, 64 foreceCarte, 64 foreceCarte, 66 sommeForeCarte, 67 rechercherAnnonce, 65 rechercherAnnonce, 65 rechercherAnnonce, 65 rechercherCarte, 67 setCarte, 68 sousmencerde, 69 vainqueurPli, 61 gestionCarte, 64 foreceCarte, 66 sommeForeCarte, 67 rechercherCarte, 67 rechercherCarte, 67 rechercherCarte, 68 rechercherCarte, 67 rechercherCarte, 67 rechercherCarte, 68 rechercherCarte, 69 sousmencerde, 60 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forecCarte, 64 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 66 sommeForeCarte, 67 rechercherCarte, 67 rechercherCarte, 68 rechercherCarte, 67 rechercherCarte, 68 rechercherCarte, 67 rechercherCarte, 68 rechercherCarte, 69 setCarte, 68 sommeForeCarte, 67 rechercherCarte, 68 rechercherCarte, 67 rechercherCarte, 68 rechercherCarte, 68 rechercherCarte, 69 setCarte, 68 sommeForeCarte, 67 rechercherCarte, 68 rechercherCarte, 69 setCarte, 68 rechercherCarte, 69 setCarte, 68 rechercherCarte, 69 setCarte, 68 rechercherCarte, 69 setCarte, 68 sommeForeCarte, 67 rechercherCarte, 68 rechercherCarte, 69 setCarte, 60 setCarte, 6	general.c, 40	ecriturePseudo, 72
militalisation, 42 manche, 43 menuPrincipal, 44 nouvellePartie, 44 pil, 45 poseCarte, 46 poseCarte, 46 poseCarte, 47 annonceContrat, 48 calculPointManche, 48 initialisation, 49 manche, 49	•	HUIT
manche, 43 menuPrincipal, 44 nouvellePartie, 44 pli, 45 poseCarte, 46 proposeContrat, 46 general, h, 47 annonceContrat, 48 calculPointManche, 48 initialisation, 49 manche, 49 menuPrincipal, 50 pli, 51 poseCarte, 52 proposeContrat, 52 GENERALE mainh, 84 genereMessage formatageChaine, a, 33 formatageChaine, h, 39 gestionCarte, c, 53 distribueCarte, 56 rechercherCarte, 57 rechercherCarte, 57 rechercherCarte, 57 setCarte, 60 vainqueurPii, 61 gestionCarte, 60 vainqueurPii, 61 gestionCarte, 64 foreCarte, 64 rechercheAnnonce, 55 rechercherCarte, 67 rechercheRanner, 68 setCarte, 66 sormmeForceCarte, 67 rechercheRanner, 68 setCarte, 66 setCarte, 67 supprimeCarte, 67 s		_
menuPrincipal, 44 nouvellePartie, 44 pii, 45 poseCarte, 46 proposeContrat, 46 general.h, 47 annonceContrat, 48 calculPointManche, 48 initialisation, 49 manche, 49 manche, 49 manche, 49 manche, 49 menuPrincipal, 50 nouvellePartie, 50 pii, 51 poseCarte, 52 proposeContrat, 52 GENERALE main.h, 84 genereMessage formatageChaine, 33 gestionCarte, 5, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 foreoCarte, 56 rechercherCarteSuperieur, 57 setCarte, 60 reinermeCarte, 60 ricCarte, 61 rechercherCarteSuperieur, 57 setCarte, 60 reinermeCarte, 60 reinermeCarte, 60 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 66 rechercherCarte, 66 rechercherCarte, 67 rechercherCarte, 68 rechercherCarte, 66 rechercherCarte, 67 rechercherCarte, 68 rechercherCarte, 68 rechercherCarte, 66 rechercherCarte, 66 rechercherCarte, 67 rechercherCarte, 68 rechercherCarte, 68 rechercherCarte, 69 ecrireStatistique, 70 ecriturePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard,		,
nouvellePartie, 44 pli, 45 pi, 45 poseCarte, 46 poseCarte, 46 poseContrat, 47 annonocContrat, 48 calculPointManche, 48 initialisation, 49 manche, 49 manche, 49 mancherincipal, 50 nouvellePartie, 50 pli, 51 poseCarte, 52 proposeContratia, 76 initialisation general.c, 42 general.h, 49 mancherincipal, 50 nouvellePartie, 50 pli, 51 poseCarte, 52 proposeContratia, 76 initialisation general.c, 42 general.h, 49 joue1000Partie autre.c, 23 autre.h, 27 Joueur main.h, 82, 84 jouerSuivant autre.c, 23 autre.h, 27 Joueur main.h, 82, 84 jouerSuivant autre.c, 23 autre.h, 27 gestionCarte, 56 foreAcarte, 59 sommatageChaine.c, 33 formatageChaine.c, 36 carteValide, 54 distribueCarte, 55 foreCarte, 56 rechercherCarteSuperieur, 57 rechercherCarteSuperieur, 57 rechercherCarte, 59 sommeForeCarte, 60 tricCarte, 60 vainqueurPil, 61 gestionCarte, 64 foreCarte, 64 foreCarte, 64 foreCarte, 64 foreCarte, 64 foreCarte, 65 rechercherCarte, 66 sommeForeCarte, 67 rechercherCarte, 67 rechercherCarte, 68 rechercherCarte, 69 ecrireStatistique, 70 ecriturePseudo, 70 gestionFichier.h, 71 poseContratl, 74 choixCarte, 81 initialisation general.c, 42 general.h, 49 initialisation general.c, 42 initialisation general.c, 42 initialisation general.c, 42 general.h, 49 initia		ia.c, 73
pli, 45 poseCarte, 46 proposeContratl, 46 proposeContratl, 46 general.h, 47 annonceContrat, 48 calculPointManche, 48 initialisation, 49 manche, 49 menuPrincipal, 50 nouvellePartie, 50 pii, 51 poseCarte, 52 proposeContrat, 52 GENERALE main.h, 84 genereMessage formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 distribueCarte, 55 forceCarte, 56 rechercherCarteSuperieur, 57 setCarte, 60 trieCarte, 60 trieCarte, 60 trieCarte, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 60 trieCarte, 60 trieCarte, 63 distribueCarte, 64 forceCarte, 64 forceCarte, 64 forceCarte, 65 rechercherCarteSuperieur, 67 setCarte-Readenonce, 65 rechercherCarteSuperieur, 66 sousmmerCarte, 60 trieCarte, 64 forceCarte, 64 forceCarte, 64 forceCarte, 64 forceCarte, 65 rechercherCarteSuperieur, 66 setCarte-Readenonce, 65 rechercherCarteSuperieur, 66 setCarte-Readenonce, 65 rechercherCarteSuperieur, 66 setCarte, 64 forceCarte, 64 forceCarte, 63 distribueCarte, 64 forceCarte, 64 forceCarte, 68 setCarte, 68 setCarte, 68 setCarte, 68 setCarte, 68 vainqueurPii, 61 gestionFichier.h, 69 ecrireStatistique, 70 ecriturePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 ender Attach Acharacter Ecapacity, 80 initialisation general.t., 75 proposeContrat, 75 proposeContrat, 42 poueur autre, 27 poueur main.h, 82 poueur autre, 23 autre, 27 poueur main.h, 82, 84 joueuroural autre, 23 autre, 27 poueur main.h, 82, 84 joueuroural autre, 23 autre, 27 poueur main.h, 82, 84 joueuroural autre, 23 autre, 27 poueur main.h, 82, 84 joueuroural autre, 23 autre, 27 poueur main.h, 82, 84 joueuroural autre, 23 autre, 27 poueur main.h, 82, 84 joueuroural autre, 23 autre, 27 poueur main.h, 82, 84 joueuroural autre, 23 autre, 27 poueur main.h, 82, 84 joueuroura	•	choixCarteIA, 73
poseCarte, 46 proposeContrat, 46 general.h, 47 annonceContrat, 48 calculProintManche, 48 initialisation, 49 manche, 49 manche, 49 manchrincipal, 50 nouvellePartie, 50 pli, 51 poseCarte, 52 proposeContrat, 52 GENERALE main.h, 84 genereMessage formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 rechercheCarte, 56 rechercheCarte, 57 setCarte, 69 supprimeCarte, 60 vainqueurPli, 61 gestionCarte.h, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forceCarte, 64 forceCarte, 64 forceCarte, 64 forceCarte, 64 forceCarte, 66 setCarte, 68 vanqueurPli, 68 gestionFichier.c, 69 ecrireStatistique, 70 ecriturePseudo, 70 gestionFichier.h, 71 ecrifeLeaderboard, 71 ecrifeLeaderboard, 71 ecrifeLeaderboard, 71 lialisation initialisation general.c, 42 general.h, 49 inter.d. 22 autre.h, 27 Joueur main.h, 82, 84 joueurSuivant autre.c, 23 autre.h, 27 init.e, 27 init.e, 28 catePlace, 24 cateValide, 54 main main.c, 77 main.h, 78 AS, 85		proposeContratla, 74
proposeContrat, 46 general.h, 47 annonceContrat, 48 calculPointManche, 48 initialisation, 49 manche, 49 menuPrincipal, 50 nouvellePartie, 50 pli, 51 poseCarte, 52 proposeContrat, 52 GENERALE main.h, 84 genereMessage formatageChaine.c, 33 formatageChaine.c, 33 formatageChaine.s, 39 gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 forceCarte, 56 rechercherCarteSuperieur, 57 setCarte, 60 trieCarte, 60 trieCarte, 60 trieCarte, 64 forceCarte, 64 forceCarte, 64 forceCarte, 65 rechercherCarteSuperieur, 66 setCarte, 68 sommeForceCarte, 65 rechercherCarteSuperieur, 66 setCarte, 68 sommeForceCarte, 67 trieCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecrireStatistique, 70 ecriturePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 ecrireLeaderboard, 71 ecrireLeaderboard, 71 ecrireLeaderboard, 71 nintialisation general.c, 42 general.h, 49 initialisation general.c, 42 general.h, 49 initialisation general.c, 42 general.h, 49 joue1000Partie autre.c, 23 autre.h, 27 joue1000Partie autre.c, 23 autre.h, 27 joueurSuvant autre.c, 23 autre.h, 27 main.h, 82, 84 joueursuvant autre.c, 23 autre.h, 27 main.h, 82 sous-menus.c, 86 sous-menus.c, 80 sous-menus.c, 86 sous-menus.c, 86 sous-menus.c, 86 sous-menu	•	ia.h, 74
