Exercices: Structures

Introduction au langage C – Coda 1 ère année – Septembre 2024

1 – Créer une structure Point qui représente les coordonnées d'un point dans un espace 2D (float x, float y)

2 – Créer une classe Vector2D composé de 2 points: origine et destination

3 – Créer une fonction qui retourne le milieu d'un Vector2D

```
Point get_middle_point(Vector2D vector);
```

4 – Écrire une fonction qui calcule la distance entre l'origine et la destination d'un vecteur 2D.

Utiliser la bibliothèque math.h et la fonction sqrt()

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

```
float get_vector_distance(Vector2D vector);
```

Exercices: Bataille

L'objectif est de coder un jeu de carte : la bataille. Les cartes sont représentées par un chiffre de 2 à 13 ainsi qu'une couleur Pique, Coeur, Carreau, Trèfle.

1 – Définir un enum pour représenter les 4 couleurs d'une carte

2 – Définir une structure Card qui représente une carte

3 – A partir de la structure suivante qui représente un paquet de carte

```
struct Deck {
    int size;
    Card cards[52];
};
```

note: size définit le nombre de carte dans le paquet.

Écrire une fonction init_deck qui initialise un paquet de carte avec les 52 combinaisons de carte possible.

4 – Écrire une fonction qui affiche toutes les cartes du paquet avec des printf.

5 – Écrire une fonction qui permet de mélanger un paquet. La fonction doit swap la position de 2 cartes et cela un certain nombre de fois

```
void shuffle (Deck *deck);
```

6 – Écrire une fonction qui permet de comparer 2 cartes. La fonction doit retourner 0 si la première carte est supérieur à la 2 eme, 1 sinon.

Dans le cas ou 2 cartes ont le même chiffre, la couleur permet de les départager dans l'ordre suivant : Pique, Coeur, Carreau, Trèfle

7– Écrire une fonction qui prends 2 Deck en paramètre et qui prend une carte au dessus du premier paquet pour la mettre en dessous du 2 eme paquet.

8– Écrire une fonction qui contient l'algorithme d'un tour de jeu

```
void playTurn(Deck *deck1, Deck *deck2)
```

Un tour fonctionne de la manière suivante : Chaque joueur tire la carte au dessus de leur paquet, celui qui a la carte la plus forte récupère les 2 cartes et les mets en dessous de son paquet.

9 – Terminer l'algorithme, votre programme doit jouer les tours automatiquement et afficher le vainqueur.