

## TD7 - Algorithmique et C - Pointeurs, paramètres variables

### Objectifs

Savoir manipuler le concept de pointeur, de variable modifiable. Savoir dessiner les schémas d'exécution de programme. Source : FS pour GIS

### 1 Rappels

Soit  $a$  une variable entière.  $\&a$  est l'adresse où est rangée  $a$  en mémoire.

Soit  $p$  une variable de type pointeur d'entier. La valeur de  $p$  est l'adresse d'une zone mémoire contenant un entier.  $*p$  est la valeur de l'entier rangé à l'adresse  $p$ .

**Exemple :**

int	<i>objet</i>	<i>adresse</i>	<i>valeur</i>	$\&a = \langle 1A00, \text{int} \rangle$
	$a$	1A00		
int *	$p$	1A08		

Si  $p = \&a$  alors  $*p = a$

int	<i>objet</i>	<i>adresse</i>	<i>valeur</i>	$p = \langle 1A00, \text{int} \rangle$
	$a$	1A00		
int *	$p$	1A08	1A00	

**Remarque :**  $*\&a = a$  et  $\&*p = p$

### 2 Exercices

EXERCICE 1 *Quelles sont les valeurs affichées à l'écran par le programme suivant et pourquoi ?*

```
int main() {
    int *p1,*p2,*p3;
    int n,m,k;
    n=21; m=12;
    p1=&m; p2=&n;
    printf("%d %d %d %d",*p1,*p2,m,n);
    p3=p1; p1=p2; p2=p3;
    printf("%d %d %d %d",*p1,*p2,m,n);
    k=*p1; *p1=*p2; *p2=k;
    printf("%d %d %d %d",*p1,*p2,m,n);
    return 0;
}
```

EXERCICE 2 Pour chacun des deux codes ci-dessus, faire le schéma d'exécution :

<pre> int algo1(int a, int b){     int tmp = a+2*b     return tmp }  int main(){     int a, int j;     j=34 ; a=2;     int resu = algo1(j,a);     return 0; } </pre>	<pre> void algo2(char* s,int b){     b = b+1;     *s = 'c'; }  int main(){     int x=1; char u;     algo2(&amp;u,x);      printf("%c, %d",u,x); } </pre>
--	--

### 3 Exercices

EXERCICE 3 Compléter le tableau ci-dessous en indiquant les valeurs des différentes variables à la fin de l'exécution de chaque instruction. Faire un dessin de mémoire simplifié.

Instructions	<i>m</i>	<i>n</i>	<i>p1</i>	<i>*p1</i>	<i>p2</i>	<i>*p2</i>
int m, n, *p1, *p2;	?	?	?	?	?	?
m=2; n=8;						
p1 = &m; p2 = &m;						
*p2 = *p2 + *p1;						
p2 = &n;						
*p1 *= *p2;						

EXERCICE 4 Écrire un algorithme `max_min_2` qui range le maximum de deux entiers *a* et *b* dans *a* et leur minimum dans *b* (cf TD4).

Écrire le sous-programme *C* correspondant et l'appel de ce sous-programme par le `main()`.

EXERCICE 5 Écrire un algorithme `conversion_minutes` qui détermine le résultat de la conversion d'un nombre donné d'heures (type entier) en minutes.

Écrire le sous-programme *C* correspondant et l'appel de ce sous-programme par le `main()`.

EXERCICE 6 Écrire un algorithme `conversion_minutes_secondes` qui détermine le résultat de la conversion d'un nombre donné d'heures (type entier) en minutes et le résultat de la conversion de ce même nombre d'heures en secondes.

Écrire le sous-programme *C* correspondant et l'appel de ce sous-programme par le `main()`.