general.h, 47 annonceContrat, 48 calculPointManche, 48 initialisation general.c, 42 general.h, 49 manche, 49 menuPrincipal, 50 nouvellePartie, 50 pli, 51 poseCarte, 52 proposeContrat, 52 GENERALE main.h, 84 genereMessage formatageChaine.c, 33 formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 rechercheRonnone, 56 rechercheRoricarte, 57 rechercherCarte, 57 rechercherCarte, 57 selCarte, 60 vainqueurPli, 61 gestionCarte.h, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forecCarte, 64 rechercheAnnonce, 65 rechercheCarte, 67 rechercheAnnonce, 65 rechercheCarte, 66 selCarte, 60 vainqueurPli, 61 gestionCarte, 64 rechercheAnnonce, 65 rechercheCarte, 64 rechercheAnnonce, 65 rechercheCarte, 64 rechercheAnnonce, 65 rechercheCarte, 64 rechercheCarte, 64 rechercheCarte, 65 rechercheCarte, 66 selCarte, 67 supprimeCarte, 67 supprimeCarte, 67 supprimeCarte, 67 supprimeCarte, 67 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 66 selCarte, 68 vanqueurPli, 68 gestionFichier, 69 ecrireStatistique, 70 ecriturePseudo, 70 gestionFichier, 71 ecrifeLeaderboard, 71 NB_CARRACTERE_LEADERBOARD, 80	·	choixCarteIA, 75
annoneContrat, 48 calculPointManche, 48 initialisation, 49 manche, 49 menuPrincipal, 50 nouvellePartie, 50 pii, 51 poseCarte, 52 proposeContrat, 52 GENERALE main, 84 genereMessage formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 distribueCarte, 55 forceCarte, 56 rechercherCarte, 57 rechercherCarte, 57 rechercherCarte, 59 supprimeCarte, 60 vainqueurPli, 61 gestionCarte, 63 distribueCarte, 64 rechercheAnnonce, 65 rechercheCarte, 65 rechercherCarte, 65 rechercherCarte, 67 suincarte, 64 rechercheCarte, 65 rechercheCarte, 65 rechercheCarte, 65 rechercheCarte, 65 rechercheCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercheCarte, 65 rechercheRonnonce, 65 rechercherCarte, 64 rechercheRonnonce, 65 rechercherCarte, 65 rechercherCa	• •	proposeContratla, 76
calculPointManche, 48 initialisation, 49 manche, 49 menuPrincipal, 50 nouvellePartie, 50 pil, 51 poseCarte, 52 pil, 51 poseCarte, 52 general, 84 genereklessage formatageChaine.c, 33 formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 rechercherCarteSuperieur, 57 selCarte, 60 trieCarte, 60 trieCarte, 60 trieCarte, 60 trieCarte, 61 gestionCarte.c, 63 cistribueCarte, 64 forecearte, 64 rechercherCarteSuperieur, 66 selCarte, 68 vainqueurPli, 68 selCarte, 68 selCarte, 68 selCarte, 68 selCarte, 68 selCarte, 68 vainqueurPli, 68 selCarte, 68 corirelCarte, 69 ecrireLeaderboard, 69 ecrirelCarteLeaderboard, 70 gestionFichier, 71 ecriteLeaderboard, 71 selCarteLeaderboard, 71	_	initialisation
initialisation, 49 manche, 49 menuPrincipal, 50 nouvellePartie, 50 pil, 51 poseCarte, 52 proposeContrat, 52 GENERALE main.h, 84 genereMessage formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte, 55 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 rechercherCarteSuperieur, 57 selCarte, 60 vainqueurPli, 61 gestionCarte, 63 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 60 vainqueurPli, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 65 rechercherCarteSuperieur, 67 supprimeCarte, 67 rechercherCarteSuperieur, 67 selcortenerCarteSuperieur, 67 selcortenerCarteSuperieur, 67 carteValide, 63 distribueCarte, 64 rechercherCarteSuperieur, 66 selCarte, 66 selCarte, 68 selCarte, 69 ecrireLeaderboard, 69 ecrireLeaderboard, 71 selCarteResuperieur, 70 eeriturePseudo, 70 gestionFichier., 71 ecrireLeaderboard, 71 selcrieLeaderboard, 71 selcrieLeaderboard, 71 selcrieLeaderboard, 71 selcrieLeaderboard, 71 selcrieLeaderboard, 71 selcrieLeaderboard, 71 supprimeCarte, 17 supprimeCarte, 72 supprimeCarte, 67	•	general.c, 42
manche, 49 menuPrincipal, 50 nouvellePartie, 50 pil, 51 poseCarte, 52 proposeContrat, 52 genereMessage formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte c, 55 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 57 rechercherCarteSuperieur, 57 selCarte, 60 trieCarte, 60 trieCarte, 60 vainqueurPli, 61 gestionCarte, 64 forceCarte, 64 rechercheAnnonce, 65 rechercheAnnonce, 65 rechercheCarte, 60 sounderpli, 61 gestionCarte, 64 forceCarte, 64 rechercheAnnonce, 65 rechercheCarte, 67 supprimeCarte, 60 trieCarte, 60 carteValide, 63 distribueCarte, 64 forceCarte, 64 rechercheAnnonce, 65 rechercheAnnonce, 65 rechercheCarte, 60 carteValide, 63 distribueCarte, 64 forceCarte, 64 rechercheAnnonce, 65 rechercheCartesUperieur, 66 setCarte, 68 setCarte, 68 setCarte, 69 setCarte, 69 vainqueurPli, 68 rechercherCarte, 67 supprimeCarte, 69 ecrireLeaderboard, 70 ecriturePseudo, 70 gestionFichier., 71 ecrireLeaderboard, 71 ecrireLeaderboard, 71 pioueurSuivant autre.c, 23 autre.h, 27 ploueur main.h, 82, 84 joueurSuivant autre.c, 23 autre.h, 27 loueur main.h, 82, 84 joueurSuivant autre.c, 23 autre.h, 27 loueurs, 84 sous-menus.c, 86 sous-menus.c,		general.h, 49
menuPrincipal, 50 nouvellePartie, 50 pil, 51 poseCarte, 52 proposeContrat, 52 groposeContrat, 53 groposeContrat, 53 groposeContrat, 53 groposeContrate, 53 groposeContrate, 53 groposeContrate, 54 distribueCarte, 55 forceCarte, 56 rechercheRCarteSuperieur, 57 rechercherCarteSuperieur, 57 rechercherCarteSuperieur, 57 rechercherCarte, 59 sommeForceCarte, 60 tricCarte, 60 vainqueurPli, 61 gestionCarte, 1, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forceCarte, 64 rechercheAnnonce, 65 rechercherCarteSuperieur, 66 selCarte, 66 sommeForceCarte, 67 supprimeCarte, 67 supprimeCarte, 68 vainqueurPli, 68 gestionFichier, 69 ecrircleaderboard, 69 ecrircleaderboard, 71 ecriroleaderboard, 71 pestineFireCarte Application Applicatio		
nouvellePartie, 50 pii, 51 poseCarte, 52 proposeContrat, 52 proposeContrat, 52 genereMessage formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 forceCarte, 56 rechercherCarte, 57 rechercherCarte, 59 supprimeCarte, 60 trieCarte, 60 vainqueurPii, 61 carteValide, 63 distribueCarte, 64 forceCarte, 64 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 67 supprimeCarte, 66 sestCarte, 66 sestCarte, 66 rechercherCarte, 67 supprimeCarte, 67 rechercherCarte, 69 carteValide, 68 distribueCarte, 64 forceCarte, 64 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 66 sestCarte, 66 rechercherCarte, 67 supprimeCarte, 67 supprimeCarte, 67 rechercherCarte, 67 supprimeCarte, 68 ecrireLeaderboard, 69 ecrireLeaderboard, 69 ecrireLeaderboard, 69 ecrireLeaderboard, 71 MODE_1_MANCHE, 79 min.h., 27 Joueur main.h., 28 Joueur, 82, 84 joueurSuhart main.h., 28 suture.h., 27 Joueur main.h., 28 Joueur, 82, 84 Joueur, 82 Jou		-
pli, 51 poseCarte, 52 proposeContrat, 52 groposeContrat, 52 genereMessage formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 rechercherCarte Superieur, 57 setCarte, 59 supprimeCarte, 60 vainqueurPli, 61 gestionCarte, 63 distribueCarte, 65 rechercherCarte Superieur, 57 setCarte, 66 vainqueurPli, 61 gestionCarte, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 rechercherCarte Superieur, 66 setCarte, 68 vainqueurPli, 61 gestionCarte, 64 rechercherCarte, 64 forecCarte, 64 forecCarte, 64 forecCarte, 64 forecCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 66 setCarte, 66 sommeForceCarte, 67 supprimeCarte, 67 supprimeCarte, 68 vainqueurPli, 68 gestionFichier, 69 ecrireLeaderboard, 69 ecrireLeaderboard, 69 ecrireLeaderboard, 71 ecrireLeaderboard, 71 Malure., 27 Joueur main.h, 82, 84 joueurSuivant autre.c, 23 autre.c, 23 autre.c, 23 autre.c, 23 autre.c, 23 autre.h, 27 main.h, 78 main.c, 77 main.c, 76 main.c, 77 main.c, 76 main.c, 77 main.c, 76 main.c, 77 main.c, 76 main.c, 77 main.h, 78 rechechard., 86 cous-menus.h, 87 CAPOT, 84 CARREAU, 83 CARREAU, 83 CARREAU, 83 CARREAU, 83 CARREAU, 83 CARREAU, 83 CARREAU, 84 CENT_CINQUANTE, 84 CENT_CINQUANTE, 84 CENT_SOIXANTE, 84 CENT_VINGT, 84 CENT_URANTE, 84 CENT_VINGT, 84 CENT_URANTE, 84 CENT_VINGT, 84 CENT_URANTE, 84 CENT_URAN	•	
