TD 3 : Pointeurs et tableaux

Programmation en C (LC4)

Semaine du 12 février 2007

▶ Exercice 1

```
#include <stdio.h>

void affichage_binaire(unsigned int n)
{
    int i;
 /* printf("%d en binaire : ", n); */
    for (i = 31; i >= 0; i--)
        printf("%d", (n >> i ) & 1);
    printf("\n");
}
```

▶ Exercice 2

programme	a	b	c	p1, *p1	p2, *p2
`int a, b, c, *p1, *p2;`	?	?	?	-, ?	-, ?
`a = 1, b = 2, c = 3;`	1	2	3	-, ?	-, ?
`p1 = &a, p2 = &c;`	1	2	3	`&a`, 1	`&c`, 3
`*p1 = (*p2)++;`	3	2	4	`&a`, 3	`&c`, 4
`p1 = p2;`	3	2	4	`&c`, 4	`&c`, 4
`p2 = &b;`	3	2	4	`&c`, 4	`&b`, 2
`*p1 -= *p2;`	3	2	2	`&c`, 2	`&b`, 2
`++*p2;`	3	3	2	`&c`, 2	`&b`, 3
`*p1 *= *p2;`	3	3	6	`&c`, 6	`&b`, 3
`a = ++*p2 * *p1;`	24	4	6	`&c`, 6	`&b`, 4
`p1 = &a;`	24	4	6	`&a`, 24	`&b`, 4
`*p2 = *p1 /= *p2;`	6	6	6	`&a`, 6	`&b`, 6

```
#include <stdio.h>

int main(void) {
    int n = 0;

    int a, b, c, *p1, *p2;
    printf("%2d:␣␣a:␣%d,␣&a:␣%p,␣b:␣%d,␣&b:␣%p,␣c:␣%d,␣&c:␣%p\n␣␣"
           "␣␣␣p1:␣%p,␣*p1:␣−,␣p2:␣%p,␣*p2:␣−\n",
           n, a, &a, b, &b, c, &c, p1, p2); n++;

    a = 1, b = 2, c = 3;
    printf("%2d:␣␣a:␣%d,␣&a:␣%p,␣b:␣%d,␣&b:␣%p,␣c:␣%d,␣&c:␣%p\n␣␣"
           "␣␣␣p1:␣%p,␣*p1:␣−,␣p2:␣%p,␣*p2:␣−\n",
           n, a, &a, b, &b, c, &c, p1, p2); n++;
```

```
    p1 = &a, p2 = &c;
    printf("%2d:␣␣a:␣%d,␣&a:␣%p,␣b:␣%d,␣&b:␣%p,␣c:␣%d,␣&c:␣%p\n␣␣"
              "␣␣␣p1:␣%p,␣*p1:␣%d,␣p2:␣%p,␣*p2:␣%d\n",
           n, a, &a, b, &b, c, &c, p1, *p1, p2, *p2); n++;

    *p1 = (*p2)++;
    printf("%2d:␣␣a:␣%d,␣&a:␣%p,␣b:␣%d,␣&b:␣%p,␣c:␣%d,␣&c:␣%p\n␣␣"
              "␣␣␣p1:␣%p,␣*p1:␣%d,␣p2:␣%p,␣*p2:␣%d\n",
           n, a, &a, b, &b, c, &c, p1, *p1, p2, *p2); n++;

    p1 = p2;
    printf("%2d:␣␣a:␣%d,␣&a:␣%p,␣b:␣%d,␣&b:␣%p,␣c:␣%d,␣&c:␣%p\n␣␣"
              "␣␣␣p1:␣%p,␣*p1:␣%d,␣p2:␣%p,␣*p2:␣%d\n",
           n, a, &a, b, &b, c, &c, p1, *p1, p2, *p2); n++;

    p2 = &b;
    printf("%2d:␣␣a:␣%d,␣&a:␣%p,␣b:␣%d,␣&b:␣%p,␣c:␣%d,␣&c:␣%p\n␣␣"
              "␣␣␣p1:␣%p,␣*p1:␣%d,␣p2:␣%p,␣*p2:␣%d\n",
           n, a, &a, b, &b, c, &c, p1, *p1, p2, *p2); n++;

    *p1 -= *p2;
    printf("%2d:␣␣a:␣%d,␣&a:␣%p,␣b:␣%d,␣&b:␣%p,␣c:␣%d,␣&c:␣%p\n␣␣"
              "␣␣␣p1:␣%p,␣*p1:␣%d,␣p2:␣%p,␣*p2:␣%d\n",
           n, a, &a, b, &b, c, &c, p1, *p1, p2, *p2); n++;

    ++*p2;
    printf("%2d:␣␣a:␣%d,␣&a:␣%p,␣b:␣%d,␣&b:␣%p,␣c:␣%d,␣&c:␣%p\n␣␣"
              "␣␣␣p1:␣%p,␣*p1:␣%d,␣p2:␣%p,␣*p2:␣%d\n",
           n, a, &a, b, &b, c, &c, p1, *p1, p2, *p2); n++;

    *p1 *= *p2;
    printf("%2d:␣␣a:␣%d,␣&a:␣%p,␣b:␣%d,␣&b:␣%p,␣c:␣%d,␣&c:␣%p\n␣␣"
              "␣␣␣p1:␣%p,␣*p1:␣%d,␣p2:␣%p,␣*p2:␣%d\n",
           n, a, &a, b, &b, c, &c, p1, *p1, p2, *p2); n++;

    a = ++*p2 * *p1;
    printf("%2d:␣␣a:␣%d,␣&a:␣%p,␣b:␣%d,␣&b:␣%p,␣c:␣%d,␣&c:␣%p\n␣␣"
              "␣␣␣p1:␣%p,␣*p1:␣%d,␣p2:␣%p,␣*p2:␣%d\n",
           n, a, &a, b, &b, c, &c, p1, *p1, p2, *p2); n++;

    p1 = &a;
    printf("%2d:␣␣a:␣%d,␣&a:␣%p,␣b:␣%d,␣&b:␣%p,␣c:␣%d,␣&c:␣%p\n␣␣"
              "␣␣␣p1:␣%p,␣*p1:␣%d,␣p2:␣%p,␣*p2:␣%d\n",
           n, a, &a, b, &b, c, &c, p1, *p1, p2, *p2); n++;

    *p2 = *p1 /= *p2;
    printf("%2d:␣␣a:␣%d,␣&a:␣%p,␣b:␣%d,␣&b:␣%p,␣c:␣%d,␣&c:␣%p\n␣␣"
              "␣␣␣p1:␣%p,␣*p1:␣%d,␣p2:␣%p,␣*p2:␣%d\n",
           n, a, &a, b, &b, c, &c, p1, *p1, p2, *p2); n++;

    return 0;
}
```

