BENESBY Quentin
COUSSON Sophie
Tp1

TP5 : Le jeu du serpent

Introduction :

Ce projet a pour but de créer un jeu du serpent dans le terminal à l'aide de la librairie ncurses.

Exercice 1 (Conception du projet) :

1. Nous avons découpé le projet de sorte à avoir 5 modules : pomme, serpent, case, monde, graphique et le main. Nous avons fait un module par type et ensuite un module pour la partie graphique.
4. Voir le fichier makefile

Exercice 2 : (Déclaration des types)

1. Voir le fichier case.h.
   On déclare une structure Case, qui contient deux entier, x et y.

2. Voir le fichier case.h.
   On déclare un type énuméré qui contient les quatre directions possibles : nord, est, sud, ouest.

3. Voir le fichier serpent.h et le fichier case.h.
   On créer un type serpent qui contient une direction et un pointeur sur un type Cases, ce dernier contenant une liste chainée de cases et un élément case.

4. Voir le fichier pomme.h.
   On créer une structure Pomme qui contient une case.

5. Voir le fichier monde.h.
   On créer une structure Monde, qui contient un serpent, l'entier qui représente le nombre de pommes mangées et une liste chainée de pommes.

Exercice 3 : (Mécaniques du jeu)

1. Voir le fichier pomme.c
   On utilise une fonction creer_case qui permet de créer une case aux coordonnées données en paramètre, la fonction pomme_aleatoire donne en paramètre de creer_case deux entier aléatoires qu'elle a elle-même reçu en paramètre.

2. Voir le fichier monde.c
   On utilise la fonction précédente pour créer une pomme aléatoire sur le terrain, à l'aide de la fonction position_valide on vérifie que la nouvelle pomme n'est ni sur le serpent si sur une pomme déjà existante, puis on l'ajoute dans la liste de pomme du monde.

3. Voir serpent.c.
   La fonction initialise un serpent et le revoit, il se dirigera d'abord en direction de l'est et on créer taille cases sui se trouvent derrière la tete du serpent.

4. Voir le fichier serpent.c.
   La fonction serpent_case_visee renvoi la case sur laquelle le serpent sera après son déplacement, elle prend en compte la direction de celui-ci.

5. Voir le fichier monde.c.
   La fonction monde_initialiser créer un monde, dans lequel elle initialise un serpent (fonction de la question 3), et initialise une liste de pommes qui est rempli grave a la fonction monde_ajouter_pomme, qui va ajouter à la liste des pommes contenues dans monde une pomme sur une case aléatoire. On vérifie que cette case n'est pas déjà occupée grâce aux fonctions : positions_valide_pomme et position_valide_serpent.

6. Voir le fichier monde.c.
   Dans la fonction monde_est_mort_serpent, on vérifie d'abord que le serpent ne sort pas de la fenêtre de jeu, sinon il est mort. Ensuite on vérifie que le serpent ne se mord pas à l'aide de la fonction position_valide_serpent utilisée précédemment.

7. Voir le fichier monde.c.
   La fonction monde_evoluer_serpent va vérifier dans un premier temps que le serpent n'est pas mort, ensuite elle va le faire avancer tout en vérifiant que le serpent ne mange pas une pomme auquel cas il va grandir.

Exercice 4 : (Graphisme)

1. Voir le fichier graphique.c.
   La fonction dessine un rectangle dans le centre de l'écran, il n'y a pas de quadrillage a l'intérieur de celui-ci car cela alourdissait le rendu.

2. Voir le fichier graphique.c.
   La fonction interface_afficher_pomme dessine un P aux coordonnées auxquelles se trouve une pomme.

3. Voir le fichier graphique.c.
   La fonction interface_afficher_serpent fait appel à deux fonctions intermédiaires qui vont dessiner soit la tête (o) soit le corps (#).

4. Voir le fichier graphique.c.
   La fonction interface_afficher_monde dessines tous les éléments précédents dans la fenêtre et rafraichi l'écran.

5. Voir la fonction graphique.c.
   La fonction interface_piloter attend un évènement sur le clavier et si jamais le joueur a pressé une des touches z, q, s ou d alors on modifie la direction du serpent, sinon il ne se passe rien.

Exercice 5 : (Assemblage du jeu)

Voir le main.

Dans le main on déclare une variable monde qu'on initialise par la suite avec la fonction monde_initialiser, ensuite on fait une boucle while qui tant que le serpent n'est pas mort le fait avancer. Lorsqu'il meurt la fenêtre se ferme.

BENESBY Quentin
COUSSON Sophie
Tp1

Exercice 6 : (Améliorations)

Nous avons fait l'amélioration qui permet de mettre les caractéristiques de l'initialisation dans un fichier (serpent.ini). Nous avons créé un nouveau module sauvegarde. Pour cette amélioration nous avons modifié beaucoup de fonction et leurs appels grâce à une nouvelle structure Partie.

Conclusion :

Ce projet ne nous a pas posé de difficultés principales dans les fonctions principales, en revanche les améliorations nous ont posé problème, nous avons eu beaucoup de segmentation fault que nous n'arrivions pas à résoudre, nous avons donc seulement laissé deux améliorations. De plus, la suppression de pommes était aussi sur le même principe nous posant les mêmes problèmes.