poseCarte, 52 proposeContrat, 54 proposeContrat, 54 proposeContrat, 53 proposeContrat, 53 proposeContrat, 53 proposeContrat, 53 proposeContrat, 53 proposeContrat, 54 proposeContrate, 55 proposeContrate, 55 proposeContrate, 55 proposeContrate, 56 proposeContrate, 57 proposeContrate, 57 proposeContrate, 57 proposeContrate, 57 proposeContrate, 59 proposeContrate, 61 proposeContrate, 61 proposeContrate, 62 proposeContrate, 63 proposeContrate, 64 proposeContrate, 64 proposeContrate, 64 proposeContrate, 65 proposeContrate, 65 proposeContrate, 64 proposeContrate, 65 proposeContrate, 67 proposeContrate, 68 proposeContrate, 69 proposeCon		•
proposeContrat, 52 GENERALE main.h, 84 genereMessage formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 forceCarte, 56 rechercheAnnonce, 56 rechercheCarteSuperieur, 57 setCarte, 69 supprimeCarte, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 60 vainqueurPli, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forceCarte, 64 forceCarte, 67 rechercherCarteSuperieur, 67 setCarte, 68 vainqueurPli, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forceCarte, 64 forceCarte, 65 rechercherCarteSuperieur, 66 setCarte, 66 sommeForceCarte, 67 supprimeCarte, 64 forceCarte, 64 forceCarte, 64 forceCarte, 65 rechercherCarteSuperieur, 66 setCarte, 66 sommeForceCarte, 67 supprimeCarte, 67 supprimeCarte, 68 vainqueurPli, 68 gestionFichier., 69 ecrireLeaderboard, 69 ecrireLeaderboard, 69 ecrireLeaderboard, 71 mEARACTERE_LEADERBOARD, 80	•	
main.h, 84 genereMessage formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 forceCarte, 56 rechercherCarte, 57 rechercherCartesUperieur, 57 setCarte, 60 vainqueurPli, 61 carteValide, 63 distribueCarte, 63 rechercherCarte, 65 rechercherCarte, 65 sommeForceCarte, 59 sommeForceCarte, 63 carte, 82 carteValide, 63 distribueCarte, 60 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 sommeForceCarte, 59 sommeForceCarte, 59 sommeForceCarte, 60 vainqueurPli, 61 cent_ollowante, 84 forceCarte, 64 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 67 supprimeCarte, 67 trieCarte, 68 vainqueurPli, 68 vainqueurPli, 68 ecrireStatistique, 70 ecrituePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 NB_CARRACTERE_LEADERBOARD, 80	•	
mann., 84 genereMessage formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 forceCarte, 56 rechercheAnnonce, 56 rechercherCarteSuperieur, 57 setCarte, 59 supprimeCarte, 60 trieCarte, 60 vainqueurPli, 61 carteValide, 63 distribueCarte, 64 forceCarte, 64 forceCarte, 66 sous-menus.c, 86 sous-menus.h, 87 main. main.c, 77 main.c, 76 main.n, 77 main.h, 78 AS, 85 CAPOT, 84 CARREAU, 83 Carte, 82 CENT, 84 CENT_CINQUANTE, 84 CENT_CINQUANTE, 84 CENT_CINQUANTE, 84 CENT_CINQUARANTE, 84 CENT_VINGT, 84 rechercheAnnonce, 65 rechercherCarte, 64 rechercherCarte, 65 setCarte, 66 sommeForceCarte, 67 supprimeCarte, 67 supprimeCarte, 67 rechercherCarte, 67 supprimeCarte, 67 supprimeCarte, 68 gestionFichier.c, 69 ecrireLeaderboard, 69 ecrireLeaderboard, 69 ecrireLeaderboard, 71 leaderboard sous-menus.c, 86 sous-menus.c.		-
genereMessage formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 rechercherCarte, 56 rechercherCarte, 57 setCarte, 59 sommeForceCarte, 59 sommeForceCarte, 60 vainqueurPli, 61 carteValide, 63 distribueCarte, 64 forceCarte, 64 rechercherCarte, 65 rechercherCarte, 67 rechercherCarte, 60 vainqueurPli, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forceCarte, 64 rechercheAnnonce, 65 rechercherCarte, 65 rechercherCarte, 66 setCarte, 66 setCarte, 66 setCarte, 66 sommeForceCarte, 67 supprimeCarte, 67 rechercherCarte, 69 ecriresItatistique, 70 eeriruleAcederoard, 69 gestionFichier.h, 71 ecrireLeaderboard, 71 leaderboard sous-menus.c, 86 sous-menus.c.c, 86 sous-meus.ce sous-menus.c.c, 86 sous-menus.c.c, 86 sous-menus.c.c, 80 sous-menus.c.c, 86 sous-menus.ce sous-menus.ce sous-menus.c.c, 86 sous-menus.ce sous-menus.ce sous-menus.ce sous-menus.ce sou	main.h, 84	
formatageChaine.c, 33 formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 forceCarte, 56 rechercheAnnonce, 56 rechercherCarteSuperieur, 57 setCarte, 59 supprimeCarte, 60 vainqueurPli, 61 carteValide, 63 distribueCarte, 64 forceCarte, 64 rechercherCarteSuperieur, 66 sous-menus.c, 86 sous-menus.h, 87 main. main. main.c, 77 main.c, 77 main.c, 76 main.c, 77 main.c, 76 main.c, 77 main.h, 78 AS, 85 CAPOT, 84 CAPOT, 84 CAPOT, 84 CAPOT, 84 CAPOT, 84 CENT_CINQUANTE, 84 CENT_CINQUANTE, 84 CENT_CINQUANTE, 84 CENT_DIX, 84 CENT_DIX, 84 CENT_UARANTE, 84 CENT_UARANTE, 84 CENT_VINGT, 84 CENT_VINCT, 84 CENT_VINCT, 84 CENT_VINCT, 84 CENT_VINCT, 84 CENT_VINCT, 8		autre.h, 27
formatageChaine.h, 39 gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 forceCarte, 56 rechercheAnnonce, 56 rechercheCarte, 57 rechercheCarte, 59 sommeForceCarte, 59 supprimeCarte, 60 vainqueurPli, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forceCarte, 64 rechercheAnnonce, 65 rechercheCarteSuperieur, 66 setCarte, 68 vainqueurPli, 69 sommeForceCarte, 69 rechercheCarteSuperieur, 66 setCarte, 68 setCarte, 69 ecrireLeaderboard, 69 ecriteLeaderboard, 71 gestionCarte, 67 gestionFichier, 71 eorireLeaderboard, 71 main. main. main.c, 77 main.c, 76 main.c, 76 main.c, 77 main.c, 78 AS, 85 CAPOT, 84 CARREAU, 83 CARREAU, 84 CENT_CINQUANTE, 84 CENT_CINQUANTE, 84 CENT_LQUARANTE, 84 CENT_LQUARANTE, 84 CENT_LRENTE, 84 CENT_LRINGT, 84 CENT_LR		laadarhaard
gestionCarte.c, 53 cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 forceCarte, 56 rechercheAnnonce, 56 rechercherCarte, 57 rechercherCarte superieur, 57 setCarte, 59 sommeForceCarte, 59 sommeForceCarte, 60 trieCarte, 60 vainqueurPli, 61 carteValide, 63 distribueCarte, 64 forceCarte, 64 rechercherCarteSuperieur, 66 setCarte, 68 setCarte, 68 setCarte, 68 vainqueurPli, 68 gestionCarte, 69 ecrireLeaderboard, 69 ecrireLeaderboard, 71 gestionCarte.c, 69 ecrireLeaderboard, 71 main.c, 77 main.c, 76 main.c, 76 main.c, 76 main.c, 77 main.h, 78 AS, 85 CAPOT, 84 CARREAU, 83 CARREAU, 83 CARREAU, 83 CARREAU, 83 CARREAU, 83 CARREAU, 83 CARTE, 82 CENT_CINQUANTE, 84 CENT_CINQUANTE, 84 CENT_DIX, 84 CENT_DIX, 84 CENT_DIX, 84 CENT_VINGT, 84 CENT_VINGT, 84 CENT_VINGT, 84 CENT_VINGT, 84 CENT_VINGT, 84 COUR, 83 COINCHE, 83 COI		
cartePlaceAvant, 54 carteValide, 54 distribueCarte, 55 forceCarte, 56 rechercheAnnonce, 56 rechercheCarte, 57 rechercherCarte, 57 setCarte, 59 sommeForceCarte, 59 sommeForceCarte, 60 trieCarte, 60 vainqueurPli, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forceCarte, 64 rechercheCarte, 65 rechercheCarte, 65 rechercheCarte, 66 setCarte, 67 trieCarte, 68 vainqueurPli, 68 gestionForceCarte, 69 ecrireLeaderboard, 69 ecrireLeaderboard, 71 main.c, 77 main.c, 76 main.c,		
distribueCarte, 55 forceCarte, 56 rechercheAnnonce, 56 rechercheCarte, 57 rechercherCarte, 57 rechercherCarte, 57 setCarte, 59 sommeForceCarte, 59 supprimeCarte, 60 vainqueurPli, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forceCarte, 65 rechercherCarte 65 rechercherCarte, 66 rechercherCarte, 67 rechercherCarte, 68 rechercherCarte, 67 rechercherCarte, 68 rechercherCarte, 69 rechercherCarte	cartePlaceAvant, 54	Sous-menus.n, or
distribueCarte, 55 forceCarte, 56 rechercheAnnonce, 56 rechercherCarte, 57 rechercherCarte, 57 rechercherCarte, 59 sommeForceCarte, 59 supprimeCarte, 60 restortellade, 63 distribueCarte, 64 forceCarte, 64 rechercheAnnonce, 65 rechercherCarte, 65 rechercherCarte, 69 supprimeCarte, 60 carte, 82 carte, 84 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forceCarte, 64 rechercheAnnonce, 65 rechercherCarte, 65 rechercherCarte, 66 setCarte, 66 setCarte, 66 setCarte, 66 setCarte, 66 sommeForceCarte, 67 supprimeCarte, 67 trieCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecrireLeaderboard, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 NB_CARRACTERE_LEADERBOARD, 80	carteValide, 54	main
