

# TD 3 : pointeurs, tableaux

Programmation en C (LC4)

Semaine du 11 février 2008

## ► Exercice 1

```
void affichage_binaire(int m){
    int i=0;
    int t=1;
    printf("%m_en_binaire_",m);
    for(i=31;i>=0;i--){
        printf("%i", (m>>i)&t);
    }
    printf("\n");
}
```

## ► Exercice 2

programme	a	b	c	p1, *p1	p2, *p2
int a, b, c, *p1, *p2;	x	x	x	x	x
a = 1, b = 2, c = 3;	1	2	3	x	x
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

```
int main()
{
    int a, b, c, *p1, *p2;
    a = 1, b = 2, c = 3;
    p1 = &a, p2 = &c;
    printf("0:a:%i_adresse_a:%i_b:%i_adresse_b:%i_c:%i_adresse_c:%i_*p1:%i_*p2:%i\n",a,&a,b,&b,c,&c,*p1,p1,*p2,p2);

    *p1 = (*p2)++;
```

```

printf("1:_a:%i_adresse_a:_:i,_b:%i,adresse_b:_:i,_c:%i,adresse_c:_:i,_*p1:%i,_p1:%i,
*p2:%i_p1:%i_\n",a,&a,b,&b,c,&c,*p1,p1,*p2,p2);

p1 = p2;
printf("2:_a:%i_adresse_a:_:i,_b:%i,adresse_b:_:i,_c:%i,adresse_c:_:i,_*p1:%i,_p1:%i,
*p2:%i_p1:%i_\n",a,&a,b,&b,c,&c,*p1,p1,*p2,p2);

p2 = &b;
printf("3:_a:%i_adresse_a:_:i,_b:%i,adresse_b:_:i,_c:%i,adresse_c:_:i,_*p1:%i,_p1:%i,
*p2:%i_p1:%i_\n",a,&a,b,&b,c,&c,*p1,p1,*p2,p2);

*p1 -= *p2;
printf("4:_a:%i_adresse_a:_:i,_b:%i,adresse_b:_:i,_c:%i,adresse_c:_:i,_*p1:%i,_p1:%i,
*p2:%i_p1:%i_\n",a,&a,b,&b,c,&c,*p1,p1,*p2,p2);

++*p2;
printf("5:_a:%i_adresse_a:_:i,_b:%i,adresse_b:_:i,_c:%i,adresse_c:_:i,_*p1:%i,_p1:%i,
*p2:%i_p1:%i_\n",a,&a,b,&b,c,&c,*p1,p1,*p2,p2);

*p1 *= *p2;
printf("6:_a:%i_adresse_a:_:i,_b:%i,adresse_b:_:i,_c:%i,adresse_c:_:i,_*p1:%i,_p1:%i,
*p2:%i_p1:%i_\n",a,&a,b,&b,c,&c,*p1,p1,*p2,p2);

a = ++*p2**p1;
printf("7:_a:%i_adresse_a:_:i,_b:%i,adresse_b:_:i,_c:%i,adresse_c:_:i,_*p1:%i,_p1:%i,
*p2:%i_p1:%i_\n",a,&a,b,&b,c,&c,*p1,p1,*p2,p2);

p1 = &a;
printf("8:_a:%i_adresse_a:_:i,_b:%i,adresse_b:_:i,_c:%i,adresse_c:_:i,_*p1:%i,_p1:%i,
*p2:%i_p1:%i_\n",a,&a,b,&b,c,&c,*p1,p1,*p2,p2);

*p2 = *p1 /= *p2;
printf("9:_a:%i_adresse_a:_:i,_b:%i,adresse_b:_:i,_c:%i,adresse_c:_:i,_*p1:%i,_p1:%i,
*p2:%i_p1:%i_\n",a,&a,b,&b,c,&c,*p1,p1,*p2,p2);

return 1;
}

```

Résultat de l'exécution :

0: a:1 adresse\_a : -846253428 , b:2,adresse\_b : -846253432, c:3,adresse\_c : -846253436,  
\*p1:1, p1:-846253428, \*p2:3 p1:-846253436

1: a:3 adresse\_a : -846253428 , b:2,adresse\_b : -846253432, c:4,adresse\_c : -846253436,  
\*p1:3, p1:-846253428, \*p2:4 p1:-846253436

