TP 6 : Révisions

Programmation en C (LC4)

Semaine du 5 mars 2007

▶ Exercice 1

```
int compte_zero(int n, int *tab) {
    int i, c = 0;
    for (i = 0; i < n; i++)
        if (tab[i] == 0)
            c++;
    return c;
}
```

▶ Exercice 2

```
#include <stdlib.h>

int *copie(int n, int *tab) {
    int i;
    int *res = malloc(n * sizeof(int));
    for (i = 0; i < n; i++)
        res[i] = tab[i];
    return res;
}
```

```
#include <stdlib.h>
#include <string.h>

int *copie2(int n, int *tab) {
    int *res = malloc(n * sizeof(int));
    return memcpy(res, tab, n * sizeof(int));
}
```

▶ Exercice 3

```
#include <stdlib.h>

int *fusion(int n1, int *tab1, int n2, int *tab2) {
    int *res = malloc((n1 + n2) * sizeof(int));
    int i1 = 0, i2 = 0;
    while (i1 < n1 && i2 < n2)
        if (tab1[i1] < tab2[i2]) {
            res[i1 + i2] = tab1[i1];
            i1++;
        } else {
            res[i1 + i2] = tab2[i2];
            i2++;
        }
```

```
    if (i1 < n1)
        for (; i1 < n1; i1++)
            res[i1 + i2] = tab1[i1];
    if (i2 < n2)
        for (; i2 < n2; i2++)
            res[i1 + i2] = tab2[i2];
    return res;
}
```

▶ Exercice 4

```
#include <stdlib.h>

char *copie_chaine(char *s) {
    int i, l = 0;
    char *res;
    for (i = 0; s[i] != '\0'; i++)
        l++;
    res = malloc(l + 1);
    for (i = 0; i <= l; i++) /* on copie aussi le '\0' final: i <= l */
        res[i] = s[i];
    return res;
}

#include <stdlib.h>
#include <string.h>

char *copie_chaine2(char *s) {
    char *res = malloc(strlen(s) + 1);
    return strcpy(res, s);
}
```

▶ Exercice 5

```
#include <stdlib.h>

char *chaine_binaire(char zero, char un, int n) {
    int i = 31;
    char *s = malloc(32 + 1);
    while (i >= 0) {
        s[31 - i] = (n & (1 << i)) ? un : zero;
        i--;
    }
    s[32] = '\0';
    return s;
}
```

▶ Exercice 6

```
#include <stdlib.h>

int **tri(int n, int *tab) {
    int i, j;
    int **res = malloc(n * sizeof(int *));
    for (i = 0; i < n; i++) {
        res[i] = NULL;
```

```
        for (j = 0; j < n; j++)
            if (tab[j] > tab[i])
                /* on utilise l'évaluation paresseuse du || pour ne
                 * déréférencer res[i] que s'il est non NULL (s'il est NULL,
                 * le || est de toute manière évalué comme vrai)
                 */
                if (res[i] == NULL || *res[i] > tab[j])
                    res[i] = &tab[j];
    }
    return res;
}
```