forceCarte, 56 rechercheAnnonce, 56 rechercherCarte, 57 rechercherCarte, 59 setCarte, 59 sommeForceCarte, 59 supprimeCarte, 60 vainqueurPli, 61 gestionCarte, 64 forceCarte, 64 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 67 supprimeCarte, 67 supprimeCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecrireLeaderboard, 71 gestionFichier.c, 71 ecrireLeaderboard, 71 main.h, 78 main.	distribueCarte, 55	
rechercheAnnonce, 56 rechercherCarte, 57 rechercherCarte, 57 rechercherCarteSuperieur, 57 setCarte, 59 sommeForceCarte, 59 supprimeCarte, 60 trieCarte, 60 vainqueurPli, 61 gestionCarte.h, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarte, 65 supprimeCarte, 66 setCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecrireLeaderboard, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 main.h, 78 main.h main.h, 78 main.h main	forceCarte, 56	•
rechercherCarte, 57 rechercherCarteSuperieur, 57 setCarte, 59 sommeForceCarte, 59 supprimeCarte, 60 trieCarte, 60 vainqueurPli, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 rechercherCarteSuperieur, 66 rechercherCarteSuperieur, 66 setCarte, 66 sommeForceCarte, 67 supprimeCarte, 67 trieCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecrireStatistique, 70 ecrireLeaderboard, 71 MODE_1_MANCHE, 79 CAPOT, 84 AS, 85 CAPOT, 84 CARREAU, 83 CENT_CINQUANTE, 84 CENT_DIX, 84 CENT_DIX, 84 CENT_OUARANTE, 84 CENT_SOIXANTE, 84 CENT_VINGT, 84 CENT_WINGT, 84 CENT_WINGT, 84 CENT_VINGT, 84 CENT_WINGT, 84 CENT_WINGT, 84 CENT_WINGT, 84 CENT_WINGT, 84 CENT_BARE CENT_BARE CENT_BARE CENT_BARE CENT_BARE CENT_UIN, 85 COINCHE, 83 COINCHE, 84 COINCHE, 83 COINCHE	rechercheAnnonce, 56	•
rechercherCarteSuperieur, 57 setCarte, 59 SommeForceCarte, 59 SupprimeCarte, 60 Carte, 82 VainqueurPli, 61 CartePlaceAvant, 62 CarteValide, 63 distribueCarte, 64 forceCarte, 65 rechercherCarteSuperieur, 66 sommeForceCarte, 67 supprimeCarte, 67 trieCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecrireLeaderboard, 71 ecrireLeaderboard, 71 CAS, 85 CAPOT, 84 CAPOT, 84 CAPOT, 84 CAPOT, 84 CARREAU, 83 CENT_CINQUANTE, 84 CENT_DIX, 84 CENT_DIX, 84 CENT_SOIXANTE, 84 CENT_VINGT, 84 CENT_COULER, 83 COINCHE, 83	rechercherCarte, 57	•
setCarte, 59 sommeForceCarte, 59 supprimeCarte, 60 trieCarte, 60 vainqueurPli, 61 gestionCarte.h, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 rechercheAnnonce, 65 rechercherCarte, 65 setCarte, 66 setCarte, 66 sommeForceCarte, 67 trieCarte, 68 gestionFichier.c, 69 ecrireLeaderboard, 69 gestionFichier.h, 71 ecrireLeaderboard, 71 CAPOT, 84 CARREAU, 83 CENT_CINQUANTE, 84 CENT_DIX, 84 CENT_DIX, 84 CENT_VINGT, 84 CENT_VINGT, 84 CENT_VINGT, 84 CENT_VINGT, 84 CENT_VINGT, 84 CENT_VINGT, 84 COINCHE, 83 COINCHE, 82 SommeForceCarte, 67 COULeur, 82, 83 SupprimeCarte, 67 DAME, 85 DEBUG_MODE, 79 DIX, 85 GENERALE, 84 HUIT, 85 ecrireLeaderboard, 69 ecrireStatistique, 70 JOUEUR, 82, 84 MODE_1_MANCHE, 79 NB_CARRACTERE_LEADERBOARD, 80	recherCarteSuperieur, 57	
sommeForceCarte, 59 supprimeCarte, 60 trieCarte, 60 trieCarte, 60 vainqueurPli, 61 GestionCarte.h, 61 cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forceCarte, 64 forceCarte, 65 rechercheAnnonce, 65 rechercherCarteSuperieur, 66 sommeForceCarte, 67 supprimeCarte, 68 gestionFichier.c, 69 ecrireLeaderboard, 69 ecrireLeaderboard, 71 ecrireLeaderboard, 71 CARREAU, 83 Carte, 82 Carte, 82 Carte, 82 CENT, 84 CENT_CINQUANTE, 84 CENT_QUARANTE, 84 CENT_SOIXANTE, 84 CENT_TRENTE, 84 CENT_VINGT, 84 CENT_CINCULATE C	setCarte, 59	•
supprimeCarte, 60 trieCarte, 60 vainqueurPli, 61 gestionCarte.h, 61 Carte, 82 CENT, 84 vainqueurPli, 61 CENT_CINQUANTE, 84 CENT_DIX, 84 CartePlaceAvant, 62 CENT_QUARANTE, 84 CarteValide, 63 CENT_SOIXANTE, 84 CENT_TRENTE, 84 CENT_TRENTE, 84 CENT_TRENTE, 84 CENT_VINGT, 84 COEUR, 83 COEUR, 83 COINCHE, 83 COINCHE, 83 COINCHE, 83 SetCarte, 66 Coinche, 82, 83 Coinche, 82, 83 SetCarte, 66 Contrat, 82 SommeForceCarte, 67 Couleur, 82, 83 DEBUG_MODE, 79 DAME, 85 DEBUG_MODE, 79 VainqueurPli, 68 DIX, 85 GestionFichier.c, 69 EST, 84 ecrireLeaderboard, 69 GENERALE, 84 HUIT, 85 ecriturePseudo, 70 Joueur, 82, 84 GestionFichier.h, 71 ecrireLeaderboard, 71 NB_CARRACTERE_LEADERBOARD, 80	sommeForceCarte, 59	•
trieCarte, 60 vainqueurPli, 61 CENT, 84 gestionCarte.h, 61 CartePlaceAvant, 62 CarteValide, 63 CENT_SOIXANTE, 84 GestionCarte, 64 CENT_SOIXANTE, 84 GestionCarte, 64 CENT_TRENTE, 84 GistribueCarte, 64 CENT_TRENTE, 84 GoreCarte, 64 CENT_VINGT, 84 rechercheAnnonce, 65 COEUR, 83 rechercherCarte, 65 coinche, 82, 83 setCarte, 66 SommeForceCarte, 67 SupprimeCarte, 67 Couleur, 82, 83 gestionFichier.c, 69 ecrireLeaderboard, 69 ecrireStatistique, 70 ecrireLeaderboard, 71 MODE_1_MANCHE, 79 REARACTERE_LEADERBOARD, 80	supprimeCarte, 60	
gestionCarte.h, 61	trieCarte, 60	•
cartePlaceAvant, 62 carteValide, 63 distribueCarte, 64 forceCarte, 64 rechercheAnnonce, 65 rechercherCarte 65 rechercherCarte Superieur, 66 setCarte, 66 sommeForceCarte, 67 culeur, 82, 83 supprimeCarte, 67 trieCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecriturePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 CENT_QUARANTE, 84 CENT_SOIXANTE, 84 CENT_SOIXANTE, 84 CENT_SOIXANTE, 84 CENT_SOIXANTE, 84 CENT_SOIXANTE, 84 CENT_SOIXANTE, 84 CENT_VINGT, 84 CENT_VINCT, 84 CENT_VINC	vainqueurPli, 61	CENT CINQUANTE, 84
carteValide, 63 distribueCarte, 64 forceCarte, 64 forceCarte, 64 rechercheAnnonce, 65 rechercherCarte, 65 rechercherCarteSuperieur, 66 sommeForceCarte, 67 couleur, 82, 83 supprimeCarte, 67 trieCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecriturePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 CENT_SOIXANTE, 84 CENT_SOIXANTE, 84 CENT_SOIXANTE, 84 CENT_SOIXANTE, 84 CENT_SOIXANTE, 84 CENT_SOIXANTE, 84 CENT_VINGT, 84	gestionCarte.h, 61	CENT_DIX, 84
distribueCarte, 64 forceCarte, 64 forceCarte, 64 rechercheAnnonce, 65 rechercherCarte, 65 rechercherCartesuperieur, 66 setCarte, 66 sommeForceCarte, 67 supprimeCarte, 67 trieCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecrireLeaderboard, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 CENT_TRENTE, 84 CENT_VINGT, 84 CENT_VINGT, 84 CENT_VINGT, 84 CENT_VINGT, 84 CENT_VINGT, 84 CENT_VINGT, 84 COBUR, 83 COINCHE, 83 Coinche, 82, 83 Coinche, 82, 83 Couleur, 82, 83 DAME, 85 DEBUG_MODE, 79 DIX, 85 EST, 84 EST, 84 HUIT, 85 Joueur, 82, 84 MODE_1_MANCHE, 79 NB_CARRACTERE_LEADERBOARD, 80	cartePlaceAvant, 62	CENT_QUARANTE, 84
forceCarte, 64 rechercheAnnonce, 65 rechercherCarte, 65 rechercherCarte, 65 rechercherCarteSuperieur, 66 setCarte, 66 sommeForceCarte, 67 supprimeCarte, 67 trieCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecrireLeaderboard, 69 ecrireLeaderboard, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 CENT_VINGT, 84 COEUR, 83 COEUR, 83 COINCHE, 82 SOMMEFORCECARTE, 67 DAME, 85 DEBUG_MODE, 79 DIX, 85 GENERALE, 84 HUIT, 85 JOUEUR, 82, 84 MODE_1_MANCHE, 79 NB_CARRACTERE_LEADERBOARD, 80	carteValide, 63	CENT_SOIXANTE, 84
rechercheAnnonce, 65 rechercherCarte, 65 rechercherCarteSuperieur, 66 setCarte, 66 sommeForceCarte, 67 supprimeCarte, 67 trieCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecrireLeaderboard, 69 gestionFichier.h, 71 ecrireLeaderboard, 71 COEUR, 83 COEUR, 83 COINCHE, 83 Coinche, 82, 83 Couleur, 82, 83 DAME, 85 DEBUG_MODE, 79 DIX, 85 EST, 84 GENERALE, 84 HUIT, 85 Joueur, 82, 84 MODE_1_MANCHE, 79 NB_CARRACTERE_LEADERBOARD, 80		CENT_TRENTE, 84
rechercherCarte, 65 rechercherCarteSuperieur, 66 coinche, 82, 83 setCarte, 66 contrat, 82 sommeForceCarte, 67 couleur, 82, 83 supprimeCarte, 67 doinche, 82, 83 contrat, 82 couleur, 82, 83 supprimeCarte, 67 doinche, 82, 83 contrat, 82 couleur, 82, 83 supprimeCarte, 68 pame, 85 doinche, 82, 83 pame, 85 pame, 80 pame, 82, 84 pame, 80	forceCarte, 64	CENT_VINGT, 84
rechercherCarteSuperieur, 66	rechercheAnnonce, 65	COEUR, 83
