Définition :

Cette notice est à destination des développeurs qui souhaitent reprendre le projet snake2024 afin de corriger les bugs ou bien d'ajouter des fonctionnalités. Cette notice met en exergue ce qui a été réalisé, ce qui reste à faire, la liste des fichiers et fonctions avec leurs définitions.

I. Ce qui a été fait :

Toutes les fonctionnalités du TP2 au TP6 ont été implémentées. Concernant les bonus demandés du TP7, la fonctionnalité qui fait grandir le serpent à chaque fruit mangé a été ajouté et plusieurs niveaux ont été créés et ajoutés dans le répertoire levels. Une fonctionnalité non demandé consistant à implémenter l'option -d via getopt_long pour ajuster la difficulté (easy, medium, hard) en modifiant la vitesse du serpent a également été réalisée.

II. Ce qui reste à faire :

Les bonus demandés du TP7, notamment :

- Utilisation du champ size lors de la capture des fruits et test du fin de jeu
- Plusieurs niveaux ont été créés mais le passage automatique d'un niveau à un autre lorsqu'on gagne reste à implémenter
- Vérification de la connexité du serpent pour valider le fichier
- Ajout de l'item qui permet des « continus »
- Modifier l'image des blocs et ajout des sons…

Corriger les bugs :

- La cellule où le serpent est initialisé en début de partie va écraser un bloc de la grille du fichier.
- Lorsqu'on utilise l'option -i le jeu va chercher dans le répertoire courant et non dans le dossier levels, il faut préciser le chemin si c'est un autre niveau que celui par défaut. Donc on peut ajouter une fonctionnalité qui va pointer directement le répertoire levels afin de rendre les commandes plus intuitives pour l'utilisateur.
- Trouver une manière d'inclure grid.h dans snake.h et inversement. Car j'ai dû faire des déclarations avancées pour la structure snake et la structure grid sans quoi j'avais des warnings.
- Lorsque les commandes de direction sont enchainées trop rapidement le serpent peut agir de manière étrange. Peut être augmenter la fréquence de rafraichissement.

III. Définition des différents fichiers :

- game.c :

C'est le fichier principal qui contient la fonction main. Il initialise le jeu, gère les arguments, génère la grille, positionne le serpent et contrôle la boucle principale du jeu. Le fichier gère les erreurs d'allocations de mémoire, d'ouverture de fichier ou lorsqu'une option non reconnue est entrée en ligne de commande en affichant l'aide.

- grid.c :

Ce fichier contient les fonctions de gestion de la grille de jeu. Il gère l'allocation de la mémoire pour la grille, le dessin de la grille, le déplacement du serpent sur la grille et le chargement de la grille depuis un fichier. Le fichier gère les erreurs d'allocation de mémoire, d'ouverture de fichier mais également la longueur des lignes des niveaux passés en paramètre. Si les longueurs ne correspondent pas à la première ligne alors le programme s'arrête avec un message d'erreur.

- grid.h :

Ce fichier contient les définitions, les structures, les énumérations et les prototypes nécessaires pour la gestion de la grille. Le fichier gère le problème des inclusions infinies avec ifndef et endif.

- snake.c :

Ce fichier contient les fonctions de gestion du serpent. Il gère la création du serpent, l'ajout de segments au serpent, la libération de la mémoire du serpent, le placement du serpent sur la grille et le calcul des coordonnées de la tête en fonction de sa direction. Le fichier gère les erreurs d'allocation de mémoire.

- snake.h :

Ce fichier contient les définitions, les structures, les énumérations et les prototypes nécessaires pour la gestion du serpent. Le fichier gère le problème des inclusions infinies avec ifndef et endif.

- makefile :

Ce fichier contient les instructions pour compiler le programme et copier les fichiers de niveaux. Il définit les options de compilation, les dépendances et les actions à effectuer pour nettoyer les fichiers compilés. Les erreurs sont gérées par le compilateur gcc et la commande cp.

