DS 1

Informatique MP2I

Julien Reichert

Toutes les questions de programmation sont à résoudre en C.

Questions de cours ou d'application directe du cours

Question de cours 1 : Dans quel contexte était-il question de l'écriture scientifique ?

Question de cours 2 : Expliquer le principe d'une preuve de terminaison dans la seule situation déjà traitée en cours où la preuve n'est pas évidente.

Question de cours 3 : Donner au moins deux notations de Landau avec leur explication précise.

Question de cours 4 : Comment s'appelle la zone de la mémoire réservée aux variables locales d'une fonction ?

Question de cours 5 : Quel est le couple de fonctions au programme impliquées en C par l'allocation et la libération dynamiques de la mémoire ?

Question de cours 6 : Donner la syntaxe précise d'une boucle en C utilisant `for` en expliquant un exemple de ligne introduisant une telle boucle. Est-ce par ailleurs en pratique une boucle inconditionnelle ou une boucle conditionnelle ?

Question de cours 7 : Donner les symboles respectifs de l'opérateur de référencement et de l'opérateur de déréférencement en C.

Question de cours 8 : Qu'est-ce qui prend le plus de place en mémoire en C, en considérant les tailles observées ici pour les types standards au programme, entre l'entier `1000643572`, le flottant `324.5`, l'adresse de ce dernier flottant et la chaîne de caractères `"abcdefgh"` ?

Question de cours 9 : Quel est le problème dans le code suivant, écrit en C ? (On ne demande pas de corriger le problème !)

```
#include <stdio.h>

int* f(int x)
{
  int moinsx = -x;
  int reponse[2];
  if (moinsx < x) { reponse[0] = moinsx; reponse[1] = x; }
  else { reponse[0] = x; reponse[1] = moinsx; }
  return reponse;
}

int main()
{
  int x = -42;
  int* x_et_moinsx = f(x);
  printf("tableau : %d et %d.\n", x_et_moinsx[0], x_et_moinsx[1]);

  return 0;
}
```

Exercices issus des TD et TP

Exercice T1 : Écrire une fonction prenant en argument un tableau d'entiers supposé croissant (**évidemment** pas à vérifier) et sa taille ainsi qu'un autre entier et renvoyant un indice où se situe cet entier dans le tableau (ou `-1` s'il n'y est pas), **en procédant obligatoirement par dichotomie** (sinon l'exercice n'est pas corrigé).

Exercice T2 : Écrire une fonction prenant en entrée deux entiers `n` et `k` et retournant la somme des puissances `k`-ièmes des entiers de `1` à `n`. On admettra que les deux entiers sont positifs et le comportement est au choix sinon.

Exercice T3 : Écrire une fonction qui prend en argument une chaîne de caractères et qui calcule sa taille exactement comme `strlen`, bien évidemment sans s'en servir.

Exercice T4 : En supposant l'existence d'une fonction de prototype `int difference_jours(jour j1, jour j2)` prenant en argument deux structures de date et renvoyant le nombre de jours entre la première et la deuxième (strictement positif si la deuxième est ultérieure à la première et négatif sinon), écrire une fonction permettant de renseigner le jour de la semaine dans une instance au choix de la structure de date et en s'appuyant sur une instance correspondant à la date actuelle.

Rappel sur la structure de date :

```
struct day { int annee; int mois; int jour; int jdls; };
typedef struct day jour;
```

Le champ `jdls` correspond au jour de la semaine, avec la convention que le lundi est associé à `1` et le dimanche à `7`. Il n'est pas forcément initialisé.

Exercices

Exercice 1 : Écrire une fonction prenant en argument un entier `n` (strictement positif mais inférieur à un milliard, ce qui n'est pas à vérifier) et renvoyant le plus grand entier égal à `n` multiplié par une puissance de deux qui soit inférieur ou égal à un milliard.

Exercice 2 : Prouver la terminaison et la correction de la fonction de l'exercice précédent.

Exercice 3 : Si dans le premier exercice il avait été écrit dix milliards au lieu d'un milliard, quel(s) aurai(en)t été le(s) problème(s) posé(s) ? (Signaler quelle aurait été l'adaptation de la fonction afin de pouvoir mettre précisément le doigt sur le problème.)