setCarte, 66 sommeForceCarte, 67 couleur, 82, 83 supprimeCarte, 67 drieCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecrireLeaderboard, 69 ecriturePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 Contrat, 82 Couleur, 82, 83 DAME, 85 DEBUG_MODE, 79 DIX, 85 EST, 84 GENERALE, 84 HUIT, 85 Joueur, 82, 84 MODE_1_MANCHE, 79 NB_CARRACTERE_LEADERBOARD, 80	recherCarte, 65	COINCHE, 83
sommeForceCarte, 67 supprimeCarte, 67 trieCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecrireLeaderboard, 69 ecriturePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 Couleur, 82, 83 DAME, 85 DEBUG_MODE, 79 DIX, 85 EST, 84 GENERALE, 84 HUIT, 85 Joueur, 82, 84 MODE_1_MANCHE, 79 NB_CARRACTERE_LEADERBOARD, 80	recherCarteSuperieur, 66	Coinche, 82, 83
supprimeCarte, 67 trieCarte, 68 DEBUG_MODE, 79 vainqueurPli, 68 DIX, 85 gestionFichier.c, 69 EST, 84 ecrireLeaderboard, 69 ecriturePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 DAME, 85 DEBUG_MODE, 79 DIX, 85 DIX, 85 EST, 84 GENERALE, 84 HUIT, 85 Joueur, 82, 84 MODE_1_MANCHE, 79 NB_CARRACTERE_LEADERBOARD, 80	setCarte, 66	Contrat, 82
trieCarte, 68 vainqueurPli, 68 gestionFichier.c, 69 ecrireLeaderboard, 69 ecriturePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 DEBUG_MODE, 79 DIX, 85 EST, 84 GENERALE, 84 HUIT, 85 Joueur, 82, 84 MODE_1_MANCHE, 79 NB_CARRACTERE_LEADERBOARD, 80	sommeForceCarte, 67	Couleur, 82, 83
vainqueurPli, 68 gestionFichier.c, 69 ecrireLeaderboard, 69 ecrireStatistique, 70 ecriturePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 DIX, 85 EST, 84 GENERALE, 84 HUIT, 85 Joueur, 82, 84 MODE_1_MANCHE, 79 NB_CARRACTERE_LEADERBOARD, 80	supprimeCarte, 67	DAME, 85
gestionFichier.c, 69 EST, 84 ecrireLeaderboard, 69 GENERALE, 84 ecrireStatistique, 70 HUIT, 85 ecriturePseudo, 70 Joueur, 82, 84 gestionFichier.h, 71 MODE_1_MANCHE, 79 ecrireLeaderboard, 71 NB_CARRACTERE_LEADERBOARD, 80		DEBUG_MODE, 79
ecrireLeaderboard, 69 GENERALE, 84 ecrireStatistique, 70 HUIT, 85 ecriturePseudo, 70 Joueur, 82, 84 gestionFichier.h, 71 MODE_1_MANCHE, 79 ecrireLeaderboard, 71 NB_CARRACTERE_LEADERBOARD, 80	•	DIX, 85
ecrireStatistique, 70 ecriturePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 HUIT, 85 Joueur, 82, 84 MODE_1_MANCHE, 79 NB_CARRACTERE_LEADERBOARD, 80	-	EST, 84
ecriturePseudo, 70 gestionFichier.h, 71 ecrireLeaderboard, 71 Joueur, 82, 84 MODE_1_MANCHE, 79 NB_CARRACTERE_LEADERBOARD, 80		GENERALE, 84
gestionFichier.h, 71 MODE_1_MANCHE, 79 ecrireLeaderboard, 71 NB_CARRACTERE_LEADERBOARD, 80	·	
ecrireLeaderboard, 71 NB_CARRACTERE_LEADERBOARD, 80		
	_	
ecrireStatistique, 72 NB_CARRACTERE_SCORE, 80		
	ecrireStatistique, 72	NB_CARRACTERE_SCORE, 80

NB_JOUEUR, 80 NB_TOATAL_CARTE, 80 NB_TOATAL_CARTE NEUF, 85 NIVEAU_JA, 80 NORD, 84 NORNAL, 83 OUEST, 84 PIQUE, 83 POSE_CARTE, 85 POSITION, NB_WANCHES_POUR_GAGNER, 80 POSITION, NB_WICTOIRE, 81 POSITION, SCORE MAX, 81 OUATRE_VINGT_DIX, 84 RESULTAT_PLI, 85 RESULTAT_PLI, 85 RANS_ATOUT, 83 SANS_COULEUR, 83 SANS_COULEUR, 83 SANS_SOULEUR, 84 SANS_MALEUR, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_WESSAGE, 81 TOUT_ATOUT, 83 TREFLE, 83 TIRPELE, 83 TIRPELE, 83 TIRPELE, 83 TIRPELE, 83 TIRPELE, 84 TAILLE_MAXI_WESSAGE, 81 TOUT_ATOUT, 83 TREFLE, 83 TIRPELE, 84 TAILLE_MAXI_MESSAGE, 81 TOUT_ATOUT, 83 TREFLE, 83 TIRPELE, 84 TAILLE_MAXI_MESSAGE, 81 TOUT_ATOUT, 83 TREFLE, 83 TIRPELE, 84 TIRPELE, 83 TIRPELE, 83 TIRPELE, 83 TIRPELE, 83 TIRPELE, 84 TIRPELE, 83 TIRPELE, 83 TIRPELE, 83 TIRPELE, 83 TIRPELE, 84 TIRPELE, 85 TIRPELE, 83 TIRPELE, 84 TIRPELE, 85 TIRPELE, 83 TIRPELE, 83 TIRPELE, 83 TIRPELE, 83 TIRPELE, 84 TIRPELE, 85 TIRPEL, 85 TIRPEL		
NB. TOATAL_CARTE, 80 MB_JOUEUR NDPoint, 82, 84 NB_JOUEUR NEVEN, 85 MB_TOATAL_CARTE NORD, 84 MB_TOATAL_CARTE NORD, 84 MB_TOATAL_CARTE NORD, 84 MB_TOATAL_CARTE NORD, 84 MB_TOATAL_CARTE POSITION, 81 MB_TOATAL_CARTE POSITION, NB_MANCHES, 85 NB_DOATAL_CARTE POSITION, NB_MANCHES, 85 NB_DOATAL_CARTE POSITION, NB_MANCHES, 81 NB_DOATAL_CARTE POSITION, NB_MANCHES, 81 NB_DOATAL_CARTE POSITION, NB_MANCHES, 81 NB_DOATAL_CARTE POSITION, NB_MANCHES, 85 NB_DOATAL_CARTE POSITION, NB_MANCHES, 81 NB_DOATAL_CARTE POSITION, NB_MANCHES, 81 NB_DOATAL_CARTE POSITION, NB_MANCHES, 85 NB_DOATAL_CARTE ROLUT, 83 NB_CARTAL_CARTE ROLUT, 83 NB_DOATAL_CARTE ROLUT, 84 MINDRES ROLUT, 84 M	NB JOUEUR, 80	NB CARRACTERE SCORE
NEPFORT, 82, 84 NEUF, 85 NIVEAU_IA, 80 NORD, 84 NORMAL, 83 OUEST, 84 PIQUE, 83 POSE_CARTE, 85 POSITION, NB WANCHES_POUR_GAGNER, 80 POSITION, NB WANCHES_POUR_GAGNER, 80 POSITION, SCORE_MAX, 81 QUATRE_VINGT, 94 QUATRE_VINGT, 94 QUATRE_VINGT, 94 QUATRE_VINGT, 94 QUATRE_VINGT, 94 QUATRE_VINGT, 94 QUATRE_VINGT, 95 RESULTAT_PLI, 85 RESULTAT_PLI, 85 RESULTAT_PLI, 85 RASU_ACULEUR, 83 SANS_JOULEUR, 83 SANS_JOULEUR, 84 SANS_MESSAGE, 85 SEPT, 85 SEPT, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_MESSAGE, 81 TOUT_TATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 Valeur, 84 Manche general.c, 43 general.h, 99 maxi tableau.c, 91 tableau.c, 91 tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 81 POSITION_NE_MANCHES_POUR_GAGNER main.h, 80 POSITION_NE_MINCHES_POUR_GAGNER main.h, 81 POSITION_NE_MINCH	-	
NEUF, 85 NIVEAU, JA, 80 NORMAL, 83 OUEST, 84 PIOUE, 83 POSTITION, NB, WANCHES, POUR, GAGNER, 80 POSITION, NB, VICTOIRE, 81 POSITION, RECORD, VICTOIRE, 81 QUATRE, VINGT, 34 QUATRE, VINGT, 10X, 84 RESULTAT, PALL, 85 RESULTAT, PLI, 85 ROI, 85 SANS, ATOUT, 83 SANS, COULEUR, 83 SANS, COULEUR, 83 SANS, SOUGUER, 84 SANS, MALEUR, 85 SANS, WALEUR, 85 SANS, WALEUR, 85 SANS, WALEUR, 85 SEPT, 85 SUD, 84 SUNCOINCHE, 83 TAILLE, MAXI, MESSAGE, 81 TAILLE, MAXI, PESSAUD, 81 TOUT, ATOUT, 83 TREFLE, 83 TREFLE	<i>'</i>	
NIVEAU_IA, 80 NORD, 84 NORMAL, 83 OUEST, 84 PIQUE, 83 POSE_CARTE 85 POSITION_NB MANCHES_POUR_GAGNER, 80 POSITION_NB VICTOIRE, 81 POSITION_RECORD_VICTOIRE, 81 POSITION_RECORD_VICTOIRE, 81 POSITION_RECORD_VICTOIRE, 81 POSITION_RECORD_VICTOIRE, 81 POSITION_SCORE MAX, 81 QUATRE_VINNST_DIX, 84 RESULTAT_PLI, 85 RESULTAT_PLI, 85 RESULTAT_PLI, 85 ROI, 85 SANS_ATOUT, 83 SANS_COULEUR, 83 SANS_COULEUR, 83 SANS_OULEUR, 84 SANS_MESSAGE, 85 SEPT, 85 SUD_BA4 SURCOINCHE, 83 TAILLE_MAXI_ESSAGE, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFIEE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.c, 43 general.c, 43 general.c, 43 general.c, 44 general.h, 50 main.h, 83 PIQUE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 82 POSITION_SCORE MAIN.B. ADDITED MITTOR_MAINCHE MAI		_
NORD, 84 NORMAL, 83 OUEST, 84 PIQUE, 83 POSE CARTE, 85 POSITION, NB, MANCHES, POUR_GAGNER, 80 POSITION, NB_WANCHES, POUR_GAGNER, 80 POSITION, RECORD, VICTOIRE, 81 POSITION SCORE MAX, 81 QUATRE_VINOT, 24 QUATRE_VINOT, 24 QUATRE_VINOT, 24 QUATRE_VINOT, 28 GRESULTAT_PLI, 85 ROI, 85 ROI, 85 RANS_ATOUT, 83 SANS_ATOUT, 83 SANS_COULEUR, 83 SANS_OULEUR, 84 SANS_MESSAGE, 85 SANS_VALEUR, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFILE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 38, 85 ZERO, 84 Manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 modifierTailleFenetre affichage, 1, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE 1_MANCHE main.h, 79 modifierTailleFenetre affichage, 1, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE 1_MANCHE main.h, 79 modifierTailleFenetre affichage, 1, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE 1_MANCHE main.h, 81 POSITION_RECORD MAX PESEUDOARD mainin, 81 POSITION_NB_MANCHES_POUR_GAGNER main.h, 81 POSITION_NB_WANCHES_POUR_GAGNER main.h, 81 POSITION_NB_MANCHES_POUR_GAGNER main.h, 81 POSITION_NB_WANCHES_POUR_GAGNER main.h, 81 POSITION_NB_W		
