Rapport personnel : Nicecraps

Yann MARTIN D'ESCRIENNE
N°etu : 21713752

Le code source du jeu de Nicecraps se partage en 3 fichiers.
Tout d'abord le fichier *craps.h* qui met en place la structure player correspondant aux joueurs de la partie, un enumerate représentant les lignes de mise du jeu et enfin tous les prototypes des fonctions de *craps.c*.
Craps.c contient plusieurs fonctions servant au bon fonctionnement du jeu. On peut les diviser en catégories : celles servant pour le highscore, celles qui concernent le joueur, celle pour la mise et enfin celles concernant le jeu.
Main.c s'occupe d'initialiser le premier joueur, le highscore etc… De plus ce fichier lance la boucle principale, s'occupe de la gestion du menu en fonction des réponses du joueur et permet le déroulement du jeu. Il s'appuie sur les fonctions de *craps.c*.

Presque toutes mes fonctions utilisent des pointeurs sur la structure player (renommée player_t grâce au typedef) afin de ne pas avoir à retourner la structure à la fin de mes fonctions la modifiant.

A l'initialisation de main.c un nouveau joueur est créé grâce a '*newplayer()*' retournant un pointeur sur la structure allouée. Il devient alors le premier joueur (pointé par *addr_first*), le joueur actuel de la boucle (pointé par *tmp*) et enfin le lanceur actuel (pointé par *ptr_lanceur*).
[Problème rencontré : Je n'ai pas fait pointer *ptr_lanceur* et *tmp* directement sur *first_player* sinon lorsque *addr_first* pointait une autre structure, les deux autres pointeurs changeaient également.]
D'autres variables sont également initialisées. Le highscore est chargé depuis le fichier *highscore.txt* s'il existe, sinon il est juste initialisé avec des 0 (fonction *init_highscore*).

Dans mon Nicecraps, Highscore ne contient que les 5 plus gros gains remportés. Il est à savoir que le joueur choisit également l'argent avec lequel il commence, il est donc facile de faire un nouveau record en rentrant avec une grosse somme de base. Les joueurs doivent donc se mettre d'accord sur quelque chose d'identique pour tout le monde.
Les joueurs possèdent également une sauvegarde qui mémorise leur nom (permet de recharger la partie leur correspondant) et leur argent avant de quitter la table. La sauvegarde ne prend pas en compte la mise actuelle, elle est alors perdue. Elle est

lue à chaque appel de *identification* par *read_save* et le joueur peut choisir de ne pas recharger sa partie qui sera alors écrasée par celle en cours.
Le jeu s'arrête quand il n'y a plus aucun joueur à la table. Le Highscore est alors enregistré, il ne faut donc pas quitter la partie en appuyant sur la croix de l'invite de commande. (sauf en cas de triche)
Pour ajouter un joueur il suffit de faire 'A' (ou 'a') dans le menu et d'entrer le nom et l'argent du nouveau joueur. Il vient s'ajouter à la fin de la table et jouera donc en dernier.

Le sens de jeu des joueurs se caractérise par une liste chaînée qui lorsqu'on arrive à la fin revient au début. Si un joueur est lanceur et qu'il quitte la partie, c'est alors le suivant qui devient lanceur ou bien le premier si on arrive à la fin de la liste. (fonction *player_leave*). D'ailleur si un joueur quitte la partie il est alors supprimé de la liste chaînée (grâce à la fonction *delete_player*), sauvegardé dans *save.txt* (fonction *write_save*) puis est *free* (par la fonction *player_leave*). Si le joueur était le dernier, *Continuer* (variable) se met à 0 et la boucle principale s'arrête.

[Problème rencontré : Pour les sauvegardes, je me trouve obligé de mettre des ";" à la fin de chaque ligne car l'argent du highscore ou des joueurs peut augmenter d'un chiffre et donc d'un caractère, ce qui écrase le '\n' voir même le nom du joueur suivant. Ce qui corrompt la sauvegarde.
Ainsi je mets un nombre variable de ';' en fonction du nombre de caractère de l'argent du joueur ou celui du record, tout cela afin d'obtenir un nombre fixe (10 caractères).
De plus, quand j'écris la nouvelle somme d'argent du joueur je rajoute un ';' à la fin car il peut passer de 1600 à 50 ce qui écrirait alors '5000'. Il peut alors y avoir des '50;0;;;' dans *save.txt* (cela ne corrompt pas la sauvegarde mais c'est moche).]

player_info (appelée par 'i' dans le menu) affiche le highscore actuel, ainsi que les joueurs, leur nom, argent et leur mise sur pass ou don't pass. Elle affiche également le point en cours.

Chaque joueur joue à tour de rôle, ils ne peuvent effectuer un changement de mise qu'une fois par suspension/point. Les dés sont lancés par le lanceur actuel une fois que tout le monde a voté pour le lancer de dé (fin de son tour; 'd' dans le menu). Le reste du jeu se passe selon les "Règles du jeu de NiceCraps".

Les mises sont séparées en deux fonctions *player_first_bet* et *player_change_bet.* *player_first_bet* s'occupe de placer la mise sur la ligne désirée par le joueur et de modifier les champs de sa structure en fonction des choix faits. *player_change_bet* permet de changer la mise du joueur ciblé en fonction du nombre des suspensions et de sa ligne (selon les règles données) .

Il est à savoir que si le joueur ne fait pas sa première mise il ne peut pas le faire par la suite et doit attendre la fin du point. (par souci d'équité)
L'argent du joueur ne peut dépasser 999999999$, ainsi le record peut être au maximum le double.
Si le joueur rentre une somme supérieure au début , on ne garde que les 9 premiers chiffres. (les autres chiffres entraînent une mauvaise saisie dans le menu juste après mais ne corrompt pas la partie)

[Problème rencontré : Si le joueur entre un nom plus grand que 29 caractères (taille allouée pour le nom) le jeu plante juste après. Cela n'a pas été corrigé.
Également lorsque le joueur entrait un caractère quand le jeu attendait un chiffre il bouclait sur l'erreur de saisie. J'ai réglé cela en passant par une variable temporaire, une chaîne de caractères, que je convertis en entier (grâce à *atoi*), et si c'est un caractère l'entier correspondant est 0. j'ai donc fait une boucle while vérifiant que l'entier est bien > 0.]

remarque : La plupart du temps quand je demande (o/n), répondre autre chose que 'o' est considéré comme 'n'.