Programmation Imperative

L1 2021-2022

Travaux Pratiques 6
(Les Structures)

Exercice 1. *Nombre rationnel*

1. Définir une structure `Rationnel` permettant de coder un nombre rationnel (i.e. un numérateur et un dénominateur).
2. Écrire une fonction `saisie` et une fonction `affichage`.
3. Écrire une fonction `addition` pour deux nombres rationnels.
   Indice : Il n'est pas nécessaire de simplifier le resultat d'addition.
4. Écrire une fonction `multiplication` pour deux nombres rationnels.

Exercice 2. *Stock d'écrans*

Une entreprise d'informatique gére un stock d'écrans. Chaque écran est caractérisé par :
— le nombre de points (**pixel**) que l'écran peut afficher,
— la taille qui se calcule en mesurant la diagonale de l'écran et est exprimée en **pouces** (un pouce équivaut à 2,54 cm),
— la résolution, qui détermine le nombre de pixels par unité de surface et est exprimée en **dpi** (en anglais DPI : Dots Per Inch).

1. Définir une structure `ecran` contenant tous les informations relatives à une écran.
2. Pour éviter la repetition de mot `struct`, définir un raccourci `Ecran` par un `typedef`.
3. Initialiser un tableau `TableEcran[N]` en utilisant votre structure personnalisée.
4. Écrire une fonction `Ecran saisie(void)` pour entrer des informations d'écran.
5. Écrire une fonction `void enregister(Ecran *tab)` qui prend les informations de N écrans et les enregistre dans le tableau.
6. Écrire une fonction `void affichage(Ecran *tab)` qui affiche les informations de N écrans enregistrés. *Indice : Vous pouvez utiliser une fonction intermédiaire* ***`affichage_ecran`*** *qui affiche une structure de type* ***`Ecran`****.*
7. Écrire une fonction `void dispo(Ecran *tab)` qui demande à l'utilisateur d'entrer la caractéristique qui l'intéresse et qui imprime tous les écrans qui ont cettes caractéristiques.
8. **Bonus :** Améliorer votre fonction `int dispo(void)` en donnant la possibilité à l'utilisateur d'entrer une/deux/trois caractéristiques qui l'intéressent.
9. **Bonus :** Modifier votre programme en ajoutant également la marque de l'écran dans la structure `ecran` (chaîne de caractères).

Exercice 3. *Occurrences d'une lettre dans la chaîne de caractères*

Faire une fonction `int occ(char *str, char lettre)` qui prendre en paramètre une chaîne de caractères et renvoie le nombre d'occurences d'une lettre donné par l'utilisateur dans la chaîne.

Exercice 4. *Sans utiliser string.h*

Faire une fonction `char *copy_char(char *str)` qui recopie une chaîne de caractères `s` et revoie une copie de cette chaîne.

Exercice 5. *Verbe du premier groupe*

Écrire un programme qui lit un verbe du premier groupe et qui en affiche la conjugaison au présent de l'indicatif, sous la forme :
je chante
tu chantes

il chante
nous chantons
vous chantez
ils chantent

Le programme devra vérifier que le mot fourni se termine bien par "er". On suppose qu'il ne peut comporter plus de 26 lettres et qu'il s'agit d'un verbe régulier. Autrement dit, on admettra que l'utilisateur ne fournira pas un verbe tel que "manger" (le programme afficherait alors : "nous mangons").

Indice : Utilisez les fonctions prédéfinies "strlen", "strcmp" et "strcpy".

Exercice 6. *Nombre de lettre "e"*

Écrire un programme déterminant le nombre de lettres "e" (minuscules) présentes dans un texte de moins d'une ligne (supposée ne pas dépasser 132 caractères) fourni au clavier.

Indice : Utilisez la fonction prédéfinie "strchr".