Mezouar Mikael Université Gustave Eiffel Peinzi Ferdyl

Rapport TP4 : Perfectionnement en programmation C

15/03/2021 L2 informatique

Introduction :

Dans ce TP, nous avons implémenté le Chomp, un jeu combinatoire abstrait à deux joueurs. Durant celui-ci , nous avons utilisé les notions de tableau statique à deux dimensions , type structuré, type énuméré et l'interface Ncurses .Ce TP nous a aussi permis de travailler sur les notions de modularisation que l'on a vu en cour ces dernières semaines . Dans ce rapport nous allons voir comment nous avons implémenté ce programme .

Écriture du programme :

Exercice 1. (Déclarations des types) :

1)

```c
/*Structure qui represente la tablette du jeu*/
typedef struct tablette {
    int tchoco[N][M];
} Tablette;
```

2)

```c
/*Structure booléene qui représente les 2 joueurs*/
typedef enum {
    JOUEUR_1,
    JOUEUR_2
} Joueur;
```

3)

```c
/*Structure qui represente une position de jeu */
typedef struct position{
    Tablette t;
    Joueur joueur;
} Position;
```

4)

```c
/*Structure qui represente un coup joué avec une coordonnée
typedef struct coup{
    int x;
    int y;
} Coup;
```

Exercice 2. (Manipulation des objets) :

1)

```c
/*fonction qui renvoie l'adversaire du joueur joueur*/
Joueur adversaire(Joueur joueur){
    Joueur rand;

    if (joueur== JOUEUR_1)
        rand = JOUEUR_2;
    else
        rand = JOUEUR_1;
    return rand;
}
```

2)

```c
/*Fonction qui permet d'initialiser la tablette*/
Tablette creer_tablette(void){
    int i,j;
    Tablette t;
    for (i = 0 ; i < N ; i++)
        for (j = 0 ; j < M ; j++)
            t.tchoco[i][j] = 1;
    return t;
}
```

3)

```c
/*fonction qui permet de manger la tablette selon les regles*/
void manger(Tablette *tab, int x, int y){
    int i,j;
    assert(x < M && x >= 0);
    assert(y < N && y >= 0);
    for (i = x ; i < N ; i++)
        for (j = y ; j < M ; j++)
            tab->tchoco[i][j] = 0;
}
```

4)

```c
/*fonction qui verifie si un coup est légal*/
int est_legal(Position pos, Coup coup){
    int x_temp, y_temp;

    x_temp= coup.x;
    y_temp= coup.y;
    if ((x_temp < 0) || ((x_temp) >= M) || ((y_temp) < 0) || ((y_temp) >= N)) /*Si le carré est en dehors de la tablette*/
        return 0;

    if (pos.t.tchoco[coup.x][coup.y] == 1){  /*Si le carré de chocolat existe*/
        return 1;
    }
    else{
        return 0;
    }
}
```

5)

```c
/*fonction qui verifie si le jeu est termine*/
int est_jeu_termine(Position pos, Joueur *joueur_gagnant){
    assert(joueur_gagnant != NULL);
    if (pos.t.tchoco[0][0] == 0){
        *joueur_gagnant = pos.joueur;
        return 1;
    }
    else{
        return 0;
    }

}
```

6)

```c
/*fonction qui joue le coup coup dans la position pos en paramètre */
void jouer_coup(Position *pos, Coup coup){
    int x_temp, y_temp;

    x_temp= coup.x;
    y_temp= coup.y;
    manger(&(pos->t), x_temp, y_temp);
    pos->joueur = adversaire(pos->joueur);
}
```

Exercice 3. (Assemblage du jeu) :

Par manque de temps et du à un problème avec ma VM (Mikael) , nous n'avons pas pu finir cette partie.

Exercice 4. (Modularisation) :

1) on peut décomposé le TP en 6 modules :

-un module **Chomp** qui va servir de fonction main et qui va appeler les autres modules .

-un module **Tablette** qui va créer type structuré permettant de stocker un tableau représentant le plateau du jeu

-Un module **Position** qui créer un nouveau type structuré qui stocke le plateau du jeu et la valeur de joueur Joueur en cours.

-Un module **Joueur** qui va créer un nouveau type énuméré qui contiendra le joueur 1 et 2.

-Un module **Coup** qui va créer un type structuré qui contiendra les coordonnées x et y d'un coup de l'un des joueurs .

-Un module **graphique** que l'on a pas réussi a faire dans le .c

Pour la suite des questions nous n'avons pas pu finir pour les même raison mentionner dans l'exercice 3.

Conclusion :

Pour conclure ce TP nous a permis une nouvelle fois de travailler avec la librairie Ncurses dans le but d'implémenter ce programme , on a aussi pu approfondir la notion de pré-assertion durant le développement de certaine fonction .Mais la grande nouveauté dans ,

c’est d’avoir pu travailler sur la notion de modularisation et d’avoir pu pouvoir réfléchir au découpage d’un projet . Par manque de temps et du a un problème sur ma VM (Mikael) régler tardivement , nous n’avons pas eu le temps de nous pencher énormément sur la partie graphique et sur la modularisation à la fin du TP .

Annexe 1 :

gcc Chomp.c -lncurses -pedantic -Wall -ansi -o chomp

Annexe 2 :