

Life'Seven Poker

FLICOCAGU POKER

Life'Seven POKER

Les membres :

- DAVIS James
 - Programmation globale, débogage et graphisme
- DELYS Olivier

 - Débogage, algorithmique et optimisation
- GUILLOU Benjamin
 - Programmation, débogage et graphisme
- ROLLET Tristan
 - Cahier des charges, Doxygen et implémentations

Ce qui ne marche pas , ce qui n'a pas été fait :

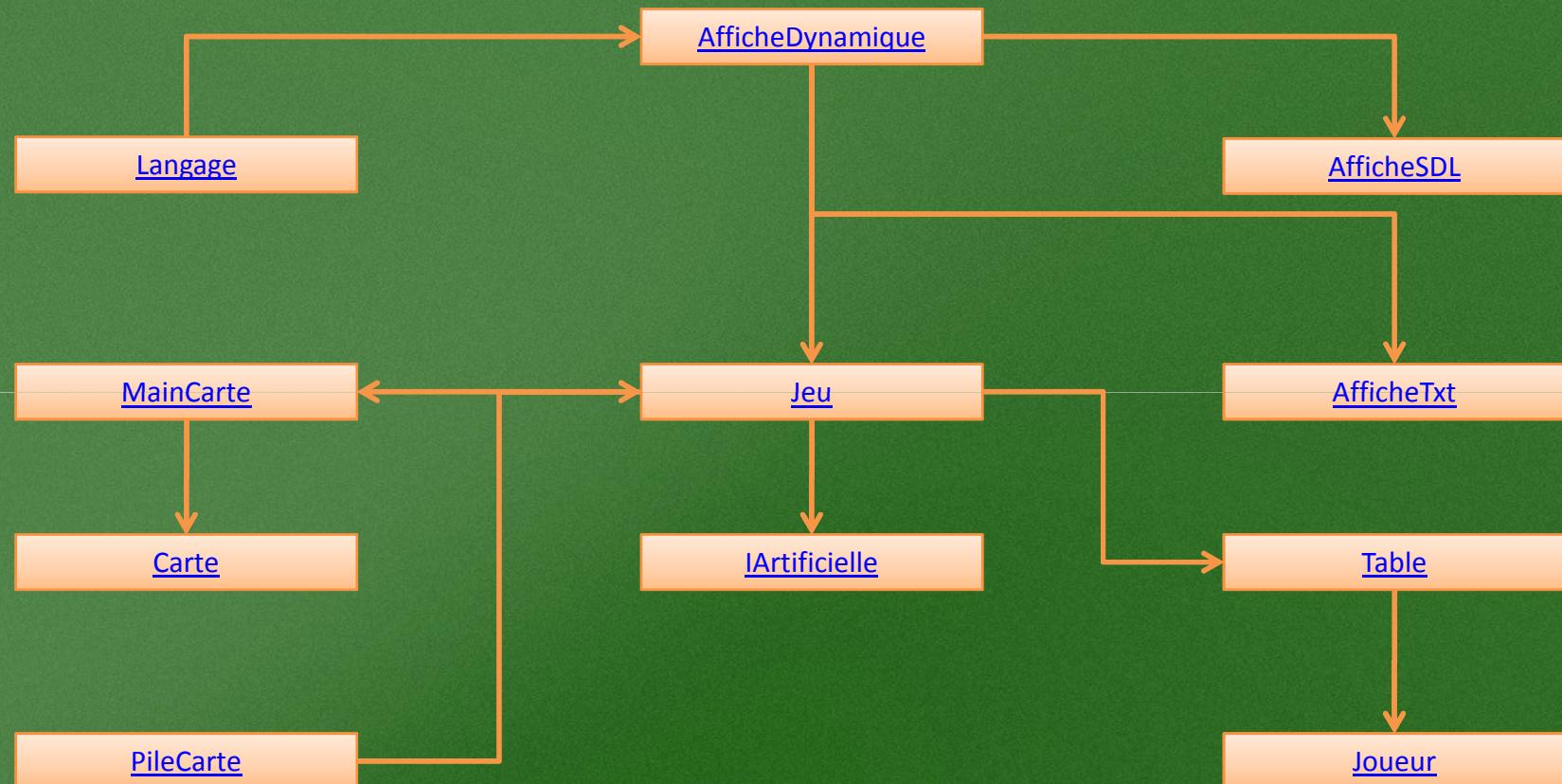
- L'IA performante
- Le réseau
- Quelques bugs dans le jeu