Exercice 4 : Un nombre rationnel est dit simplement normal en base b si la partie périodique de son développement en base b est d'une longueur multiple de b et que tous les chiffres utilisés dans la base b y apparaissent le même nombre de fois. Par exemple, en base 2, le nombre $\frac{1}{3}$ est simplement normal car sa partie périodique contient un 0 et un 1, et le nombre $\frac{3}{5}$ l'est aussi car sa partie périodique est 1001. Écrire une fonction prenant en argument deux entiers représentant le numérateur et le dénominateur d'un nombre rationnel et déterminant si le nombre est simplement normal en base 10.

Exercice 5 : Même exercice pour la base 2.

Exercice 6 : Même exercice avec la base en troisième argument.

Problème : Manipulations sur la représentation des nombres.

Une suite faible de Goodstein est construite de la manière suivante (et définie à partir du rang 2 pour simplifier) : u_2 est un entier non nul arbitraire, puis pour tout $n \geq 2$, on obtient u_{n+1} en partant de l'écriture de u_n en base n et en convertissant la même séquence de « chiffres » depuis la base $n+1$ puis en soustrayant 1 au résultat.

Ainsi, si $u_2 = 3$ par exemple, on écrit $3 = \overline{11^2}$, d'où u_3 s'obtient en soustrayant 1 au nombre représenté par $\overline{11^3}$ (c'est-à-dire 4) donc $u_3 = 3$ également (représenté en tant que $\overline{10^3}$), et u_4 s'obtient en soustrayant 1 au nombre représenté par $\overline{10^4}$ (c'est-à-dire 4), donc $u_4 = 3$ lui aussi, puis u_5 s'obtient en soustrayant 1 au nombre représenté par $\overline{3^5}$, donnant cette fois-ci 2, et ainsi de suite ($u_6 = 1$ puis $u_7 = 0$ et on choisit d'arrêter la suite une fois 0 atteint).

Une telle suite a une particularité intéressante : pour u_2 valant ne serait-ce qu'une dizaine, les valeurs de u_n deviennent rapidement énormes, et pourtant...

Les suites de Goodstein, qui ne seront pas abordées ici, imposent que les exposants successifs utilisent eux-mêmes la base n à chaque étape.

Théorème de Goodstein : Toute suite de Goodstein finit par atteindre 0. [Ce résultat reste évidemment vrai pour une suite faible de Goodstein.]

Question P1 : Calculer à la main les cinq premiers termes de la suite faible de Goodstein pour $u_2 = 4$. On détaillera le calcul au maximum.

On représente informatiquement dans ce problème un nombre en base b par un tableau d'entiers entre 0 et $b-1$, le « chiffre » des unités étant le premier (donc l'ordre n'est pas celui qu'on a l'habitude d'utiliser). Quant aux entiers manipulés, ils auront le type `long int` (on aurait certes pu les choisir non signés en plus). Pour information, on peut écrire `long` au lieu de `long int` pour gagner du temps.

Question P2 : Exprimer alors le tableau correspondant à 421 en base 12.

Question P3 : Écrire une fonction prenant en argument un tableau d'entiers, sa taille et la base utilisée et renvoyant l'entier ainsi représenté.

Question P4 : En vue de construire la fonction réciproque, il est bon de savoir la taille du tableau à construire. Plutôt que d'utiliser une fonction logarithme, on peut faire une boucle préliminaire, éventuellement dans une autre fonction. Écrire ainsi une fonction prenant en argument un entier et la base dans laquelle l'entier sera représenté et renvoyant le nombre de « chiffres » nécessaire.

Question P5 : Écrire à présent une fonction prenant en argument un entier et la base dans laquelle il doit être représenté et renvoyant le tableau qui représente l'entier dans la base demandée. Faire bien attention à la manière de construire la valeur de retour.

Question P6 : Écrire une fonction prenant en argument un entier et calculant successivement les termes de la suite faible de Goodstein démarrant sur cet entier. On renverra le nombre d'étapes nécessaires pour arriver à zéro (par exemple 5 pour la suite démarrant à 3, en ne considérant pas la valeur initiale comme comptant pour une étape). Le comportement est au choix si les calculs sont amenés à provoquer un dépassement arithmétique, mais une version idéale arrêterait le calcul avant qu'un dépassement ne se produise.

Question P7 : Quelle relation lie par ailleurs le nombre d'étapes nécessaires pour arriver à zéro et la plus grande valeur rencontrée ?

Question P8 : Prouver que toute suite faible de Goodstein finit par atteindre zéro.

Pour que le verso soit rentabilisé, un petit quiz pour le fun (ne rapportant pas de points).