Travaux Dirigés sur le langage C n°.2

Cours d'Informatique de Deuxième Année

—Deug SSM M3—

Ce deuxième *TD* présente des notions un peu plus évoluées. Les objectifs sont les suivants :

- Mise au point sur les fonctions,
- Notion de *valeur*, notion d'*adresse*,
- Tableaux, introduction aux chaînes de caractères,

Fonctions, valeurs et adresses

▸ **Exercice 1**. Cet exercice met en évidence la différence entre la notion de *valeur* et d'*adresse*. Il demande beaucoup d'attention mais les notions introduites sont en fait très simples et intuitives. On illustre ces idées par l'exemple suivant que l'on demande de 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**. On considère le 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:  a = %d et b = %d\n",a,b);
  swap(a, b);
  printf("Apres:  a = %d et b = %d\n",a,b);
}
```

Quel est le résultat de l'exécution de ce programme ? Comment le modifier pour qu'il réalise vraiment le swap des variables `a` et `b` ?

Les tableaux...

▸ **Exercice 3**. Ecrire un programme déclarant un tableau de caractères et remplissant ce tableau avec **au plus 50 caractères** lus au clavier. On arrêtera de lire au clavier quand on verra le caractère '?'[1]. Le programme devra alors afficher le tableau.

▸ **Exercice 4**. Refaire le même exercice mais cette fois en utilisant une fonction pour saisir le tableau de caractères. Celui-çi sera passé en argument à la fonction. La fonction retournera le nombre de caractères lus.

▸ **Exercice 5**. Ecrire une fonction prenant en argument deux tableaux de caractères `t1` et `t2` et un entier `l1` préçisant le nombre de caractères contenus dans le tableau `t1`, qui recopie le tableau t1 dans le tableau t2 à l'envers.

▸ **Exercice 6**. En déduire un programme complet qui demande à l'utilisateur de rentrer une chaîne de caractères et qui teste si cette suite de caractères est un *palindrome*. On rappelle qu'un palindrome est un mot dont les lectures de gauche à droite et de droite à gauche sont les mêmes : en particulier, tout mot de 1 lettre est un palindrome. Les mots "laval", "non" ou encore "z" sont des palindromes mais on trouve aussi des exemples plus conséquents comme par exemple "Ésope reste élu par cette crapule et se repose" ou "Noël a trop par rapport à Léon". Cependant, on ne demande pas dans cet exercice de faire un test sur les "*phrases palindromes*" mais uniquement sur des "*mots palindromes*". Le programme devra aussi afficher la chaîne de caractères.

▸ **Exercice 7**. Une chaîne de caractères est définie en *C* par une suite de caractères se terminant par le caractère '\0'. C'est ainsi qu'un tableau de caractères "sera vu" en *C* comme une chaîne de caractères dès lors qu'il se termine par un '\0'. On demande dans cet exercice de tester si une chaîne de caractères entrée au clavier est un palindrome sans recopie de cette chaîne.

▸ **Exercice 8**. Ecrire une fonction de saisie d'un tableau de `char` passé en argument. On entrera des chiffres (au plus 10) jusqu'à ce que l'utilisateur entre le caractère '='. La fonction devra alors retourner l'entier en base 10 correspondant à cette suite de chiffres.

Exercices supplémentaires

▸ **Exercice 9**. Ecrire une fonction de **tri par sélection du maximum**. Pour trier un tableau `t` de taille `n`, on recherche dans le tableau la plus grande valeur. Soit `m` cette valeur qui est à la position `pm`, i.e. `t[pm]` vaut `m`. On "*swap*" alors dans le tableau les valeurs aux places `pm` et `n` positionnant ainsi la valeur `m` à sa place. On recommence cet algorithme sur les `n-1` premières valeurs du tableau.

▸ **Exercice 10**. Déduire de l'algorithme précédent la complexité[2] en temps de ce tri.

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

[2] On regardera le nombre de comparaisons effectuées.