NORMAL, 83 OUEST, 84 PIQUE, 83 POSE_CARTE, 85 POSITION NB. MANCHES. POUR_GAGNER, 80 POSITION_NB_VICTOIRE, 81 POSITION_SCORE MAX, 81 QUATRE_VINGT_DIX, 84 RESULTAT_MANCHE, 85 RESULTAT_PLI, 85 ROI, 85 SANS_ACTOUT, 83 SANS_JOUEUR, 83 SANS_JOUEUR, 84 SANS_MESSAGE, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TOUT_ATOUT, 83 TREFIEE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 maxi tableau.c, 91 tableau.c, 91 tableau.h, 97 menuPrincipal general.h, 49 maxi tableau.c, 91 tableau.c, 91 tableau.h, 97 menuPrincipal general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage,c, 16 affichage,d, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage,c, 16 affichage,d, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 19 modifieTailleFenetre affichage,c, 16 affichage,d, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 19 modifieTailleFenetre affichage,d, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 19 modifieTailleFenetre affichage,d, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 81 POSITION_NB_MANCHES_POUR_GAGNER main.h, 81 POSITION_NB_WICTOIRE main.h, 81 POSITION_NB_VICTOIRE main.h, 81 POSITION_NB_VICTOIRE main.h, 81 POSITION_NB_VICTOIRE main.h, 81 POSITION_NB_CORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_TCOIRE, and repeated and		
OUEST, 84 PIQUE, 83 POSE_CARTE, 85 POSITION_NB_MANCHES_POUR_GAGNER, 80 POSITION_NB_VICTOIRE, 81 POSITION_RECORD_VICTOIRE, 81 POSITION_SCORE_MAX, 81 OUATRE_VINGT, 84 QUATRE_VINGT, 84 QUATRE_VINGT, 85 ROI, 85 ROI, 85 ROI, 85 SANS_ATOUT, 83 SANS_COULEUR, 83 SANS_JOUEUR, 84 SURCOINCHE, 83 SANS_WALEUR, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFLE, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 38, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifie TailleFenetre affichage,c, 16 affichage,c, 16 affichage,h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifie TailleFenetre affichage,c, 16 affichage,h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 19 position_NB_WICTOIRE main.h, 81 preneur Contrat, 7 NBCUF main.h, 85 NIVEAU_IA nanin.h, 85 NIVEAU_IA nanin.h, 83 nouvellePartie general.c, 44 general.h, 50 Tout.atout.s parametre sous-menus.c, 86 position_NB_WICTOIRE main.h, 81 POSITION_NB_WANCHES_POUR_GAGNER main.h, 81 POSITION_NB_WICTOIRE main.h, 81 POSITION_NCORE_MAX main.h, 81 preneur Contrat, 7 proposeContrat		
PIQUE, 83 POSE_CARTE, 85 POSITION_NB_MANCHES_POUR_GAGNER, 80 POSITION_RECORD_VICTOIRE, 81 POSITION_SCORE_MAX, 81 QUATRE_VINAT_BAY QUATRE_VINAT_DIX, 84 RESULTAT_PLI, 85 RESULTAT_PLI, 85 RO, 85 SANS_ATOUT, 83 SANS_COULEUR, 83 SANS_JOUEUR, 84 SANS_MESSAGE, 85 SEPT, 85 SUD, 84 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_PESSEUDO, 81 TOUT_ATOUT_ATOUT_83 TREFLE, 83 TREFLE, 84 Manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 82, 84 nbPoint Contrat, 7 NEUF main.h, 85 NORMAL main.h, 80 NORD main.h, 80 NORD main.h, 83 nouvellePartie general.c, 44 general.h, 50 UEST main.h, 84 POUES main.h, 84 POUES main.h, 84 POUE main.h, 84 POUE main.h, 84 POUE main.h, 85 poseCarte main.h, 83 pii general.c, 45 general.c, 45 general.c, 45 general.h, 50 poseTortone main.h, 80 POSE_CARTE main.h, 85 poseCarte main.h, 81 POSITION_NB_MANCHES_POUR_GAGNER main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_TRECORD_VICTOIRE main.h, 81 POSITION_TRECORD_VICTOIR		
POSE CARTE, 85 POSITION NB_MANCHES_POUR_GAGNER, 80 POSITION_NB_VICTOIRE, 81 POSITION_SCORE_MAX, 81 OUATRE_VINGT_BX, 84 OUATRE_VINGT_DIX, 84 RESULTAT_MANCHE, 85 ROI, 85 ROI, 85 ROI, 85 ROI, 85 ROI, 85 SANS_ATOUT, 83 SANS_COULEUR, 83 SANS_JOUEUR, 84 SANS_MESSAGE, 85 SANS_VALEUR, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifiefailleFenetre affichage.c, 16 affichage.c, 16 affichage.d, 21 moyenneTableau tableau.c, 92 tableau.h, 98 NDE_CARRACTERE_LEADERBOARD		•
POSITION_NB_MANCHES_POUR_GAGNER, 80 POSITION_NB_VICTOIRE, 81 POSITION_SCORE_MAX, 81 OUATRE_VINGT_BLX, 84 RESULTAT_MANCHE, 85 RESULTAT_PLI, 85 ROI, 85 SANS_ATOUT, 83 SANS_COULEUR, 83 SANS_JOUEUR, 84 SANS_MESSAGE, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 main.h, 82 NORMAL main.h, 83 nouvellePartie general.c, 44 general.h, 50 UEST main.h, 84 POSITION_NB_WANCHES_POUR_GAGNER main.h, 83 pli general.c, 44 general.h, 50 general.c, 45 general.h, 50 mini tableau.c, 92 tableau.h, 97 modifieTailleFenetre affichage.c, 16 affichage.c, 16 affichage.d, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE general.h, 50 POSITION_NB_VICTOIRE main.h, 81 POSITION_NE_VICTOIRE main.h, 81 POSITION_TRUCTOIRE main.h, 81 POSITION_TRU		
POSITION_NECORD_VICTOIRE, 81 POSITION_SCORE_MAX, 81 QUATRE_VINGT, 84 QUATRE_VINGT_DIX, 84 RESULTAT_MANCHE, 85 RESULTAT_MANCHE, 85 ROI, 85 SANS_ATOUT, 83 SANS_JOUEUR, 84 SANS_MESSAGE, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_PESEUDO, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFLE, 83 TAYLET, 85 Valeur, 83, 85 Valeur, 84 Manche general.c, 43 general.c, 44 general.c, 45 general.c, 46 general.c, 46 general.c, 46 general.c, 46 general.c, 50 mini tableau.c, 92 tableau.h, 98 MODE_I_MANCHE main.h, 79 modifie failleFenetre affichage.c, 16 affichag	— · · · · · · · · · · · · · · · · · · ·	
POSITION_SCORE_MAX, 81 POSITION_SCORE_MAX, 81 QUATRE_VINGT_DIX, 84 QUATRE_VINGT_DIX, 84 RESULTAT_MANCHE, 85 RESULTAT_PLI, 85 ROI, 85 SANS_ATOUT, 83 SANS_COULEUR, 83 SANS_JOUEUR, 84 SANS_MEESAGE, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_PESEUDO, 81 TAILLE_MAXI_PESEUDO, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFIE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.c, 91 tableau.c, 92 tableau.h, 98 MODE_1_MANCHE affichage.c, 16 affichage.c, 16 affichage.c, 12 moyenneTableau tableau.c, 98 tableau.h, 98 MODE_1_MANCHE affichage.c, 16 affichage.c, 16 affichage.c, 12 moyenneTableau tableau.c, 98 tableau.h, 98 NEUF main.h, 85 NIVEAU_IA main.h, 80 NORD main.h, 84 NORMAL main.h, 83 nouvellePartie general.c, 44 general.h, 50 veralle autableau.c, 92 tableau.h, 99 OUEST main.h, 84 OUCONDER main.h, 84 OVAILET, 85 veralle autableau.c, 98 pointPili autre.c, 24 autre.h, 28 POSE_CARTE main.h, 85 poseCarte general.c, 46 general.c, 46 general.h, 50 POSITION_RECORD_VICTOIRE main.h, 81 porneur Contrat, 7 proposecontrat		
POSITION_SCORE_MAX, 81 QUATRE_VINGT, 84 RESULTAT_MANCHE, 85 RESULTAT_PLI, 85 ROI, 85 SANS_ATOUT, 83 SANS_COULEUR, 83 SANS_JOUEUR, 84 SANS_JOUEUR, 84 SANS_MESSAGE, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TOUT_ATOUT, 83 TREFLE, 83 TREFLE, 83 TREFLE, 83 TREFLE, 83 Tout_ATOUT, 84 TOUT_ATOUT, 84		
QUATRE_VINGT_B1X, 84 RESULTAT_MANCHE, 85 RESULTAT_PLI, 85 RESULTAT_PLI, 85 ROI, 85 RESULTAT_PLI, 85 ROI, 85 ROID RIVER RIN		•
QUATRE_VINGT_DIX, 84 NIVEAU_IA RESULTAT_PLI, 85 main.h, 80 RSOURTE_VIN, 85 main.h, 84 ROI, 85 main.h, 84 SANS_COULEUR, 83 main.h, 83 SANS_JOUEUR, 84 main.h, 83 SANS_MESSAGE, 85 general.c, 44 SANS_VALEUR, 85 general.c, 44 SEPT, 85 gurdeneral.n, 50 SUD, 84 ordonnerTableau SURCOINCHE, 83 tableau.c, 93 TAILLE_MAXI_MESSAGE, 81 OUEST TAILLE_MAXI_PESEUDO, 81 main.h, 84 TOUT_ATOUT, 83 parametre SUSS-menus.h, 84 sous-menus.c, 86 VALET, 85 parametre Valeur, 83, 85 PIQUE ZERO, 84 main.h, 83 maxi peneral.c, 45 general.h, 49 general.n, 83 pointPli autre.c, 24 autre.c, 24 autre.c, 24 autre.b, 28 poseCarte general.h, 50 general.n, 50 mini tableau.c, 92 poseCarte tableau.h, 98		
RESULTAT_MANCHE, 85 main.h, 80 RESULTAT_PUI, 85 NORD ROI, 85 main.h, 84 SANS_ATOUT, 83 NORMAL SANS_DUEUR, 84 NORMAL SANS_MESSAGE, 85 general.n, 50 SEPT, 85 JOUEUR, 81 ordonner Tableau SUD, 84 ordonner Tableau tableau.c, 93 SURCOINCHE, 83 tableau.c, 93 tableau.c, 93 TAILLE_MAXI_COULEUR, 81 OUEST main.h, 84 TOUT ATOUT, 83 TREFLE, 83 parametre TREFLE, 83 parametre sous-menus.c, 86 VALET, 85 PIQUE main.h, 83 Valeur, 83, 85 PIQUE main.h, 83 Tableau.c, 91 parametre sous-menus.c, 86 Sous-menus.h, 88 PIQUE main.h, 83 Post Care peneral.n, 51 pointPil maxi pointPil autre.c, 24 autre.c, 24 aute, 29 poseCarte main.h, 85 poseCarte movenini Tableau, h, 98 main.h, 81 POSITION_NB_MANCHES_POUR_GAGNER <t< td=""><td></td><td>•</td></t<>		•
RESULTAT_PLI, 85 ROI, 85 ROI, 85 SANS_ATOUT, 83 SANS_COULEUR, 83 SANS_JOUEUR, 84 SANS_MESSAGE, 85 SANS_VALEUR, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFLE, 83 TREFLE, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 Valeur, 83, 85 Valeur, 84 manche general.c, 43 general.c, 43 general.c, 43 general.c, 43 general.c, 91 tableau.c, 91 tableau.c, 91 tableau.c, 91 tableau.c, 91 tableau.c, 92 tableau.h, 98 MODE_I_MANCHE main.h, 79 modifiefailleFenetre affichage.c, 16 affichage.c, 16 affichage.d, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MDE_CARRACTERE_LEADERBOARD NORMAL main.h, 84 NORMAL main.h, 84 NORMAL main.h, 83 nouvellePartie general.c, 44 general.h, 50 Vrice of the second sec		-