Ce qui marche , ce qui a été fait :

- Le jeu dans son ensemble
- Une IA basique
- La préparation à une implémentation réseau
- Interface ergonomique
- Jeu disponible en anglais et français
- Contrôles intuitifs
- Application multiplateformes
- Fichiers de configuration



Diagramme des modules :



Module AfficheDynamique

- void affAffichageVainqueur(SDL_Surface* affichage, Table & t, const char langue[][50], bool afficheCarteGagnant);
- void calculVainqueurTapis(Table & t, int tabResultat[][6][2]);
- SDL_Rect AffCenterer(SDL_Surface* source, SDL_Surface* destination, int option);
- void AffStartUp(SDL_Surface* affichage, SDL_Surface* tapis);
- void AffAffichePot(SDL_Surface* affichage, const Table & t, const char langue[][50]);
- int AffMenu(SDL_Surface* affichage, const char langue[][50]);
- void AffAfficheJoueur(SDL_Surface* affichage, const Joueur & j, const Table & table, int joueurJouant, const char langue[][50]);
- void AffAffichageInfosJoueurs(SDL_Surface* affichage, const Table & t, int joueurJouant, const char langue[][50]);
- void AffCartesJoueursJeu(SDL_Surface* affichage, const Table & t);
- void AffCartesJoueursJeu(SDL_Surface* affichage, const Table & t, bool cache);
- void AffCarteDecouvertes(const Table & t, SDL_Surface* affichage);
- void AffCarteDecouvertes(const Table & t, SDL_Surface* affichage, bool evidence, int tabResultat[6][2]);
- void AffCartesJoueursJeuFinale(SDL_Surface* affichage, const Table & t, int tabResultat[6][2], int i);
- void AffInfosJoueur(SDL_Surface* affichage, const Joueur & j, const Table & table, const char langue[][50]);
- void AffAfficheBoutonRelance(SDL_Surface* affichage, int relance, bool dessus, const char langue[][50]);
- int scanActionJoueur(SDL_Surface* affichage, int & relance, Statut & s, int & montant, const Joueur & j, const Table & t, const char langue[][50]);
- void miseDansPot(Table & t);
- void calculGainTapisJoueur(Table & t);
- int lancePartie(SDL_Surface* affichage, SDL_Surface* tapis, const char langue[][50], int NOMBRE_JOUEUR_PC, int argentDepart);



Module AfficheSdl

- void pause();
- bool initSDL(SDL_Surface* & screen, const int & screen_width, const int & screen_height, const int & screen_bpp, const char* caption);
- SDL_Surface *load_image(const char* filename);
- void apply_surface(int x, int y, SDL_Surface* source, SDL_Surface* destination);
- void affichageOmbreCarte(SDL_Surface* affichage,int x,int y,double zoom);
- void AffAfficheCarte(SDL_Surface* affichage, Carte* c, int x, int y, double zoom);
- void AffAfficheCarte(SDL_Surface* affichage, Carte* c, int x, int y, double zoom,bool evidence);
- void AffAfficheTexte(SDL_Surface* destination,const char* message,int x,int y,int r,int g,int b,int style,int size);
- void AffAfficheTexte(SDL_Surface* destination,const char* message,int x,int y,int r,int g,int b);
- void AffActualiser(SDL_Surface* affichage);
- void AffAfficheTapis(SDL_Surface* aff,SDL_Surface* tapis);



