AP2 TP5

M. WIART

Objectifs : Manipulation de structure et de tableaux de structure

Continuez à utiliser GDB pour comprendre le déroulement de vos programmes.
*Continuez à utiliser la bibliothèque assert pour insérer des assertions et la bibliothèque stdbool pour utiliser **true** et **false**.*
Compilez vos programmes avec l'option -Wall exemple gcc -c -Wall toto.c et n'acceptez pas qu'il reste un warning. Un warning doit vous alerter sur une erreur de programmation.

Calcul du score d'une partie de bowling

Règles de comptage des points:

Nombre de quilles tombées :

Au premier lancer de chaque frame, vous obtenez un nombre de points en fonction du nombre de quilles que vous aurez fait tomber. Si vous faites tomber toutes les quilles dès le premier lancer, vous aurez 10 points et ceci sera appelé un **Strike**.

S'il reste des quilles debout après le premier lancer, vous aurez le droit à un second lancer et le total des quilles qui tomberont sera ajouté à votre premier lancer. Si vous faites tomber le reste des quilles à ce deuxième lancer, vous aurez 10 points et ceci sera appelé un **Spare**.

Au bout de 10 frames, si vous réussissez à faire un **Strike** ou un **Spare**, vous aurez le droit (respectivement) à 1 ou 2 lancers supplémentaires.

Points supplémentaires :

En plus du nombre de quilles que vous aurez fait tomber, vous aurez le droit à des points de bonus en cas de **Strike** ou de **Spare** :

- Si vous faites un **Strike**, vos deux prochains lancers seront additionnés aux 10 quilles déjà tombées.
- Si vous faites un **Spare**, votre prochain lancer sera additionné aux 10 quilles déjà tombées

Il faut donc faire le plus gros score possible après un spare ou un strike pour marquer facilement des points. Le score maximal est de 300 et vous ne pourrez l'obtenir qu'en faisant 12 **Strikes** à la suite.

Structure de données

Un type structuré sera utilisé pour décrire les frames :

```
typedef struct {
  int lancer1 ;
  int lancer2 ;
  int lancer3 ;
  } Frame ;
```

Et un type structuré Joueur est également défini

```
typedef struct {
char * nom_joueur ;
int score ;
 Frame * jeu ;
   } Joueur ;
```

Exercice 1 : Ecrire une fonction d'initialisation d'un joueur (saisie du nom et allocation)

Exercice 2 : Ecrire une fonction de calcul du score après la fin du frame.

Exercice 3 : Ecrire une fonction d'affichage du score de la forme

toto	1	2	3	4	5	6	7	8	9	10
7	5 2									

Nous définissons un type structuré pour une partie

```
typedef struct {
int nb_joueur ;
 Joueur * joueurs ;
   } Partie ;
```

Exercice 4 : Ecrire une fonction d'initialisation d'une partie

Exercice 5 : Ecrire une fonction de déroulement d'une partie

Exercice 6 : Ecrire une fonction d'affichage de tous les joueurs.

Exercice 7 : Ecrire une fonction qui effectue le classement des joueurs

Facultatif : mais vivement conseillé !!!

Ecrire une fonction de sauvegarde des scores dans un fichier texte avec les noms des joueurs et une fonction de chargement qui lit le fichier et qui permet comme cela de créer une partie en cumulant les scores avec ceux de la partie précédente.