1 Coordonnées GPS

Les coordonnées GPS (*Global Positioning System*) ont une importance primordiale dans notre société moderne. Elles permettent de géolocaliser précisément un point à n'importe quel endroit sur la surface de la Terre. Elles sont évidemment à la base de tous les systèmes de navigation. L'Université des Sciences de Limoges se trouve par exemple au point de coordonnées suivantes :

$$45°50'14'' \text{ N}, 1°14'17'' \text{ E}.$$

Formellement, les coordonnées GPS se divisent en 2 parties : la latitude et la longitude. Elles sont toutes les deux données avec 3 nombres représentant des mesures d'angle en degrés (°), minutes (′) et secondes (″) [1].

La latitude finit soit par N pour Nord ou S pour Sud et la longitude par W ou E pour Ouest et Est. Notez par exemple que la longitude de Limoges est décalée d'un seul degré par rapport à celle du Méridien de référence de Greenwich, c'est en quelque sorte comme si nous étions exactement "en dessous" de la ville anglaise.

FIGURE 1 – Le fameux ticket de loterie de la série *Breaking Bad*, il est utilisé par Walter White pour stocker discrètement des coordonnées GPS de première importance...

1. Le fait que les sous-unités du degré soient semblables à celles du temps n'a rien d'étonnant car les deux nous viennent des Babyloniens qui comptaient en base 60 (écriture sexagésimale).

1.1 Exercice 1 : paramètres et structures

Question 1 : Définir une structure `gps_position` contenant deux tableaux de 4 entiers, un pour la latitude et un pour la longitude. Le 4ème entier de chaque tableau vaut 0 pour Nord et Ouest et 1 pour Sud et Est. Attention à ne pas prendre des types d'entiers trop grands sachant que les unités des coordonnées GPS ne dépassent jamais 180.

Question 2 : Créez une fonction de type `void` qui reçoit en paramètre une structure `gps_position` (pas le pointeur, la structure en elle-même) et qui en modifie le contenu.

Question 3 : Même question, mais cette fois-ci la fonction a comme paramètre un élement de type `gps_position*`.

Question 4 : Qu'observez-vous sur les deux manières de procéder des précédentes questions [2] ? Une fois que vous aurez choisi la meilleure, utilisez-la pour créer une fonction `void set_limoges(...)` qui modifiera une structure `gps_position` déclarée dans le `main` pour lui donner les valeurs des coordonnées GPS de l'Université de Limoges.

Question 5 : Créez une fonction `void affiche_gps` qui admet comme paramètre un pointeur vers une structure `gps_position`. Cette fonction affiche alors la structure initialisée lors de la question précédente comme ceci :

```
$ latitude : 45 deg 50' 14'' N, longitude : 1 deg 14' 17'' E
```

Bonus : petit challenge, réalisez cette fonction sans utiliser de condition `IF/ELSE`.

2. Ne paniquez pas, il est tout à fait normal que l'une des deux ne marche pas !

1.2 Exercice 2 : paramètres et tableaux

Dans cet exercice, on manipule encore des coordonnées GPS, mais cette fois-ci une coordonnées GPS est juste un tableau de 8 entiers (les 4 premiers pour la latitude et les 4 suivants pour la longitude).

Question 1 : Déclarez un tableau contenant des coordonées GPS dans le `main` et initialisez le à 0.

Question 2 : Créez une fonction `void set_limoges` qui admet comme paramètre un tableau de coordonnées GPS et qui initialise ses valeurs à celles de l'Université des Sciences de Limoges comme dans l'exercice précédent.

Question 3 : Quelle différence remarquez-vous par rapport aux questions 2 et 3 de l'exercice précédent ?

1.3 Exercice 3 : Manipulation mémoire et structures

Pour cet exercice, vous allez reprendre l'exercice 1 et importer la bibliothèque `string.h`.

Question 1 : Sans coder, simplement en réflechissant, quelle est la taille (en octets) de la structure `gps_position` ?

Question 2 : Quelle fonction en C permet de vérifier votre réponse à la question précédente ?

Le but de cet exercice est de modifier la latitude de l'Université de Limoges dans la structure déjà initialisée (appelons-la `fst_limoges`) **simplement en manipulant un pointeur vers cette structure.**

Question 3 : Créez un pointeur vers une structure de type `gps_position` et donnez-lui la valeur de l'adresse de la structure `fst_limoges`.

On souhaite modifier la valeur 50 pour la remplacer par 0, pour cela on va utiliser la fonction `memset(adresse,nouvelle_valeur,1)` qui permet de modifier un seul octet [3] à l'adresse `adresse` pour lui donner la valeur `nouvelle_valeur`. Le problème est qu'il faut pour cela connaître l'adresse précise de l'octet contenant la valeur 50, et pour le moment on ne connaît que l'adresse de la structure elle-même.

Question 4 : Pour résoudre le problème mentionné ci-dessus, commencez par voir comment varie la valeur du pointeur si vous cherchez à afficher la valeur de *pointeur*+1. Le phénomène que vous observez est en lien avec ce que l'on appelle l'*arithmétique des pointeurs*. Pour arriver à modifier correctement la valeur de ce pointeur, caster-le en pointeur vers un `unsigned char`. Vous avez alors tous les éléments en main pour finir la question.

Question 5 : En utilisant la fonction

```
memcpy(adresse_source, adresse_destination, nombre_octets)
```

3. On peut en modifier plus, mais dans le cadre de cet exercice considérons simplement que le dernier paramètre de la fonction vaut 1.

qui permet de copier un bloc de `nombre_octets` octets d'une adresse source vers une adresse destination, remplacez la latitude de Limoges par sa longitude. Vous aurez pour cela besoin de manipuler des pointeurs comme dans la question précédente.

Question 6 : (Question de cours, pas besoin de coder) La fonction `strcpy(adresse_source,adresse_destination)` permet de copier une chaîne de caractères d'une adresse source vers une adresse de destination, c'est une fonction cousine de `memcpy()`. Cependant, si vous regardez bien, elle a un argument en moins, pourquoi ?