Module AfficheTxt

- void afficheTxtGainTapis(const Table & t,const char* titre);
- void afficheTab2(int tab[][2],int x);
- void afficheTab3(int tab[][6][2],int x);
- void afficheTab(int tab[],int x);
- void afficheInfoJoueur(const Joueur & j);
- void afficheInfoTable(const Table & t);
- void afficheMainCarte(const MainCarte & m,char titre[]);



Module Carte

- void initialisationCarte(Carte & c);
- int getCarteRang(const Carte & c);
- int getCarteCouleur(const Carte & c);
- void setCarte(Carte & c,const int & couleur,const int & rang);
- int compareCarte(const Carte & c1, const Carte & c2);
- void getCarteNomFichier(const Carte & c, char* sortieNomFichier);



Module IArtificielle

- void calculIA(const Table & t,const Joueur & j,int montant,int &relance);
- void definieStatut(const Table & t,Statut & s,const Joueur & j,int montant,int & relance);
- Main determineMeilleureMainIA(const MainCarte & mainJoueur,const MainCarte & cartesDecouvertes);
- float probaActionIA(const Table & table, const Joueur & joueur,Main meilleureMainJoueur);



Module Jeu

- bool compareTabResultat(const int tabResultat1[6][2],const int tabResultat2[6][2]);
- int atendsActionJoueur(SDL_Surface* aff,const Table & t,const Joueur & j,int & relance,Statut & s,int & montant,const char langue[][50]);
- void joueurPetiteBlind(Table & t,Joueur & j);
- void joueurGrosseBlind(Table & t,Joueur & j);
- void distribuer2CartesJoueursJeu(Table & table);
- void distribuer1CarteDecouverteJeu(Table & table,int n);
- void initialiseTab3d(int tabResultat[10][6][2]);
- void jeuDetermineVainqueur(const Table & t,int tabResultat[10][6][2]);
- int trieTab3d(int tabResultat[10][6][2]);
- int fonctionGlobaleDeterminationVainqueur(const Table & t,int tabResultat[10][6][2]);
- void trieTableauRang(int tab7Carte[7][2]);
- int codageScoreMain(const MainCarte &m, int tabResultat[6][2],const Table & table);



Module Joueur

- | | |
|--|--|
| <ul style="list-style-type: none">• int getTapisJoueur(const Joueur & j);• void setTapisJoueur(Joueur & j,int n);• int getGainTapisJoueur(const Joueur & j);• void setGainTapisJoueur(Joueur & j,int n);• void ajouteGainTapisJoueur(Joueur & j,int n);• void initJoueur(Joueur & joueur);• int getPositionJoueurX(const Joueur & j);• int getPositionJoueurY(const Joueur & j);• void setPositionJoueurX(Joueur & j,int x);• void setPositionJoueurY(Joueur & j,int y);• void initJoueur (Joueur & joueur,const char* pseudo);• Joueur* creeJoueur();• void setStatutJoueur (Joueur & joueur, const Statut statut);• Statut getStatutJoueur (const Joueur & joueur);• void setTypeJoueur(Joueur & joueur, const TypeJoueur typejoueur); | <ul style="list-style-type: none">• TypeJoueur getTypeJoueur(const Joueur & joueur);• void joueurLibere(Joueur & joueur);• void joueurDetruit(Joueur* & joueur);• void reinitialiseMainJoueur(Joueur & joueur);• void setIdJoueur(Joueur & joueur, int n);• int getIdJoueur(const Joueur & joueur);• void setMiseJoueur(Joueur & joueur, const int & n);• int getMiseJoueur(const Joueur & joueur);• int ajoutMiseJoueur(Joueur & joueur,int n);• MainCarte* getMainJoueur(const Joueur & joueur);• void setArgentJoueur(Joueur & joueur, int n);• int getArgentJoueur(const Joueur & joueur);• void ajoutArgentJoueur(Joueur & joueur, int n);• void reinitialisationMainJoueur(Joueur & j);• void actionJoueur(Joueur & j,Statut s,int & montant,int relance); |
|--|--|



