Vingt et un

version 2.0.0

DIVERSITY by EPITECH

I. Introduction

Le langage **C** est un des premiers langages de programmation accessible gratuitement. Il a été créé pour permettre d'écrire des programmes compilés plus facilement. Aujourd'hui, il est encore souvent utilisé pour écrire des programmes pour des logiciels embarqués qui ont souvent des restrictions de mémoire ou puissance de calcul, car il permet un contrôle total sur la mémoire et puissance utilisée par le programme.

Le jeu du 21, aussi appelé « Black Jack », est un jeu créé en France au XVIIIe siècle. Il est dit que Marie Antoinette y jouait. Ce jeu est parfois joué en casino et c'est l'un des seuls jeux d'argent à avoir un gain supérieur à un, ça veut dire qu'on y gagne plus d'argent qu'on en perd en moyenne. Pour cette raison, beaucoup de casinos choisissent de ne pas le proposer.

Aidez Carty à créer ce jeu de carte !

Un jeu de cartes mal rangé

II. Consignes

* Sur ce projet, il vous sera demandé de choisir comme nom de repository : cc_21.
* N'oubliez pas de push régulièrement.
* En cas de question, pensez à demander de l'aide à votre voisin de droite. Puis de gauche. Demandez enfin à un Cobra (ceux-là ne mordent pas) si vous êtes toujours bloqué(e).
* Vous avez tout à fait le droit d'utiliser internet pour trouver des réponses ou pour vous renseigner.
* N'hésitez pas à faire des bonus et à ajouter des fonctionnalités lorsque votre projet sera terminé et validé.

Explication du code:

* Tout ce qui est écrit après un `//` ou entre `/* ... */` est un commentaire pour t'aider à comprendre.

III. Les règles du jeu

Carty va jouer au vingt-et-un avec un jeu de carte modifié. Il n'y a ni Roi ni Reine qui auraient une valeur trop élevée et il n'y a pas non plus les 7 et 8 pour éviter de tomber sur trop ou trop peu, par conséquent de rendre le jeu moins prévisible à la fin. Elle va utiliser 3 jeux de cartes mélangés soit 108 cartes ! C'est 12 fois chaque carte !

Les valeurs des cartes se calculent normalement (l'as vaut 1, le 2 vaut 2, etc.) sauf pour le valet qui vaut soit 11 soit 1 selon le choix du joueur. Lorsque 6 cartes sont tirées sans dépasser 21 le score du joueur est compté comme un 21.

Le jeu se joue contre un croupier et le but d'une partie est de dépasser le nombre du croupier pour récupérer sa mise.

Une partie commence par la distribution de 2 cartes. Le croupier tire un nombre entre 14 et 21. Ensuite le croupier et le Carty misera entre 5 et 10 selon leur main. Puis commencent les tours de Carty au cours desquels il doit se rapprocher au maximum de 21 mais sans le dépasser. A chaque tour, elle demandera une autre carte ou s'arrêtera à sa main actuelle. Si elle fait 21 le croupier double la mise de Carty. À condition que le croupier fait 21 il ramasse les mises directement.

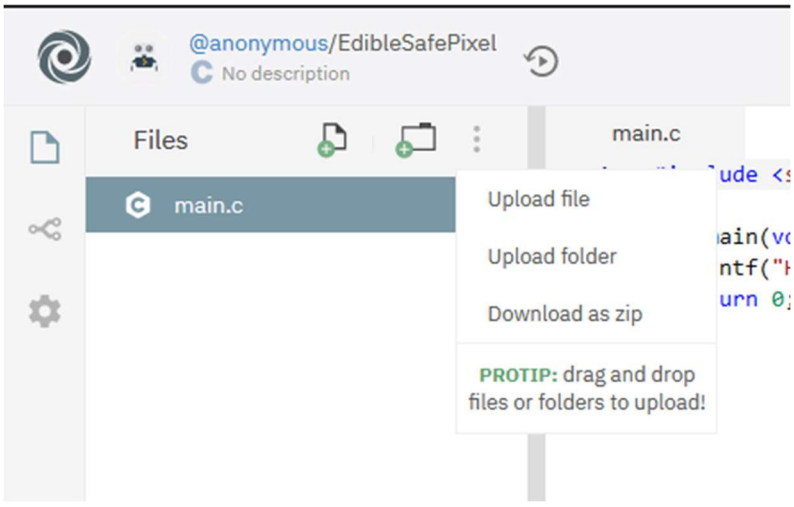
Dans le cas où le croupier a misé 8 et le joueur 7 et qu'elle fait 21, elle remporte 14 (soit le double de sa mise) en plus des 8 du croupier ! Au début de la partie Carty aura 15 pions à miser. Le jeu dure 5 parties et le joueur doit obtenir un maximum de points dans ces 5 parties.

IV. Coder un jeu en C

a. Préparation du jeu de cartes

Pour exécuter ce petit programme vous devez utiliser le site repl.it.

Dès que Carty arrive sur le site, il lui propose de créer un nouveau projet dans différents langages, sélectionnez l'option « Import From GitHub » et sélectionnez le repl que vous avez créé au début de la session. Allez dans le menu déroulant et sélectionnez « Upload folder ». Puis dans votre recherche de dossier sélectionnez le dossier « C21 » et appuyez sur envoyer.