2: a:3 adresse\_a : -846253428 , b:2,adresse\_b : -846253432, c:4,adresse\_c : -846253436,  
\*p1:4, p1:-846253436, \*p2:4 p1:-846253436

3: a:3 adresse\_a : -846253428 , b:2,adresse\_b : -846253432, c:4,adresse\_c : -846253436,  
\*p1:4, p1:-846253436, \*p2:2 p1:-846253432

4: a:3 adresse\_a : -846253428 , b:2,adresse\_b : -846253432, c:2,adresse\_c : -846253436,  
\*p1:2, p1:-846253436, \*p2:2 p1:-846253432

5: a:3 adresse\_a : -846253428 , b:3,adresse\_b : -846253432, c:2,adresse\_c : -846253436,

\*p1:2, p1:-846253436, \*p2:3 p1:-846253432

6: a:3 adresse\_a :-846253428 , b:3,adresse\_b :-846253432, c:6,adresse\_c :-846253436,  
\*p1:6, p1:-846253436, \*p2:3 p1:-846253432

7: a:24 adresse\_a :-846253428 , b:4,adresse\_b :-846253432, c:6,adresse\_c :-846253436,  
\*p1:6, p1:-846253436, \*p2:4 p1:-846253432

8: a:24 adresse\_a :-846253428 , b:4,adresse\_b :-846253432, c:6,adresse\_c :-846253436,  
\*p1:24, p1:-846253428, \*p2:4 p1:-846253432

9: a:6 adresse\_a :-846253428 , b:6,adresse\_b :-846253432, c:6,adresse\_c :-846253436,  
\*p1:6, p1:-846253428, \*p2:6 p1:-846253432

### ► Exercise 3

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

```
int main() {
    int ec1=4, ec2=2;
    printf("%i,_%i_\n",ec1, ec2);
    exchange(&ec1, &ec2);
    printf("%i,_%i_\n",ec1, ec2);
}
```

### ► Exercise 4

```
void exchange_tab(int** t, int** r){
    int * tmp;
    tmp = *t;

    *t = *r;
    *r = tmp;
}
```

```
void affiche_vecteur(int * vecteur, int dimension)
{
    int i;
    printf("_");
    for(i=0;i<dimension;i++)
        printf("%d_",vecteur[i]);
    printf("\n");
}
```

```
int main() {
    int * t=malloc(3*sizeof(int));
    int * r=malloc(3*sizeof(int));
    int * tmp;
    t[0]=1;
```

```

t[1]=6;
t[2]=2;
r[0]=5;
r[1]=3;
r[2]=9;
echange_tab(&t, &r);
printf("t_: \n");
affiche_vecteur(t,3);
printf("r_: \n");
affiche_vecteur(r,3);

return 1;
}

```

► **Exercise 5**

```

int* concat_tab(int n, int t[], int m, int r[]){
    int i;
    int * res = malloc((n+m)*sizeof(int));
    for(i=0;i<n;i++)
        res[i]=t[i];
    for(i=n;i<m+n;i++)
        res[i]=r[i-n];
    return res;
}

```

► **Exercise 6**

```

char *concat_string(char *s1, char *s2)
{
    int len;
    char *t, *s, *r;

    t = s1;
    while (*t != '\0') {
        t++;
    }
    len = t - s1;

    t = s2;
    while (*t != '\0') {
        t++;
    }
    len += t - s2;

    r = s = malloc(len + 1);
    while (*s1 != '\0') {
        *r = *s1;
        r++;
        s1++;
    }
    while (*s2 != '\0') {
        *r = *s2;
        r++;
        s2++;
    }
}

```

```

    }
    *r = '\0';

    return s;
}

```

► **Exercice 7**

```

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

```

```

void affiche_bib(int n,struct lvr * bib){
    int i=0;
    for(;i<n;i++){
        printf("titre_:_%s,_cote_:_%i,_prix_:_%i_\n",bib[i].titre, bib[i].cote,bib[i].prix);
    }
}

```

```

int main(void){
    struct lvr* bib = init(2);
    aff_lvr(bib[0]);
    aff_lvr(bib[1]);
}

```

► **Exercice 8**

```

void echange_lvr(int i, int j, struct lvr * bib){
    struct lvr tmp;
    tmp = bib[i];
    bib[i]=bib[j];
    bib[j]=tmp;
}

```