Langage C – TD4 – Fonctions et Pointeurs

J. SEINTURIER / Université de Toulon (julien.seinturier@univ-tln.fr / http://www.seinturier.fr)

1. Fonctions

Exercice 1.1.
Expliquer ce qu'est une fonction en langage C. Comment une fonction est intégrée à un programme ?

Exercice 1.2.
Analyser le programme suivant (donner la séquence des affichages produits) :

```
#include <stdio.h>

int f1(int i){return i+1;}

int f2(int i){return i++;}

int f3(int i) {
  printf("f3 : %d\n",i==0);
  return i;
}

int f4(int i) {
  printf("f4 : %d\n",i=0);
  return i;
}

int main(){
  int a,b;
  a=f1(0);
  b=f2(1);
  printf("1 : a=%d, b=%d\n",a,b);
  a=f3(a);
  b=f4(a);
  printf("2 : a=%d, b=%d\n",a,b);
}
```

Exercice 1.3.
Comment sont passés les paramètres du programme principal à une fonction ? Quelles limitations cela engendre-t-il ?

Exercice 1.4.
Ecrire une fonction int `compare(int x, int y)` qui pour deux nombres *x* et *y* passés en paramètre renvoie 1 si *x* est le plus grand nombre, -1 si *y* est le plus grand nombre et 0 si les deux nombres sont égaux.

Exercice 1.5.
Ecrire une fonction float `valeurAbsolue(float x)` qui renvoie la valeur absolue de x.

Exercice 1.6.
Ecrire une fonction `int lePlusProcheDe10(int x, int y)` qui renvoie le nombre x ou y qui est le plus proche de 10. En cas de distance égale, la fonction renvoie le plus grand des deux nombres.

Exercice 1.7.
Ecrire une fonction int `sommeDesCarres(int n)` qui renvoie la somme des n premiers entiers au carré.

Exercice 1.8.
Ecrire une fonction `int sommeDesProduits(int n)` qui renvoie la somme des produits $i \times j, 1 \leq i \leq j \leq n$.

Exercice 1.9.
Ecrire une fonction unsigned `int reste(unsigned int x, unsigned int y)` qui renvoie le reste de la division entière de x par y (utiliser une boucle while, l'opérateur modulo % est interdit).

Exercice 1.10.
Ecrire une fonction void echange(int a, int b) qui échange les valeurs de a et b. Que donne alors le programme suivant :

```
#include <stdio.h>

int main(){
  int a=1,b=2;
  echange(a, b);
  printf("a=%d, b=%d\n",a,b);
}
```

2. Pointeurs simples

Exercice 2.1.
Décrire ce que sont une variable et un pointeur au sens informatique du terme. Faire un schéma pour expliquer leur fonctionnement. Expliquer ensuite comment passer de l'un à l'autre.

Exercice 2.2.
Décrire ce que font les instructions suivantes :

```
[1] int i = 10;          [2] int* p;      [3] p = &i;
[4] printf("%p, %d \n", &i, i);          [5] printf("%p, %d \n", p, *p);
```

Exercice 2.3.
Ecrire un programme qui réalise les actions suivantes :

- Déclarer un entier i initialisé à 0
- Déclarer un pointeur vers un entier p
- Affecter à i une valeur arbitraire
- Faire pointer p vers l'adresse de i
- Saisir une nouvelle valeur pour la variable pointée par p
- Afficher la valeur de i

Exercice 2.4.
Trouver l'erreur dans cet extrait de programme :

```
int x=5;
int *p = &x;
p = 9;
```

Exercice 2.5.
Simplifier les expressions suivantes où p est de type `int*` et i de type `int` :

```
p = &*&i;
i = *&*&j;
```

Exercice 2.6.
Refaire l'exercice *1.10.* en écrivant une fonction d'échange permettant réellement d'échanger les 2 valeurs. Modifier le programme principal pour que l'échange soit effectif.