L'interface de repl.it

Carty va maintenant ouvrir le fichier « main.c », elle y trouve ce code :

```
#include <stdio.h>
#include "game.h"

int main()
{
    t_game * g = game_init();

    game_loop(g);
    return 0;
}

void game_loop(t_game * g)
{

}
```

Les deux premières lignes servent à dire au programme que certaines fonctions existent hors de ce fichier. Par exemple, ici dans la fonction "main", délimitée par les accolades { }, on utilise la fonction game_init(). En C, "main" est une fonction qui sera appelée dès le lancement du programme. On ne touchera pas à cette fonction pendant cette session. Celle qui va intéresser Carty est "game_loop", juste en dessous.

b. A votre tour de jouer

Pour commencer essayons de faire fonctionner un simple tour de table :

- Distribution des cartes
- Etape de mise
- Calcul des scores

A chaque étape, on va aussi rajouter de l'affichage pour voir le déroulement du tour.

On va se servir des fonctions décrites ci-dessous :

```
print_player_coins(g->player); // Afficher le nombre de jetons du joueur
broker_new_hand(g); // Le croupier pioche
player_new_hand(g); // Le joueur pioce
print_player_hand(g->player); // Afficher la main du joueur
broker_place_bet(g); // Le croupier mise
player_place_bet(g); // Le joueur mise
sum_player_hand(g->player); /* Le joueur choisi la valeur de ses Valets (1
ou 11) et cette function "return" le score du joueur */
print_turn_results(g, p_hand); /* Afficher le score du croupier et du
joueur (le score du joueur doit être donné par la variable p_hand) */
```

Si Carty met ces quelques lignes dans la fonction "game_loop", entre les accolades, au moment de cliquer sur "run" vous verrez ce message apparaitre :

```
make
gcc -o main.o -c main.c -Wall -W -g
main.c: In function 'game_loop':
main.c:23:25: error: 'p_hand' undeclared (first use in this function)
   print_turn_results(g, p_hand); /* Afficher le score du croupier et du joueur
                         ^~~~~~
main.c:23:25: note: each undeclared identifier is reported only once for each function it
 appears in
Makefile:13: recipe for target 'main.o' failed
make: *** [main.o] Error 1
exit status 2
```

Message d'erreur lors de l'execution

Lorsque qu'elle appuye sur "run" elle n'est pas en train de lancer le programme mais de le "compiler". La compilation est une étape très importante de la programmation en C, elle sert à transformer le code que vous avez écrit en instructions que l'ordinateur est capable de lire et d'exécuter.

Pour le savoir ce qu'il s'est passé, Carty doit regarder cette partie de l'affichage : "*error: 'p_hand' undeclared (first use in this function)*". Le compilateur ne trouve pas une variable. Les variables sont des boîtes dans la mémoire de l'ordinateur dans lesquels on stocke des informations. En l'occurrence la fonction "`print_turn_results`" ne trouve pas la variable "`p_hand`".

Carty vas créer la variable "p_hand" et lui donner la valeur correspondant à son score. Son score est déjà présent dans ce code. Elle doit chercher la valeur de retour d'une fonction.

Si vous avez réussi, Carty à maintenant ce message quand elle appuye sur « run » :

```
make
gcc -o main.o -c main.c -Wall -W -g
gcc -o D21 main.o game.o
```

La compilation s'est bien passée

Elle peut même faire un tour rapide !

```
make
gcc -o main.o -c main.c -Wall -W -g
gcc -o D21 main.o game.o
./D21
You have 15 coins
Your hand : 4, 9
Broker bet : 8
Place your bet (5-10)
5
Your bet : 5
Broker score is : 17; your score is : 13
```

Première exécution du jeu

c. Remporter la manche

Carty va maintenant voir une autre notion de programmation : les conditions. Quand on veut qu'un programme agisse de façon différente dans certains cas on utilise des conditions. En C elles ressemblent à ça :

```
if (condition)
{
    ...
}
```

Elle a à disposition la variable « g->broker » qui correspond au score du croupier et ces 4 fonctions qui correspondent aux 4 résultats possibles d'une manche.

```
player_breakthrough(g); // Le joueur dépasse 21
player_lost(g); // Le joueur perd
player_jackpot(g); // Le joueur fait 21
player_win(g); // Le joueur gagne
```

En respectant les règles du jeu, Carty vas devoir détecter les différents cas de fin de manche avec les conditions d'exécution qui correspondent.

d. Tirer des cartes en plus

On va ensuite proposer au joueur de tirer plus de cartes pour se rapprocher de 21. La fonction à appeler pour ça est la suivante :

```
player_ask_cards(g);
```

La fonction « player_ask_cards » retourne en valeur le nombre de carte que le joueur a tiré. Attention, lorsque 6 cartes sont tirées sans dépasser 21 le score du joueur est compté comme un 21. Il faudra que Carty ajoute ce cas de figure dans son programme.

Le croupier qui distribue les cartes

e. Boucle de jeu

On a maintenant un tour de jeu complet ! Il ne reste plus qu'à Carty le répéter jusqu'à la fin de la partie. La fin arrive dans deux cas : soit le joueur a perdu tous ses jetons, soit il a joué 5 tours. Il y a plusieurs façons de faire des boucles en C et je vais vous laisser trouver celle que vous préférez utiliser. Pour cette partie vous pouvez utiliser ces fonctions :

```
is_game_finished(g) // Retourne vrai lorsque le jeu est passé par
game_end(g)
game_end(g); // change la valeur de retour de is_game_finished(g)
```

BlackJack, c'est gagné !

V. Conclusion

Et voilà ! Carty vas pouvoir se mettre au défi de faire le plus de score dans les 5 tours impartis ! Elle a maintenant un super jeu du vingt-et-un, il ne reste plus qu'à battre des records et améliorer le jeu aussi loin que possible !

Voici quelques exemples de bonus réalisables :

* Ajouter la possibilité de faire plusieurs parties.
* Ajouter un petit récapitulatif en fin de partie.
* Rajouter un deuxième joueur.
* Ajouter une gestion d'erreur pour les interaction avec du joueur
* Ajouter un mode en ligne, pour jouer avec des amis !