Travaux Dirigés n°5

Visibilité des variables et appels à fonctions

Exercice 1

Définissez l'affichage produit par l'exécution du programme. Expliquez.

```
#include <stdio.h>

int nb = 3;
//=============================================================

void mal_ecrit_1 (int nb)
{
  printf ("nb mal_ecrit_1 = %d\n",nb++);
  {
    int nb = 14;
    printf ("nb mal_ecrit_1 = %d\n",nb);
  }
  printf ("nb mal_ecrit_1 = %d\n",nb);
}
//=============================================================

int mal_ecrit_2 (int x, int y)
{
  printf ("nb mal_ecrit_2 = %d\n",x + nb);
  printf ("nb mal_ecrit_2 = %d\n",y + nb);
  return (nb *= 0);
}
//=============================================================

int main(void)
{
    int x = 3;
    int y = 6;

    printf ("nb main = %d\n",nb);
    mal_ecrit_1(nb);
    printf ("nb main = %d\n",nb);
    printf("resultat de mal_ecrit_2 = %d\n",mal_ecrit_2(y,x));
    printf ("nb = %d\n",nb);
```

```
    return 0;
}
```

Exercice 2

On considère le programe suivant :

```
#include <stdio.h>

int chose(int a);
int machin();

int chose(int a)
{
  return a+17+machin();
}

int machin()
{
  int a=-1;
  return a;
}

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

Que se passse-t-il si l'on enlève les déclarations des fonctions `chose` et `machin` ? Qu'affiche le programme ?

Exercice 3

On considère le programe suivant :

```
#include <stdio.h>
int a = 27;

int chose(int a);
int machin();

int chose(int a)
{
  return a+17+machin();
}

int machin()
{
```

```
  return a;
}

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

Qu'affiche le programme ?

Exercice 4

Implémenter l'algorithme de tri par sélection en définissant les fonctions :

- int min(int arr[N], int dim, int indice) qui calcule l'indice de l'élément minimum du sousarray comprit entre indice et N ;
- void echange(int array[N], int indice1, int indice2) qui permet l'échange de deux éléments dans l'array ;
- void affiche(int arr[N], int dim) qui affiche les éléments du tableau.

Exercice 5

Ecrire une fonction permettant de calculer le nième terme de la suite de Fibonacci définie par :

$$\begin{cases} u_1 = 1 \\ u_2 = 1 \\ u_n = u_{n-1} + u_{n-2} \end{cases}$$

L'ordre n sera l'argument de cette fonction.

Récursivité

Exercice 6

La suite de Fibonacci est définie de manière récursive par la relation : $u_n = u_{n-1} + u_{n-2}$. Cette définition doit être complétée par une condition d'arrêt. Dans notre cas, si n est égal à 0 ou 1 alors : $u_0 = u_1 = 1$.

On peut tracer les calculs. Ainsi, si on veut calculer u_4 et que l'on suppose que le premier terme de la somme est calculé en premier, on est confronté à la suite de calculs :

```
Fib(4)  =          Fib(3)     +     Fib(2)
   =     Fib(2)     +  Fib(1)  +     Fib(2)
   =  Fib(1)  +  Fib(1)  +  Fib(1)  +     Fib(2)
   =  1  +  Fib(1)  +  Fib(1)  +     Fib(2)
   =  1  +  1  +  Fib(1)  +     Fib(2)
   =     2     +  Fib(1)  +     Fib(2)
   =     2     +  1  +     Fib(2)
   =           3     +     Fib(2)
   =           3     +  Fib(1)  +  Fib(1)
```

```
=           3     +  1  +  Fib(1)
=           3     +  1  +  1
=           3     +     2
=           5
```

Construire un programme qui calcul récursivement le nième terme de la suite de Fibonacci.

Exercice 7

Ecrire un programme qui demande à l'utilisateur d'entrer un entier relatif et qui retourne l'entier relatif écrit en sens inverse (par exemple -2344 deviendra -4432).