***README***

Présentation :

CardYard est un jeu de cartes stratégique jouable en local via un terminal. Chaque joueur dispose de cartes personnelles disposées face cachée devant lui et doit tenter d’obtenir le score le plus bas en fin de partie. Les joueurs piochent et échangent des cartes à tour de rôle en utilisant une pioche commune et des défausses individuelles.

Règles du jeu :

- Chaque joueur commence avec un nombre de cartes étalées faces cachées.

- Les valeurs des cartes vont de -2 à +12 :

- 5 cartes de valeur -2

- 10 cartes de valeur -1

- 15 cartes de valeur 0

- 10 cartes pour chaque valeur positive (de 1 à 12)

- À tour de rôle, les joueurs :

1. Piochent une carte depuis la pioche centrale ou n’importe quelle défausse.

2. Échangent cette carte avec l’une de leurs cartes étalées (visible ou non).

3. La carte échangée est placée face visible sur leur défausse.

- La partie se termine lorsqu’un joueur a retourné toutes ses cartes étalées.

- Chaque joueur additionne les valeurs de ses cartes visibles. Le joueur avec le score le plus bas remporte la partie.

Installation :

Pour pouvoir jouer au jeu il faut avoir un compilateur C.

Téléchargement du projet :

Cloner le dépôt :

git clone (<https://github.com/adm95560>)

Se rendre dans le dossier du projet :

cd <https://github.com/adm95560/Card_Yard>

Puis se placer dans la branche main.c :

git checkout <https://github.com/adm95560/Card_Yard/tree/main.c>

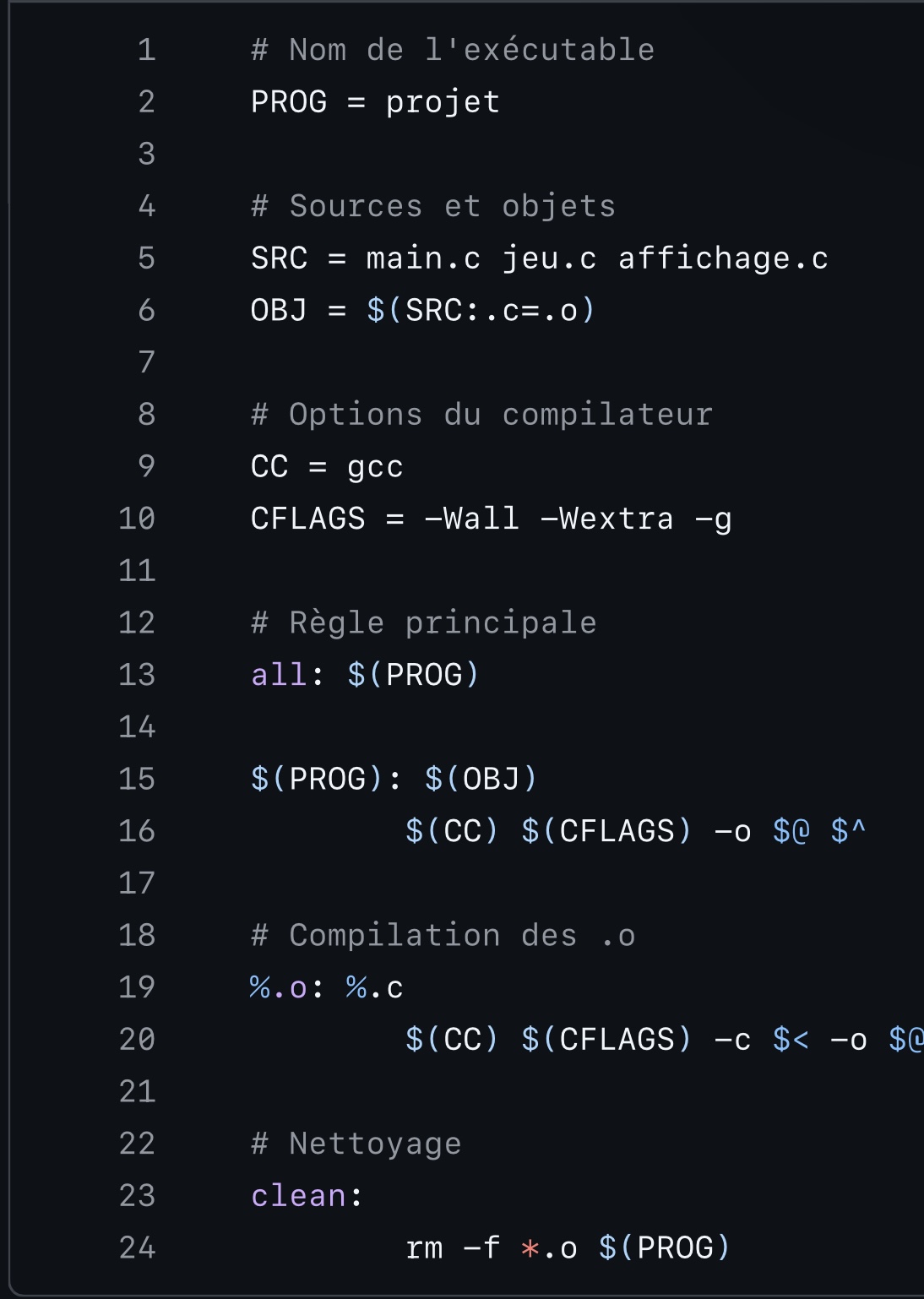
Lancement du jeu :

Utiliser le Makefile pour compiler et lancer le jeu :

<https://github.com/adm95560/Card_Yard/blob/main.c/makefile>

Dans ce projet, le fichier Makefile automatise le processus de compilation et de gestion des fichiers générés. Il permet de compiler facilement les fichiers sources du projet, de nettoyer les fichiers intermédiaires et de générer l’exécutable sans avoir à écrire manuellement les commandes dans le terminal.

**Composition du Makefile**



**Nom de l'exécutable**

PROG = projet  
Le programme compilé sera nommé 'projet'.

**Liste des fichiers sources et objets**

SRC = main.c jeu.c affichage.c  
OBJ = $(SRC:.c=.o)  
Cette ligne liste les fichiers .c à compiler et génère la liste des fichiers .o correspondants.

**Paramètres du compilateur**

CC = gcc  
CFLAGS = -Wall -Wextra -g  
- -Wall : active les avertissements importants.  
- -Wextra : active des avertissements supplémentaires.  
- -g : ajoute des informations utiles pour le débogage.

**Compilation de l’exécutable**

$(PROG): $(OBJ)  
   $(CC) $(CFLAGS) -o $@ $^  
Crée l’exécutable à partir des fichiers objets.

**Compilation des fichiers objets**

%.o: %.c  
   $(CC) $(CFLAGS) -c $< -o $@  
Compile chaque fichier source .c en fichier objet .o.

**Nettoyage des fichiers**

clean:  
   rm -f \*.o $(PROG)  
Supprime les fichiers objets et l’exécutable généré.

**Comment éxécuté le programme ?**

make clean

make

./projet

Contenu du projet

Le projet est organisé comme suit :

- Affichage : programmes responsables de l’interface.

- Gestion du jeu : programmes liés à la mécanique de jeu.

- Makefile : facilite la compilation et le lancement du projet.

-Rapport PDF : explique la répartition des tâches au sein du groupe.

Auteur

- Adam

-Mehdi

-Rayane

(<https://github.com/adm95560>)