- level :

Il existe 5 fichiers de level à inclure avec l'option -i. Ils sont construits de la même manière, c'est-à-dire avec des espaces, des « w » pour les murs, des « f » pour les fruits et les longueurs de chaque ligne doivent correspondre à la première. Les erreurs de level sont gérées par le fichier grid.c et notamment par la fonction grille_fic.

- .gitignore :

Ce fichier caché ne permet que d'afficher un dossier vide sur git car il ne permet pas de suivre les répertoires vides.

IV. Définitions des différentes fonctions :

Le détail des lignes est donné en commentaire directement sur le code.

Les fonctions du fichier game.c :

- print_help (ligne 9) : affiche le message d'aide et se termine. Elle ne gère pas d'erreurs.
- main (ligne 17) : c'est la fonction principale du programme. Elle initialise le jeu, gère les arguments provenant de la ligne de commande, charge la grille de jeu, créé le serpent et contrôle la boucle principale du jeu. Elle va gérer les erreurs d'option non reconnue, les ouvertures de fichier et l'allocation de mémoire. Elle affiche un message d'erreur et termine le programme sauf pour l'erreur des options non reconnues où elle va simplement afficher l'aide.

Les fonctions du fichier grid.c :

- allocate_grid (ligne 10) : permet d'allouer de la mémoire pour une grille de dimension n*m. La fonction gère les problèmes d'allocation de mémoire pour la structure grid ainsi que les erreurs d'allocation de mémoire pour les lignes de la grille. Elle affiche un message d'erreur et termine le programme.
- free_grid (ligne 49) : permet de libérer la mémoire allouée pour la grille. Elle ne gère pas d'erreurs.
- debug (ligne 59) : affiche la grille dans la console. Elle ne gère pas d'erreurs.
- compute_size (ligne 70) : permet de calculer la taille des cellules de la grille en fonction de la taille de la fenêtre. Elle ne gère pas d'erreurs.
- draw_grid (ligne 83) : permet de dessiner la grille dans la fenêtre graphique. Elle ne gère pas d'erreurs.
- move_snake (ligne 103) : permet de déplacer le serpent sur la grille. La fonction gère des erreurs d'allocation de mémoire pour un nouveau segment de la tête du serpent. Elle affiche un message d'erreur et termine le programme.
- compte_ligne (ligne 155) : permet de compter le nombre de lignes dans un fichier. Elle est utilisée dans la fonction grille_fic pour déterminer le nombre de lignes dans le fichier de configuration de la grille. Elle ne gère pas d'erreurs.
- copie (ligne 167) : permet de copier une chaine source vers une chaine de destination. Elle est utilisée la fonction grille_fic pour copier le contenu des lignes lues du fichier dans les lignes de la grille. Elle ne gère pas d'erreurs.
- grille_fic (ligne 177) : permet de charger une grille à partir d'un fichier. Il gère les ouvertures de fichier, la conformité des fichiers level (longueur des lignes par rapport à la première), l'allocation de mémoire et la lecture de fichier. Elle affiche un message d'erreur et termine le programme.

Les fonctions du fichier snake.c :

- new_snake (ligne 7) : permet de créer un nouveau serpent et gère l'allocation de mémoire pour la structure snake. Elle affiche un message d'erreur et termine le programme.
- add_segment (ligne 21) : permet d'ajouter un segment au serpent et gère l'allocation de mémoire pour les nouveaux segments. Elle affiche un message d'erreur et termine le programme.
- free_snake (ligne 45) : permet de libérer la mémoire allouée pour le serpent. Elle ne gère pas d'erreurs.
- place_snake (ligne 58) : permet de placer le serpent sur la grille. Elle ne gère pas d'erreurs.
- crawl (ligne 71) : permet de calculer les coordonnées de la tête du serpent en fonction de sa direction actuelle. Elle ne gère pas d'erreurs.