Module Langage

- int ecrireDansTableau(char menu[50][50],const char nomFichier[50]);
- void afficherMenu(const char c[50][50],int a);



Module MainCarte

- void reinitialisationMain(MainCarte & m);
- int compareMain(int tabResultat1[6][2],int tabResultat2[6][2]);
- void initialisationMain(MainCarte & m);
- int getMainCarteNbCarte(const MainCarte & m);
- Carte* getMainCarteLeMeilleurCarte(const MainCarte & m, int i);
- void ajouteCarte(MainCarte & m, Carte* c);
- void MainCarteLibere(MainCarte & m);
- int couleurMainCarte(int tab7Carte[8][2]);
- int suiteMainCarte(int tab7Carte[8][2]);
- void nombreOccurrenceCarte(const int tab7Carte[10][2],int tab[]);
- int quinteFlushMainCarte(int tab7Carte[8][2],int couleur);
- void choixCarteMultiple(int tab7Carte[][2],const int tabO[],int tabResultat[][2]);
- void afficheMainCarte(const MainCarte & m,const char* titre);



Module PileCarte

- void initPileCarte (PileCarte & pileCarte);
- PileCarte* creePileCarte();
- void pileCarteLibere (PileCarte & pileCarte);
- void pileCarteDetruit(PileCarte* & pileCarte);
- Carte* tirerCarte(PileCarte & pileCarte);
- void nouvellePileCarte(PileCarte & pileCarte);
- int hasard(int x,int y);



Module Table

- int getTablePot(const Table & t);
- void setTablePot(Table & t,int n);
- void reinitialisationMainJoueurTable(Table & t);
- void initTable(Table & table);
- Table* creeTable();
- void tableLibere(Table & table);
- void tableDetruit(Table* & table);
- void ajoutJoueurTable (Table & table, Joueur* joueur);
- void supprimeJoueurTable (Table & table, Joueur* joueur);
- void setMaxJoueurTable (Table & table, int n);
- int getMaxJoueurTable (const Table & table);
- PileCarte* getTablePileCarteTable (const Table & table);
- void setNJoueurTable (Table & table, int n);
- int getNJoueurTable (const Table & table);
- int placeVide (const Table & table);
- void changeDealerTable(Table & table);
- Joueur* getlemeJoueur(const Table & table,int n);
- MainCarte* getMainCarteTable(const Table & table);
- void setPetiteBlindTable (Table & table, int n);
- int getPetiteBlindTable (const Table & table);
- int getJoueurSuivant(const Table & table,int i);
- int getPositionDealerTable(const Table & table);
- void afficheInfoTable(const Table & t);



Un design sobre et soigné :



Life'Seven POKER

L'intégralité des images a été réalisée par nos soins, sauf pour les cartes qui ont été récupérées sur Wikimedia et modifiées.

Manuel du jeu :

- <*m*> pour le menu , <*Echap*> pour en sortir
- <*Echap*> en cours de partie pour quitter le jeu
- <*Clic gauche*> pour poursuivre le déroulement de la partie à la fin d'une manche
- <*Clic gauche*> sur credits dans le menu : affiche les crédits dans la console

Le fichier de configuration et l'interface dynamique

- Grâce au fichier de configuration, on peut facilement personnaliser l'interface :
 - 1^{ere} ligne : ‘f’ pour français, ‘e’ pour anglais
 - 2^e ligne : nombre de joueur en plus de 1 à 7
 - 3^e ligne : argent de départ
 - 4^e ligne : nom du joueur local

L'adaptation à plusieurs langages :