Résultat de l'exécution :

```
 0:  a: 0, &a: 0023FF68, b: 0, &b: 0023FF64, c: 2009116333, &c: 0023FF60
```

```
     p1: 00000008, *p1: -, p2: 00032C38, *p2: -
 1:  a: 1, &a: 0023FF68, b: 2, &b: 0023FF64, c: 3, &c: 0023FF60
     p1: 00000008, *p1: -, p2: 00032C38, *p2: -
 2:  a: 1, &a: 0023FF68, b: 2, &b: 0023FF64, c: 3, &c: 0023FF60
     p1: 0023FF68, *p1: 1, p2: 0023FF60, *p2: 3
 3:  a: 3, &a: 0023FF68, b: 2, &b: 0023FF64, c: 4, &c: 0023FF60
     p1: 0023FF68, *p1: 3, p2: 0023FF60, *p2: 4
 4:  a: 3, &a: 0023FF68, b: 2, &b: 0023FF64, c: 4, &c: 0023FF60
     p1: 0023FF60, *p1: 4, p2: 0023FF60, *p2: 4
 5:  a: 3, &a: 0023FF68, b: 2, &b: 0023FF64, c: 4, &c: 0023FF60
     p1: 0023FF60, *p1: 4, p2: 0023FF64, *p2: 2
 6:  a: 3, &a: 0023FF68, b: 2, &b: 0023FF64, c: 2, &c: 0023FF60
     p1: 0023FF60, *p1: 2, p2: 0023FF64, *p2: 2
 7:  a: 3, &a: 0023FF68, b: 3, &b: 0023FF64, c: 2, &c: 0023FF60
     p1: 0023FF60, *p1: 2, p2: 0023FF64, *p2: 3
 8:  a: 3, &a: 0023FF68, b: 3, &b: 0023FF64, c: 6, &c: 0023FF60
     p1: 0023FF60, *p1: 6, p2: 0023FF64, *p2: 3
 9:  a: 24, &a: 0023FF68, b: 4, &b: 0023FF64, c: 6, &c: 0023FF60
     p1: 0023FF60, *p1: 6, p2: 0023FF64, *p2: 4
10:  a: 24, &a: 0023FF68, b: 4, &b: 0023FF64, c: 6, &c: 0023FF60
     p1: 0023FF68, *p1: 24, p2: 0023FF64, *p2: 4
11:  a: 6, &a: 0023FF68, b: 6, &b: 0023FF64, c: 6, &c: 0023FF60
     p1: 0023FF68, *p1: 6, p2: 0023FF64, *p2: 6
```

