1 Grad Info Soir
Langage C - Examen Juin 2007

1. Explications

L'examen comprend un projet (à choisir parmi les deux projets décrits ci-dessous) à réaliser à domicile et à documenter :

- structure des données,
- les routines dont dispose chaque structure.
- organisation du programme par un schéma bloc
- listing de l'application

Le dossier doit être rendu une semaine avant le jour de l'examen (début juin)**.**

L'examen proprement dit est la présentation orale du programme, sa démonstration et l'explication de son fonctionnement.

2. Les incontournables

L'accent est mis sur un bon découpage du programme en structures, fonctions et fichiers

L'utilisation de **structures** est obligatoire.

L'utilisation de la **compilation séparée** est obligatoire : un fichier .C et un fichier .H par structure réunis à l'aide d'un fichier projet.

Tous les tableaux, vecteurs, listes doivent utiliser l'**allocation dynamique** (malloc, free) sauf dans le cas de tableaux dont le nombre d'éléments est connu est constant par définition (p.ex.: la liste des touches acceptées dans un menu).

Le projet doit contenir et utiliser au moins à un endroit **une pile** pour stocker les structures (p.ex. les lignes des menus)

Le programme doit permettre la sauvegarde et la reprise de données dans un **fichier**.

Le programme doit permettre une **recherche** sur base d'un critère quelconque (par nom, etc.)

3. Projet Encryption

3.1. Enoncé

On demande de réaliser un programme qui permet de crypter n'importe quel fichier à l'aide d'une clé de cryptage. Le procédé utilisé est un XOR entre les bytes du fichier et les bytes de la clé de cryptage. La même routine est utilisée pour le cryptage et le décryptage.

Les fichiers cryptés recoivent un nom aléatoire (de longueur constante) et une extension standard (p.ex. .CRY). Tous les fichiers sont stockés dans un répertoire commun défini par l'utilisateur.

Le programe conserve les caractéristiques de tous les fichiers cryptés :

- chemin original,
- nom original,
- extension original,
- nom après cryptage (clé identifiante)
- clé de cryptage
- date et heure

Ces informations sont elles-mêmes cryptées et conservées dans une base de données du programme (un fichier) toutefois, la clé de cryptage employée à ce niveau peut être codée "en dur" dans l'exécutable.

3.2. Fonctionnement

On propose ci-dessous une structure de menu et de fonctions. Il est permis de s'en écarter ou de réorganiser les options l'idée est de présenter les fonctions de création, modification, recherche et suppression pour chaque item.

Le menu principal propose à l'utilisateur

- le cryptage d'un fichier
- le décryptage d'un fichier sauvé
- suppression d'un fichier crypté
- les options
- quitter

Le menu de cryptage permet

- saisie du chemin+nom du fichier
- introdution de la clé de cryptage
- valider (et crypter) ou abandonner

Le menu de decryptage permet

- affichage de la liste des fichiers cryptés
- choix du fichier à décrypter
- recherche par nom ou extension
- introdution de la clé de cryptage
- valider (et décrypter) ou abandonner

Le menu des options permet

- défintion du répertoire par défaut,
- etc.

4. Projet JEU

4.1. Enoncé

On demande de réaliser un programme qui permet de définir des personnages destinés à un jeu.

Le programme permet à un nombre quelconque de joueurs de définir ses personnages. Chaque joueur peut créer un maximum de P personnages (option) dispose d'un capital C de points.

Chaque joueur est caractérisé par

- un code
- un pseudo

Chaque personnage est caractérisé par

- son nom
- le code du joueur auquel il appartient
- sa force
- son agilité
- son intelligence
- son expérience
- sa santé

Tout nouveau personnage reçoit 100 points de santé et 0 point d'expérience.

Le joueur attribue des points de force, d'agilité et d'intelligence à ses personnages en puisant dans le capital dont il dispose avec un maximum de 100 points par qualité.

Les joueurs ont la possibilité de sauver la définition des personnages et d'y revenir par la suite.

L'implémentation du jeu proprement dit (interactions entre personnages) n'est pas obligatoire.

4.2. Fonctionnement

On propose ci-dessous une structure de menu et de fonctions. Il est permis de s'en écarter ou de réorganiser les options l'idée est de présenter les fonctions de création, modification, recherche et suppression pour chaque item.

Le menu principal propose à l'utilisateur

- la gestion des joueurs
- la gestion des personnages
- les options
- jouer (pas obligatoire)
- quitter

Le menu de gestion d'un joueur permet

- création d'un joueur
- recherche et sélection d'un joueur
- modification du joueur sélectionné
- destruction du joueur sélectionné
- retour au menu précédent

Le menu de gestion des personnages

- recherche et sélection d'un joueur
- création d'un personnage

- recherche et sélection d'un personnage
- modification du personnage sélectionné
- destruction du personnage sélectionné
- retour au menu précédent

Le menu des options permet

- défintion du répertoire par défaut,
- définition du capital de points par défaut
- définition du nombre de personnage maximum
- etc.

Pour ceux qui vourdraient aller plus loin

Le menu jouer propose
- sélection du joueur 1
- sélection du joueur 2
- jouer un tour

Lors d'un tour, chaque joueur sélectionne un de ses personnages. Les caractéristiques des deux personnages sont transimises à une routine qui calcule

- la diminution de points de vie en fonction des différences de force, d'agilité, d'expérience, etc des personnages + un élément aléatoire. On ne gagne jamais des points de vie
- l'augmentation des point d'expérience selon le même principe. On ne perd jamais des points d'expérience.
- etc

Les modifications sont sauvées dans le fichier des personnages.

5. Conseils pour la réalisation

La réalisation des projets est moins compliquée qu'il n'y paraît si on tient compte du fait que le projet est structuré verticalement :

Menu Ù Entité (structure) Ù Fichier

En d'autres mots, il "suffit" de résoudre séparément la gestion des menus d'un côté, celle des entités de l'autre et celle des fichiers d'un troisième.

Il est conseillé de commencer par l'implémentation des "fiches" en simplifiant le problème au maximum

- définir la structure avec ses fonctions habituelles :
- tester chacune de ces fonctions dans un programme principal élémentaire,
- ajouter des fonctions liées au fichier telles que sauvegarde, reprise, recherche(s)
- définir les menus

  Un menu n'est qu'une pile de strings. En plus des fonctions habituelles, il lui faut (entre autres) une fonction de chargement à partir d'un fichier, une fonction d'affichage et une fonction de choix.
- tester le menu *seul,*
- interfacer le menu avec la structure dont il assure la gestion.
- enfin, hiérarchiser les menus.

6. Pour terminer

La discussion entre élèves est encouragée mais chaque élève doit rendre un projet personnel (ce qui se reconnaît immédiatement au style de programmation) et le connaître sur le bout des doigts (ce qui se voit immédiatement par les explications sur la résolution de certains problèmes). La résolution du problème peut être implémentée de différentes manières, chacun peut choisir sa solution.

Mieux vaut un projet dans lequel on intégre proprement *tous* les "incontournables" dans un nombre limité de fonctions qu'un foutoir dans lequel on fait tout n'importe comment. Ceci dit, seul un projet complet et correct mérite le maximum de points.

Bon courage !