Création d'un fichier TXT comprenant tous les mots utilisés dans le jeu

Création d'un tableau de chaines de caractères initialisé avec le fichier TXT

Utilisation du tableau pour l'affichage des mots

Jeu facilement adaptable à toutes les langues du monde

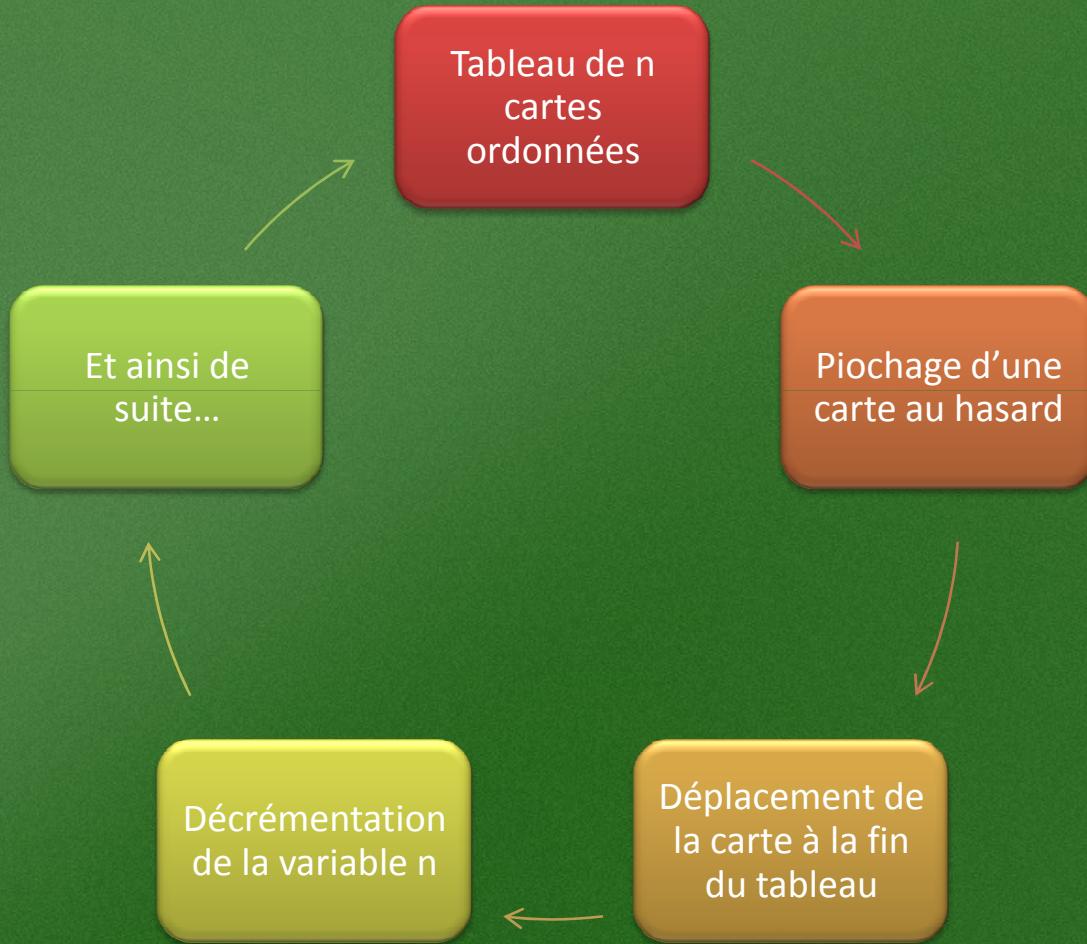
Une partie... in English :



Life'Seven POKER

La gestion dynamique des langues permet à n'importe quelle personne comprenant l'anglais ou le français de le traduire dans une autre langue.