ROI, 85 SANS_ATOUT, 83 SANS_COULEUR, 83 SANS_COULEUR, 84 SANS_JOUEUR, 84 SANS_MESSAGE, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.c, 91 tableau.c, 91 tableau.h, 97 meulPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modificalialleFenetre affichage.c, 16 affichage.l, 21 moyenneTableau tableau.c, 92 tableau.h, 98 NB_CARRACTERE_LEADERBOARD main.h, 81 novellePartie general.c, 44 general.c, 44 general.c, 92 tableau.n, 98 main.h, 84 OUEST main.h, 84 OUEST main.h, 84 OUEST main.h, 84 POUE main.h, 83 ploue main.h, 83 ploue main.h, 83 ploue main.h, 83 POSE_CARTE main.h, 85 poseCarte general.c, 46 general.c, 46 general.n, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX proposeContrat		
SANS_ATOUT, 83 SANS_COULEUR, 84 SANS_JOUEUR, 84 SANS_JOUEUR, 84 SANS_MESSAGE, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.c, 43 general.c, 43 general.c, 49 maxi tableau.c, 91 tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.c, 44 general.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifie TailleFenetre affichage.c, 16 affichage.c, 16 affichage.c, 16 affichage.c, 16 affichage.c, 16 affichage.d, 21 moyenneTableau tableau.c, 92 tableau.h, 98 NORMAL main.h, 83 nouvellePartie general.c, 44 general.h, 50 rodonnerTableau tableau.e, 99 tableau tableau.e, 99 OUEST main.h, 84 OUEST main.h, 84 POUE main.h, 83 PIOUE main.h, 83 PIOUE main.h, 83 PIOUE main.h, 83 pointPil autre.c, 24 autre.c, 24 autre.h, 28 POSE_CARTE main.h, 85 poseCarte general.c, 46 general.c, 46 general.h, 50 POSITION_NB_MANCHES_POUR_GAGNER main.h, 81 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81	- ·	
SANS_JOULEUR, 83 SANS_JOULEUR, 84 SANS_MESSAGE, 85 SANS_VALEUR, 85 SANS_VALEUR, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 Valeur, 83, 85 Valeur, 83, 85 Valeur, 84 manche general.c, 43 general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modificialileFenetre affichage.c, 16 affichage.c, 16 affichage.d, 21 moyenneTableau tableau.c, 92 tableau.h, 98 NB_CARRACTERE_LEADERBOARD main.h, 81 preneur Contrat, TPOPOSEOCONTAI main.h, 81 preneur Contrat, TPOPOSEOCONTAI		•
SANS_JOUEUR, 84 SANS_MESSAGE, 85 SANS_VALEUR, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.c, 91 tableau.c, 91 tableau.c, 91 general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modificTailleFenetre affichage.c, 16 affichage.c, 16 affichage.c, 16 affichage.uc, 92 tableau.u, 98 modelerative general genera	-	_
SANS_MESSAGE, 85 SANS_VALEUR, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_MESSAGE, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.c, 16 affichage.c, 16 affichage.c, 92 tableau.c, 92 tableau.d, 98 MDE_CARRACTERE_LEADERBOARD SURCOINCHE, 85 SURCOINCHE, 83 TordonnerTableau tableau.c, 94 tableau.c, 95 tableau.c, 96 mini tableau.c, 97 modifieTailleFenetre affichage.c, 16 affichage.c, 16 affichage.c, 16 affichage.c, 92 tableau.c, 93 tableau		
SANS_VALEUR, 85 SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_MESSAGE, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modific Taille Fenetre affichage.c, 16 affichage.c, 16 affichage.c, 16 affichage.d, 21 tableau.c, 92 tableau.h, 98 MD_CARRACTERE_LEADERBOARD general.c, 80 rdonnerTableau tableau.c, 92 tableau.h, 98 TREFLE, 83 TypeMessage, 82, 84 OUEST main.h, 84 OUEST main.h, 84 OUEST main.h, 84 POUE main.h, 83 PIQUE main.h, 83 POSE_CARTE main.h, 85 poseCarte general.c, 46 general.c, 46 general.h, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 POPOSECOntrat POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_TOURE TOUT Contrat, 7 ProposeContrat		
SEPT, 85 SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 Valeur, 83, 85 Valeur, 84 Manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.c, 16 affichage.c, 16 affichage.c, 92 tableau.h, 98 NB_CARRACTERE_LEADERBOARD OUEST main.h, 84 POUE main.h, 83 PIQUE main.h, 83 pli general.c, 45 general.h, 51 pointPli autre.c, 24 autre.h, 28 POSE_CARTE main.h, 85 poseCarte general.h, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 ProposeContrat		_
SUD, 84 SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifie TailleFenette affichage.c, 16 affichage.c, 16 affichage.c, 12 moyenne Tableau tableau.c, 92 tableau.c, 92 tableau.c, 92 tableau.c, 92 tableau.c, 96 modifie TailleFenette affichage.c, 16 affichage.c, 16 affichage.c, 16 affichage.c, 22 tableau.c, 92 tableau.c, 92 tableau.c, 92 tableau.c, 92 tableau.d, 98 MODE_1_MANCHE main.h, 81 POSITION_NB_MANCHES_POUR_GAGNER main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_TOPOSECONITET TOPOSECONITET TOPOTITED TOPOSECONITET TOPOTITED TOPOSECONITET TOPOTITED TOPOSECONITET TOPOTITED TOPOSECONITET TOPOTITED TOPOSECONITET TOPOSECONITET TOPOTITED TOPOSECONITET TOPOTITED TOPOSECONITET TOPOTITED TOPOSECONITET TOPOTITED TOPOSECONITET TOPOTITED TOPOTITED TOPOSECONITET TOPOTITED		general.h, 50
SURCOINCHE, 83 TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_PESEUDO, 81 TAILLE_MAXI_MESSAGE, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 Valeur, 83, 85 Valeur, 83, 85 VALET, 85 Valeur, 83, 85 Valeur, 83, 85 Valeur, 83, 85 Valeur, 84 Manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.c, 91 tableau.h, 97 POSE_CARTE main.h, 85 poseCarte general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifie TailleFenetre affichage.c, 16 affichage.c, 16 affichage.h, 21 moyenne Tableau tableau.c, 92 tableau.h, 98 TOUEST main.h, 84 POSITION_NB_MANCHES_POUR_GAGNER main.h, 81 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_TOPOSECONTRAT NB_CARRACTERE_LEADERBOARD		
TAILLE_MAXI_COULEUR, 81 TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_PESEUDO, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifie TailleFenetre affichage.c, 16 affichage.c, 16 affichage.c, 22 tableau.h, 98 Tout_Atableau Tableau.c, 92 tableau.c, 92 tableau.d, 98 modifie TailleFenetre affichage.c, 16 affichage.c, 16 affichage.c, 22 tableau.h, 98 Tout_Atableau Tableau.c, 92 tableau.h, 98 Tout_Atableau Tableau.c, 92 tableau.h, 98 Tout_Atableau Tableau Table		
TAILLE_MAXI_MESSAGE, 81 TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifie TailleFenetre affichage.c, 16 affichage.c, 16 affichage.d, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MD_CARRACTERE_LEADERBOARD OUEST main.h, 84 OUEST main.h, 84 Parametre sous-menus.c, 86 s	SURCOINCHE, 83	
TAILLE_MAXI_PESEUDO, 81 TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 preneur Contrat, 7 proposeContrat	TAILLE_MAXI_COULEUR, 81	
TOUT_ATOUT, 83 TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.d, 98 moyenneTableau tableau.c, 92 tableau.h, 98 position_RECORD_VICTOIRE main.h, 81 position_RECORD_VICTOIRE main.h, 81 position_RECORD_VICTOIRE main.h, 81 preneur Contrat, 7 proposeContrat	TAILLE_MAXI_MESSAGE, 81	
TREFLE, 83 TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 Valeur, 83, 85 Valeur, 83, 85 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifietTailleFenetre affichage.c, 16 affichage.c, 16 affichage.d, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MCDE_1_MANCHE main.h, 79 modifietTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 POSCONTRAT POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 proposeContrat	TAILLE_MAXI_PESEUDO, 81	main.h, 84
TypeMessage, 82, 84 VALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.c, 92 tableau.c, 92 tableau.c, 92 tableau.c, 92 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.c, 92 tableau.h, 98 MCDE_ARRACTERE_LEADERBOARD sous-menus.c, 86 sous-menus.h, 88 PIQUE main.h, 83 peneral.c, 45 general.c, 24 autre.c, 24 autre.c, 24 autre.c, 24 autre.c, 24 autre.c, 24 autre.h, 28 POSE_CARTE main.h, 85 poseCarte general.h, 51 poseCarte peneral.h, 52 poseCarte main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 proposeContrat	TOUT_ATOUT, 83	
