RAPPORT TP3

Rapport de TP

Tables des matières:

Objectifs/Enjeux

L'objectif de ce TP est de programmer le jeu du serpent, avec la **bibliothèque graphique MLV**. Le principe est de piloter un serpent pour attraper des pommes et grandir progressivement. La difficulté principale consiste au mouvement du serpent et le contact avec les bords de l'écran ou avec lui-même ce qui provoque la défaite.
L'objectif était également la modularisation des fichiers du projet, c'est pourquoi chaque fonctions et structures sont regroupées dans des fichiers séparés en fonction de leur(s) utilité(s).

Moyens mis en œuvre

Pour le bon déroulement de ce TP, nous avons eu recours à différentes documentations :

1. La documentation de la **bibliothèque graphique MLV;**
   http://www-igm.univ-mlv.fr/~boussica/mlv/index.html

2. L'utilisation d'un fichier **Makefile**;

3. l'utilisation de **module** .c et .h.

Une fois le programme lancé sur le terminal, on a à présent devant nous le menu principal. Nous pouvons cliquer sur « **Jouer »**, «**Meilleur score »** ou encore « **Quitter ».** Cliquez sur la **croix de la fenêtre** ou encore sur le bouton « **Quitter** » pour fermer le jeu. Pour rejouer la ou les parties il suffit de cliquer sur le bouton « **Rejouer** » qui se présente à la fin du jeu précédé par l'affichage du TOP 3 au début et à la fin du jeu.

Questions du TP

Exercice 1 (Conception du projet) :

Question 2 :

On suppose que notre programme sera réparti en plusieurs fichier avec un nom explicite qui permettra de regrouper les fonctions du même genre.

On a découpé notre programme en **5 fichiers .c suivi de leur fichiers .h respectifs** , le **main.c** et le **Makefile** :

- **main.c** correspond au fichier principal ;
- **graphique.c** regroupe la partie graphique du jeu ;
- **jeu.c** qui contient les fonctions principale pour jouer;
- **monde.c** correspond à l'initialisation et du déroulement du jeu ;
- **score.c** à les fonctions qui servent à sauvegarder dan un fichier le nouveau score qui est accepté suivi de son temps effectué ;
- **menu.c** concerne toute les fonctions qui permettent l'affichage et l'utilisation du menu.

Structures rajoutées

```
typedef struct Bouton {
        int x1;
        int y1;
        int x2;
        int y2;
        int numero_bouton;
} Bouton;
```

Cette structure sert à redimensionnée et afficher les boutons du menu.

```
typedef struct{
        int largeur, hauteur;
        int nb_pommes;
        int taille_serpent;
        int vitesse;
        int duree_tour;
} Partie;
```

La structure Partie possède tout les paramètre d'initialisation, donné dans le fichier « **Serpent.ini** », pour le commencement d'une partie.

Amélioration

En plus des consignes du TP, diverses fonctionnalités ont été implantées au programme. Il y au total 12 étoiles accumulées :

- Touche pause implémentée en appuyant sur espace.
- Graphisme amélioré par des images pour chaque éléments correspondant.
- Pommes empoisonnées générées qui provoque une défaite lorsqu'on en mange.
- Pomme en or qui apporte deux points supplémentaire au score.
- Possibilité de modifier la durée de jeu dans le fichier « **Serpent.ini** » pour effectuer un meilleur score durant le temps imparti.
- Affichage des trois meilleurs scores avec le temps effectué. Possibilité de voir le tableau de score sur le menu, avant de jouer et après.
- Musique implémentée dans le jeu : musique de fond, musique de gain de point, musique de pause, musique de fin de partie.
- Programme paramétrable à partir du fichier « **Serpent.ini** ».

Exécution du programme

L'exécutable **main** se génère en exécutant le fichier le script **make** mais aussi **./snake**. Faire un **make clear** pour effacer les **fichiers .o**

Conclusion

En conclusion, ce TP nous a fait travailler sur deux points essentiels : l'utilisation de la bibliothèque graphique MLV, ainsi que sur la modularité. Le fait de découper son projet en plusieurs fichiers le rend plus organisé. De plus certains fichiers peuvent être indépendants et ainsi être utilisés dans de futurs projets.

Le problème majeur se situait dans la modularisation du programme, il était difficile de concevoir un schéma avant de programmer.

L'utilisation d'un fichier .ini est tout nouveau , il est utile de pouvoir paramétrer un programme de cette façon.