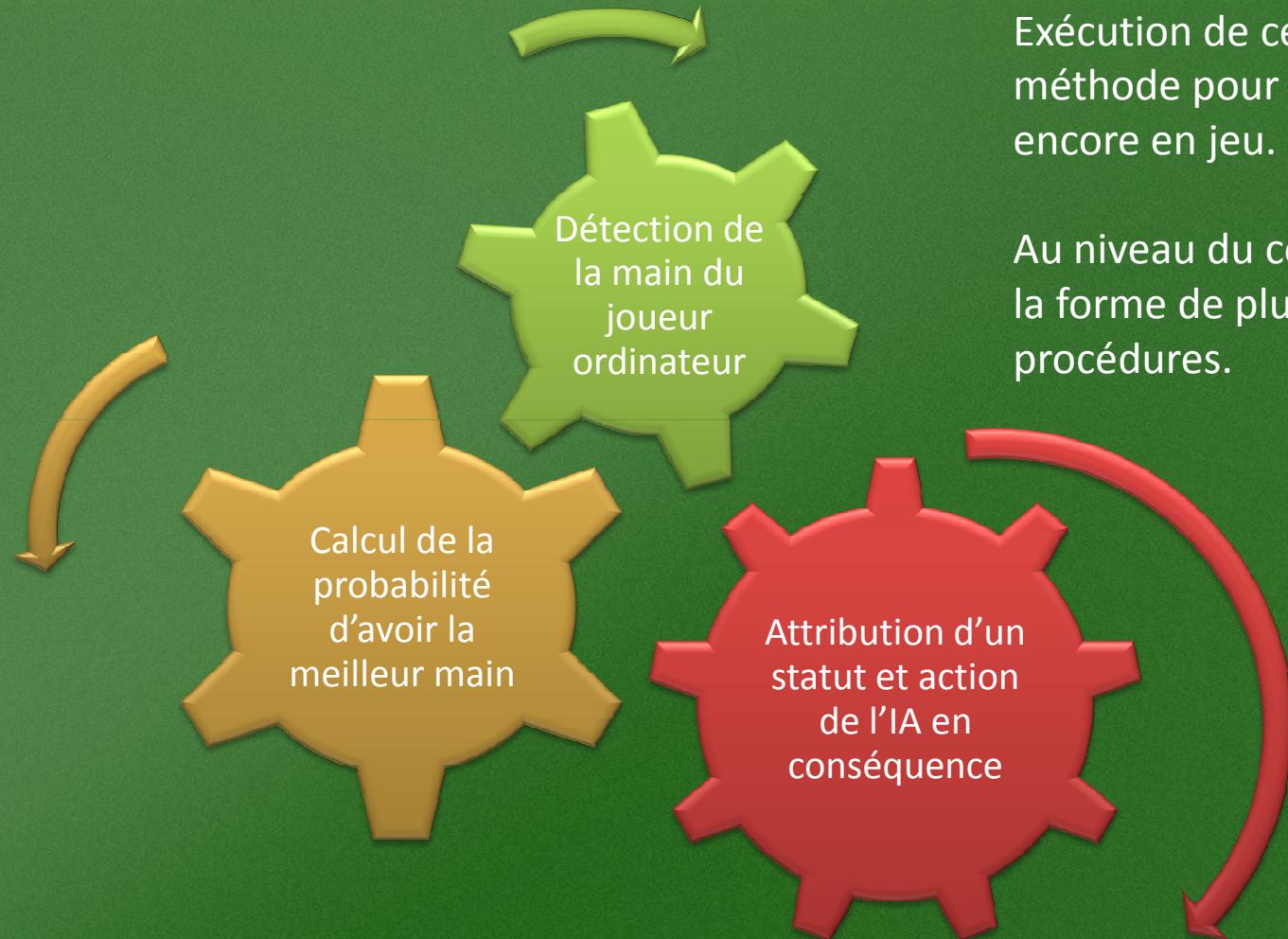
Mélange des cartes :



L'identification du vainqueur et sa main gagnante :



L'intelligence Artificielle Basique



Exécution de cette méthode pour toutes les IA encore en jeu.

Au niveau du code , prend la forme de plusieurs procédures.

Ce que l'on a appris :

- Utiliser Code::blocks, SVN, Valgrind et Doxygen
- Ecrire un cahier des charges et tout faire pour le respecter.
- Structurer le code
- Gérer le temps et le travail en équipe
- Programmer un affichage graphique en C
- Utiliser un débuggeur tel que gdb intégré à Code::blocks

Si c'était à refaire qu'est ce que l'on changerait ?

- Un code plus organisé
- La répartition du travail dans le temps : un travail plus régulier