Travaux Dirigés d'Algorithmique n°2

Cours d'Informatique de Deuxième Année

—Deug SSM M3-MI—

Ce deuxième TD constitue la suite de l'introduction au langage *C*. Nos objectifs sont les suivants :

- appels de fonctions, notion de valeur et notion d'adresse ;
- tableaux d'entiers, tri par recherche du minimum.

▸ **Exercice 1.** (**passage par valeur et passage par adresse**) *Cet exercice met en évidence la différence entre les notions de valeur et d'adresse. Il demande beaucoup d'attention mais les notions introduites sont en fait très simples et intuitives. Nous illustrons ces idées par l'exemple suivant qu'il faut commenter :*

```
#include <stdio.h>
int biz(int i, int * j, int k)
{ *j=4; i=5; k+=10;
  return k;
}

void main(void)
{ int * a, b=3, c;
  a=&b; b=2;
  printf("La valeur a l'adresse %ld est %d\n", a, *a);
  printf("L'entier b est stocke a l'adresse %ld\n", &b);
  c=biz(b, a, *a);
  printf("*a=%d  b=%d  c=%d\n",*a,b,c);
}
```

▸ **Exercice 2.** (**réalisation d'un swap de deux variables**) *Vérifions nos connaissances en matière de passage de paramètres à une fonction, en commentant le résultat de l'exécution du programme suivant :*

```
#include <stdio.h>
void swap(int a, int b)
{ int tmp=a;
  a=b; b=tmp;
  printf("Dans la fonction : a = %d et b = %d\n", a, b);
}
```

```
void main(void)
{ int a=5,b=3;
  printf("Avant de rentrer dans la fonction  : a = %d et b = %d\n", a, b);
  swap(a, b);
  printf("Apres etre rentre dans la fonction : a = %d et b = %d\n", a, b);
}
```

Comment modifier ce programme afin qu'il réalise vraiment l'échange des valeurs des deux variables ?

▸ **Exercice 3.** (**tableau d'entiers**) *Ecrire un programme déclarant un tableau de 50 entiers (*`int`*) et remplissant ce tableau avec* **au plus 50 entiers** *lus au clavier. On arrêtera de lire au clavier lorsque l'on rencontrera un entier négatif*[1]*. Le programme devra alors afficher le tableau en le parcourant de la droite vers la gauche.*

▸ **Exercice 4.** (**tableau en paramètre d'une fonction**) *Refaire le même exercice en utilisant cette fois une fonction pour remplir le tableau d'entiers. Celui-ci sera passé en argument à la fonction, laquelle retournera le nombre d'entiers effectivement lus.*

▸ **Exercice 5.** (**tri par recherche du minimum**) *On se propose dans cet exercice de réaliser une fonction de tri en ordre croissant (par exemple) d'un tableau d'entiers. On divise le problème en plusieurs étapes.*

1. *Ecrire une fonction de nom* `MinTab` *prenant en paramètre un tableau d'entiers ainsi que sa taille, et retournant la* **position** *du minimum des valeurs du tableau ;*
2. *Maintenant que l'on sait trouver la position du minimum des n valeurs du tableau, on peut donner l'algorithme de tri par recherche du minimum. En effet, le minimum occupera la première case du tableau trié. Ayant amené la dite valeur à sa place, il restera à trier la fin du tableau, c'est-à-dire $n-1$ valeurs. Notre algorithme s'arrêtera dès lors qu'il n'y aura plus qu'une valeur à trier.*
   (a) *donner en pseudo-langage, l'algorithme de tri par recherche du minimum d'un tableau de n éléments ;*
   (b) *donner une fonction de nom* **TriMin** *implémentant cet algorithme ;*
   (c) *en déduire le programme complet de saisie d'un tableau d'entiers, de tri puis d'affichage du résultat.*
   (d) *Quelle est la complexité en temps de cet algorithme de tri ?*

[1] Ce dernier entier ne devra pas être écrit dans le tableau.