TD1: Manipulation des tableaux en C

Programmation en C (LC4)

Semaine du 29 janvier 2007

Nous modélisons des permutations de $[0 \dots n-1]$ dans $[0 \dots n-1]$ à l'aide d'un tableau. La valeur de la case i du tableau correspond à l'image de i par la bijection. Ainsi la bijection :

$$\begin{pmatrix} 0 & 1 & 2 & 3 \\ 1 & 3 & 0 & 2 \end{pmatrix}$$

est dénotée par le tableau C :

```
int f[4] = { 1, 3, 0, 2 };
```

Exercice 1 Ecrire une fonction `int est_identite(int n, int f[])` qui teste si une permutation f de $[0 \dots n-1]$ est l'identité. Cette fonction renverra un entier valant 1 en cas de réponse positive et 0 autrement.

Exercice 2 Une inversion d'une permutation σ est un couple (i, j) d'entiers tels que $i < j$ et $\sigma(i) > \sigma(j)$. Ecrire une fonction `int nombre_inversions(int n, int f[])` qui calcule le nombre d'inversions d'une permutation.

Exercice 3 Ecrire la fonction
`void compose_permutations(int n, int perm1[], int perm2[], int perm_composee[])`
qui calcule la composée σ, de σ_1 et σ_2, telle que $\sigma = \sigma_1 \circ \sigma_2$ ($\sigma(i) = \sigma_1(\sigma_2(i))$).

Exercice 4 Ecrire un fonction `void inverse_permutation(int n, int perm[], int perm_inverse[])` qui calcule l'inverse σ^{-1} de la permutation σ, tel que $\sigma^{-1} \circ \sigma = Id$.

Exercice 5 Ecrire une fonction `void permutation_aleatoire(int n, int f[])` générant une permutation aléatoire. On utilisera la fonction C suivante :

```
/* La ligne suivante est a rajouter au debut de votre fichier,
   si elle n'y est pas deja. */
#include <stdlib.h>

// Cette fonction renvoie un entier pris aleatoirement entre 0 et n - 1.
int entier_hasard (int n) {
   return (rand () % n);
}
```

Exercice 6 Ecrire une fonction `void applique_permutation(int n, int perm[], int src[], int dst[])` qui modifie `dst` tel que pour tout $i < n$, `dst[perm[i]] = src[i]`.

Exercice 7 Ecrire une fonction `int est_trie(int n, int t[])` qui teste si un tableau est trié.

Exercice 8 Ecrire une fonction `void un_singe_trie(int n, int non_trie[], int trie[], int perm[])` qui applique une permutation aléatoire au tableau `non_trie` jusqu'à en obtenir une version triée dans le tableau `trie`.

Exercice 9 Ecrire une fonction `void genere_permutation_de_tri(int n, int t[], int perm[])` qui calcule la permutation `perm` que l'on doit appliquer au tableau `t` pour le trier.