Exercices : module en C

Les exercices suivants correspondent à différents module d’une même application. L’application en elle-même sera une fonction main dans un fichier dédié, qui inclura le .h de chaque module. Aucun scanf ni aucune instruction susceptible de recevoir des lettres tapées au clavier ne doivent être appelées au sein des modules. Elles ne peuvent être appelées QUE dans main.c

# Module 1 : entrepôts et statistiques

On souhaite réaliser un module de statistiques sur les ventes d'une boutique. L'objectif de cet exercice est d'écrire un programme qui permette de définir des produits et qui, pour chaque produit, permette de faire évoluer son stock, pour au final faire des statistiques sur les produits les plus vendus, les moins vendus, etc...

- Ecrire une structure produit contenant un nom (au plus 50 caractères), un identifiant (entier), un prix (flottant), un stock (entier) et le nombre de baisses de stock (entier).

- Ecrire une structure entrepôt contenant un nom de magasin et une collection de produits (au plus 500 produits).

- Créer un alias vers un pointeur sur entrepôt ainsi qu'un alias vers un produit (pas un pointeur sur produit).

- Ecrire une fonction qui reçoit un nom en paramètre et qui créé et renvoie un entrepôt ayant pour nom celui reçu et une collection de produits vide. Cette fonction ne doit pas renvoyer un pointeur sur entrepôt mais bien un entrepôt. Un produit est vide si son identifiant est égal à 0;

Toutes les fonctions suivantes prendront un pointeur sur entrepôt en paramètre. Elles prendront peut-être également d'autres paramètres, à voir selon vos besoins.

- Ecrire une fonction ayant comme prototype void PrintWarehouse(Warehouse myWarehouse). Cette fonction doit afficher sur la console le nom du magasin reçu en paramètre ainsi que les identifiants, noms, stocks et nombres de baisses de stocks de chaque produit non vide dans le magasin.

- Ecrire une fonction ayant comme prototype int CreateProduct(Warehouse myWarehouse, Product newProduct). Cette fonction doit chercher si un produit ayant le même identifiant que newProduct existe. Si c'est le cas, elle renverra 0. Sinon elle ajoutera newProduct à la collection de produits présente dans myWarehouse.

- Ecrire une fonction qui permette de modifier le stock d'un produit dont l'identifiant est reçu en paramètre.

- Ecrire une fonction qui renvoie la quantité de produits dont le stock est inférieur à une valeur reçue en paramètre.

# Module 2 : sérialisation en fichier texte

On souhaite réaliser un module permettant de stocker le contenu d’un entrepôt dans un fichier. Chaque produit tiendra sur une ligne, chaque champ du produit sera séparé des autres par un pipe (symbole |).

* Ecrire une fonction statique permettant d’écrire un produit dans un fichier
* Ecrire une fonction statique permettant d’écrire un entrepôt (pas ses produits) dans un fichier
* Ecrire une fonction appelable depuis un autre module permettant d’enregistrer un entrepôt et chacun de ses produits dans un fichier en faisant appel aux deux précédentes.
* Ecrire une fonction qui renvoie un entrepôt à partir d’un fichier dont le nom est reçu en paramètre.

# Module 3 : panier

On souhaite réaliser un module permettant de stocker les produits que l’on voit dans un panier, pour ensuite éventuellement passer une commande.

* Ecrire une structure produit contenant un nom (au plus 50 caractères), un identifiant, un prix et une quantité (entier positif ou nul)
* Ecrire une structure panier contenant un ensemble de produits
* Créer un alias vers un pointeur sur panier.
* Ecrire une fonction permettant de créer un panier.
* Ecrire une fonction permettant d’ajouter un produit à un panier.
* Ecrire une fonction permettant d’afficher sur la console le prix total d’un panier ainsi que tous ses composants.
* Ecrire une fonction permettant de retirer un produit d’un panier.

# Module 4 : passage de commande

On souhaite réaliser un module permettant de décrire les différentes étapes du passage de commandes d’un site de e-commerce. Chaque annulation ou validation de commande doit être enregistrée dans un fichier texte.

* Ecrire une structure commande contenant un identifiant, un état (en cours, annulé, validé), et un panier (cf module de panier).
* Ecrire une fonction permettant d’annuler une commande en cours.
* Ecrire une fonction permettant de valider une commande en cours.
* Ecrire une fonction permettant d’enregistrer une commande dans un fichier texte dont le nom sera reçu en paramètre.
* Ecrire une fonction permettant d’afficher sur la console toutes les commandes enregistrées.

# Main : orchestration des modules

Ecrire dans le fichier main.c une fonction main et tout un menu utilisateur permettant au choix, de gérer ses entrepôts ou de faire ses courses.

* Gérer ses entrepôts renvoie sur les fonctions du premier module. On doit pouvoir créer des produits, des entrepôts, etc. Toute modification d’un entrepôt ou d’un produit entrainera le réenregistrement dudit produit ou entrepôt dans un fichier texte dont le nom sera fixe pour tout le programme.
* Faire ses courses envoie aux fonctions du panier. A tout moment l’utilisateur doit pouvoir passer sa commande. Toute commande validée ou annulée sera immédiatement enregistrée.