▶ **Exercice 3**

```
#include <stdio.h>

void echange(int *a, int *b) {
    int tmp = *a;
    *a = *b;
    *b = tmp;
}

int main(void) {
    int i = 4, j = 2;
    printf("i=%d, j=%d\n", i, j);
    echange(&i, &j);
    printf("i=%d, j=%d\n", i, j);
    return 0;
}
```

▶ **Exercice 4**

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

void echange_tab(int **ptr_tab1, int **ptr_tab2) {
    int *t;
    t = *ptr_tab1;
    *ptr_tab1 = *ptr_tab2;
    *ptr_tab2 = t;
}

void affiche_tab(int n, int *tab)
{
```

```
    int i;
    for (i = 0; i < n; i++)
        printf("%d ", tab[i]);
    printf("\n");
}

int main(void) {
    int *tab1 = malloc(3 * sizeof(int));
    int *tab2 = malloc(3 * sizeof(int));
    tab1[0] = 1; tab1[1] = 6; tab1[2] = 2;
    tab2[0] = 5; tab2[1] = 3; tab2[2] = 9;
    printf("tab1: ");
    affiche_tab(tab, 3);
    printf("tab2: ");
    affiche_tab(tab, 3);
    echange_tab(&tab1, &tab2);
    printf("tab1: ");
    affiche_tab(tab, 3);
    printf("tab2: ");
    affiche_tab(tab, 3);
    free(tab1);
    free(tab2);
    return 0;
}
```

► **Exercice 5**

```
#include <stdlib.h>

int *concat_tab(int n1, int tab1[], int n2, int tab2[]) {
    int i;
    int *resultat = malloc((n1 + n2) * sizeof(int));
    for (i = 0; i < n1; i++)
        resultat[i] = tab1[i];
    for (i = n; i < n1 + n2; i++)
        resultat[i] = tab2[i-n1];
    return resultat;
}
```

► **Exercice 6**

```
#include <stdlib.h>

char *concat_string(char *str1, char *str2)
{
    int len;
    char *t, *resultat;
    t = str1;
    while (*t != '\0')
        t++;
    len = t - str1;
    t = str2;
    while (*t != '\0')
        t++;
    len += t - str2;
    resultat = t = malloc(len + 1);
```

```
    while (*str1 != '\0') {
        *t = *str1;
        t++;
        str1++;
    }
    while (*str2 != '\0') {
        *t = *str2;
        t++;
        str2++;
    }
    *t = '\0';
    return resultat;
}
```

► **Exercice 7**

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

struct livre *init_bib(int n) {
    int i;
    struct livre *resultat = malloc(n * sizeof(struct livre));
    for (i = 0; i < n; i++) {
        resultat[i].titre[0] = '\0';
        resultat[i].cote = 0;
        resultat[i].prix = 0;
    }
    return resultat;
}

void affiche_bib(int n, struct livre *bib) {
    for (i = 0; i < n; i++)
        printf("titre: %s, cote: %d, prix: %d\n",
                bib[i].titre, bib[i].cote, bib[i].prix);
}

int main(void) {
    struct livre *bib = init_bib(2);
    bib[0].titre[0] = 'L';
    bib[0].titre[1] = 'i';
    bib[0].titre[2] = 'v';
    bib[0].titre[3] = 'r';
    bib[0].titre[4] = 'e';
    bib[0].titre[5] = '1';
    bib[0].titre[6] = '\0';
    bib[0].cote = 41;
    bib[0].prix = 20;
    bib[0].titre[0] = 'L';
    bib[0].titre[1] = 'i';
    bib[0].titre[2] = 'v';
    bib[0].titre[3] = 'r';
    bib[0].titre[4] = 'e';
    bib[0].titre[5] = '2';
    bib[0].titre[6] = '\0';
    bib[1].cote = 42;
    bib[1].prix = 10;
```

```
    affiche_bib(2, bib);
    free(bib);
    return 0;
}
```

► **Exercice 8**

```
void echange_livre(int i, int j, struct livre *bib) {
    struct livre tmp;
    tmp = bib[i];
    bib[i] = bib[j];
    bib[j] = tmp;
}
```