TD1 : Allocation Dynamique de Mémoire & Enregistrements

Exercice N° 1 :

Une école souhaite calculer les résultats d'une classe. Le nombre d'étudiants varie d'une classe à une autre et n'est donc pas connu à l'avance.

Écrire un programme C qui :

1. Demande à l'utilisateur de saisir le nombre d'étudiants n.
2. Déclare un pointeur **notes** et alloue dynamiquement un tableau permettant de stocker les **n** notes.
3. Vérifie que l'allocation mémoire a réussi.
4. Demande à l'utilisateur de saisir les notes des n étudiants.
5. Calcule et affiche :
   - la moyenne de la classe ;
   - la note maximale ;
   - la note minimale ;
6. Libère la mémoire allouée avec **free**().

Exercice N° 2 :

Un magasin souhaite enregistrer le nombre de produits vendus chaque jour pendant une période dont la durée n'est pas connue à l'avance.

1. Déclarer les fonctions suivantes :
   void saisirVentes(int *ventes, int n);
   void afficherVentes(int *ventes, int n);
   int totalVentes(int *ventes, int n);
   int jourMeilleureVente(int *ventes, int n);
   Ces fonctions permettent de :
   - saisirVentes : saisir le nombre de produits vendus chaque jour ;
   - afficherVentes : afficher les ventes ;
   - totalVentes : calculer le nombre total de produits vendus ;
   - jourMeilleureVente : retourner le numéro du jour ayant enregistré le plus de ventes ;
2. Écrire un programme C qui :
   a) Demande à l'utilisateur le nombre de jours n.
   b) Alloue dynamiquement un tableau de n entiers avec malloc().
   c) Vérifie que l'allocation a réussi.
   d) Affiche les résultats obtenus des fonctions déclarées.
   e) Libère la mémoire avec free().

Exercice N° 3 :

Nous voulons développer une petite application pour gestion des étudiants.

1. Déclarer une structure **Etudiant** pour modéliser un étudiant (nom, age, note).
2. Créer une fonction « **Etudiant lireEtudiant()** » qui permet de lire et retourner un étudiant.
3. Ecrire une fonction « **void afficherEtudiant(Etudiant etu)** » qui affiche les informations d'un étudiant passé en paramètre.
4. Ecrire une fonction « **float moyenne(Etudiant *et, int n)** » qui calcule et retourne la moyenne des **n** étudiants passés en paramètres.
5. Dans la fonction main(), demander à l'utilisateur le nombre n d'étudiants, allouer dynamiquement un tableau de n étudiants, lire leurs informations, puis afficher la liste et la moyenne générale.

Exercice N° 4 :

Écrire un programme en C qui :

1. Définit deux structures : une structure **Adresse** contenant : ville, rue, codePostal et une structure **Employe** contenant : nom et un champ adr de type Adresse.
2. Ecrire une fonction « **Employe lireEmploye()** » qui lit les informations d'un employé (nom et son adresse complète).
3. Affiche toutes les informations de l'employé à l'aide d'une fonction « **void afficherEmploye (Employe e)** ».
4. Une fonction **main** pour lire n employés et les afficher sur écran.