yALET, 85 Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MD_CARRACTERE_LEADERBOARD sous-menus.h, 88 PIQUE main.h, 83 pli general.c, 45 general.c, 45 general.h, 51 poseCarte main.h, 85 poseCarte general.c, 46 general.h, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 proposeContrat	TREFLE, 83	•
Valeur, 83, 85 ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.c, 16 affichage.c, 16 affichage.c, 16 affichage.c, 16 affichage.c, 12 tableau.c, 92 tableau.c, 92 tableau.c, 92 tableau.d, 98 movernetableau tableau.c, 92 tableau.d, 98 movernetableau tableau.c, 92 movernetableau tableau.c, 92 movernetableau tableau.c, 92 tableau.h, 98 movernetableau tableau.c, 92 tableau.h, 98 movernetableau tableau.c, 92 tableau.h, 98 poseCarte general.c, 46 general.c, 46 general.h, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD	TypeMessage, 82, 84	
ZERO, 84 manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 main.h, 81 pointPli autre.c, 24 autre.h, 28 POSE_CARTE main.h, 85 poseCarte general.c, 46 general.c, 46 general.h, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD	VALET, 85	
manche general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.h, 50 mini tableau.c, 92 tableau.h, 98 modifie TailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 maxi general.c, 45 general.h, 51 pointPli autre.c, 24 autre.h, 28 POSE_CARTE main.h, 85 poseCarte general.c, 46 general.c, 46 general.h, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX preneur Contrat, 7 ProposeContrat	Valeur, 83, 85	
general.c, 43 general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 peneral.c, 46 general.h, 50 poseCarte general.c, 46 general.h, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX preneur Contrat, 7 PNB_CARRACTERE_LEADERBOARD	ZERO, 84	
general.h, 49 maxi tableau.c, 91 tableau.h, 97 menuPrincipal general.h, 50 poseCarte general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_T_CARTE main.h, 80 POSE_CARTE main.h, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 80 POSITION_NB_WICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD	manche	pli
maxi pointPli tableau.c, 91 autre.c, 24 tableau.h, 97 POSE_CARTE menuPrincipal main.h, 85 general.c, 44 general.h, 50 poseCarte mini general.c, 46 tableau.c, 92 tableau.h, 98 POSITION_NB_MANCHES_POUR_GAGNER main.h, 79 POSITION_NB_VICTOIRE main.h, 79 POSITION_RECORD_VICTOIRE affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 positive. ### POSITION_RECORD_VICTOIRE main.h, 81 ### POSITION_SCORE_MAX main.h, 81 ### POSITION_SCORE_MAX main.h, 81 ### POSITION_SCORE_MAX main.h, 81 ### POSITION_SCORE_MAX ### POSITION_SCORE_MAX ### TopoposeContrat ### POSITION_SCORE_MAX ### POSITION_SCORE_MAX ### POSITION_SCORE_MAX ### POSITION_SCORE_MAX ### POSITION_SCORE_MAX ### POSITION_SCORE_MAX ### PO	general.c, 43	-
maxi pointPli tableau.c, 91 tableau.h, 97 menuPrincipal POSE_CARTE general.c, 44 general.h, 50 mini general.h, 52 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 poseCarte general.c, 46 general.h, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD	general.h, 49	general.h, 51
tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_Tableau.c, 92 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 poseCarte general.c, 46 general.h, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD		pointPli
tableau.h, 97 menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 menuPrincipal poseCaRTE main.h, 85 poseCarte general.c, 46 general.h, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD	tableau.c. 91	autre.c, 24
menuPrincipal general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 POSITION_NB_WANCHES_POUR_GAGNER main.h, 81 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD		autre.h, 28
general.c, 44 general.h, 50 mini tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 Manch, 85 poseCarte general.h, 82 POSITION_NB_MANCHES_POUR_GAGNER main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD		POSE_CARTE
general.h, 50 mini general.c, 46	•	main.h, 85
mini general.c, 46 tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD		poseCarte
tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 general.h, 52 POSITION_NB_MANCHES_POUR_GAGNER main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD		general.c, 46
tableau.h, 98 MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHES_POUR_GAGNER main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD		general.h, 52
MODE_1_MANCHE main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MODE_1_MANCHE main.h, 80 POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD		-
main.h, 79 modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 MB_CARRACTERE_LEADERBOARD POSITION_NB_VICTOIRE main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 ProposeContrat		
modifieTailleFenetre affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 NB_CARRACTERE_LEADERBOARD main.h, 81 POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 proposeContrat		
affichage.c, 16 affichage.h, 21 moyenneTableau tableau.c, 92 tableau.h, 98 NB_CARRACTERE_LEADERBOARD POSITION_RECORD_VICTOIRE main.h, 81 POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 proposeContrat		
affichage.h, 21 main.h, 81 moyenneTableau POSITION_SCORE_MAX tableau.c, 92 main.h, 81 tableau.h, 98 preneur		
moyenneTableau tableau.c, 92 tableau.h, 98 NB_CARRACTERE_LEADERBOARD POSITION_SCORE_MAX main.h, 81 preneur Contrat, 7 proposeContrat		
tableau.c, 92 main.h, 81 tableau.h, 98 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD proposeContrat		•
tableau.h, 98 preneur Contrat, 7 NB_CARRACTERE_LEADERBOARD proposeContrat	•	
Contrat, 7 NB_CARRACTERE_LEADERBOARD proposeContrat		
NB_CARRACTERE_LEADERBOARD proposeContrat	tableau.h, 98	•
	NR CARRACTERE LEADERROADD	
general.c, 40	-	
	mainin, oo	general.c, 📆

general.h, 52	sous-menus.c, 85
proposeContratla	leaderboard, 86
ia.c, 74	parametre, 86
ia.h, 76	statistiqueJoueur, 86
proposeContratUtilisateur	sous-menus.h, 87
affichage.c, 16	leaderboard, 87
affichage.h, 21	parametre, 88
	statistiqueJoueur, 88
QUATRE_VINGT	statistiqueJoueur
main.h, 84	sous-menus.c, 86
QUATRE_VINGT_DIX	sous-menus.h, 88
main.h, 84	stockeInfoCarte
	formatageChaine.c, 34
rechercheAnnonce	formatageChaine.h, 40
gestionCarte.c, 56	SUD
gestionCarte.h, 65	main.h, 84
rechercheDichotomie	supprimeCarte
tableau.c, 93	gestionCarte.c, 60
tableau.h, 99	gestionCarte.h, 67
rechercherCarte	SURCOINCHE
gestionCarte.c, 57	main.h, 83
gestionCarte.h, 65	,
rechercherCarteSuperieur	tableau.c, 89
gestionCarte.c, 57	acquiertTableau, 89
gestionCarte.h, 66	afficheTableau, 90
rempliEspace	aleatoireTableau, 90
formatageChaine.c, 34	constanteTableau, 91
formatageChaine.h, 39	copie, 91
RESULTAT_MANCHE	maxi, 91
main.h, <mark>85</mark>	mini, 92
RESULTAT_PLI	moyenneTableau, 92
main.h, 85	ordonnerTableau, 93
ROI	rechercheDichotomie, 93
main.h, 85	sommeTableau, 94
	zeroSiSuperieur, 94
SANS ATOUT	tableau.h, 95
 main.h, <mark>83</mark>	acquiertTableau, 95
SANS_COULEUR	afficheTableau, 96
main.h, 83	aleatoireTableau, 96
SANS_JOUEUR	constanteTableau, 97
main.h, 84	copie, 97
SANS MESSAGE	maxi, 97
main.h, 85	mini, 98
SANS VALEUR	moyenneTableau, 98
main.h, 85	ordonnerTableau, 99
SEPT	rechercheDichotomie, 99
main.h, 85	sommeTableau, 100
setCarte	zeroSiSuperieur, 100
gestionCarte.c, 59	TAILLE MAXI COULEUR
gestionCarte.b, 66	main.h, 81
setContrat	
	TAILLE_MAXI_MESSAGE
autre.c, 25	main.h, 81
autre.h, 28	TAILLE_MAXI_PESEUDO
sommeForceCarte	main.h, 81
gestionCarte.c, 59	TOUT_ATOUT
gestionCarte.h, 67	main.h, 83
sommeTableau	TREFLE
tableau.c, 94	main.h, 83
tableau.h, 100	trieCarte

```
gestionCarte.c, 60
    gestionCarte.h, 68
TypeMessage
    main.h, 82, 84
vainqueurPli
    gestionCarte.c, 61
    gestionCarte.h, 68
VALET
    main.h, 85
Valeur
    main.h, 83, 85
valeur
    Carte, 5
ZERO
    main.h, 84
zeroSiSuperieur
    tableau.c, 94
    tableau.h, 100
```