Corrigé série 6.1 : Les fonctions I

Exercice 1

Exercice 1

Qu'affiche le programme suivant :

```c
#include <stdio.h>

int fct(int);

int main(void)
{
  int n, p = 5;
  n = fct(p);
  printf("p = %d, n = %d\n", p, n);

  return 0 ;
}

int fct(int r)
{
  return 2*r;
}
```

```
p = 5, n = 10
```

Exercice 2

```
#include <stdio.h>

void f1(void);
void f2(int nb);
int  f3(int nb);

int main()
{
  int j;
  f1();
  f2(3);
  j = f3(3);

  return 0;
}

void f1(void)
{
  printf("bonjour\n");
}

void f2(int n)
{
  int i;
  for(i = 0; i < n; i++)
    f1();
}

int f3(int n)
{
  f2(n);
  return 0;
}
```

Notez la réutilisation du code, f2 et f3 utilisent la fonction f1 plutôt que de la recoder.

Exercice 3

Quels résultats fournira le programme ci-dessous :

```
#include <stdio.h>
int n = 10, q = 2;     // Variables globales !!!
int fct(int);
void f(void);

int main(void)
{
  int n = 0, p = 5;
  n = fct(p);
  printf("A : dans main, n = %d, p = %d, q = %d\n", n, p, q);
  f();

  return 0 ;
}

int fct(int p)
{
  int q;
  q = 2 * p + n;
  printf("B : dans fct, n = %d, p = %d, q = %d\n", n, p, q);
  return q;
}

void f(void)
{
  int p = q * n;
  printf("C : dans f, n = %d, p = %d, q = %d\n", n, p, q);
}
```

```
B : dans fct, n = 10, p = 5, q = 20
A : dans main, n = 20, p = 5, q = 2
C : dans f, n = 10, p = 20, q = 2
```

Pour B : n est une variable globale, p est une variable locale à fct() (c'est le paramètre), q est une variable locale à fct().

Pour A : n est une variable locale à main(), p est une variable locale à main(), q est une variable globale.

Pour C : n est une variable globale, p est une variable locale à f(), q est une variable globale.

Exercice 4

Écrire un programme qui **calcule le carré des nombres inférieurs ou égal** à la valeur saisie au clavier. Définissez une fonction **`carre()`** qui reçoit une valeur et retourne son carré, pour faire ce calcul.

```
Entrez une valeur:
3
1 au carre vaut 1
2 au carre vaut 4
3 au carre vaut 9
```

Choix de la boucle :

I. **Est-ce qu'on connait le nombre d'itérations à effectuer ?** Oui => **`for`**

```c
#include <stdio.h>

int carre(int n)
{
  return n * n;
}


int main()
{
  int i, N;

  printf("Entrez une valeur :");
  scanf(" %d",&N) ;

  for( i=1 ; i<= N ; i++)
  {
     printf("%d au carre vaut %d\t", i, carre(i));
  };

  return 0;
}
```

Optionnel : écrire un programme qui **calcule tous les carrés inférieurs ou égal à une valeur** saisie au clavier. Utiliser une fonction **`carre()`** pour faire ce calcul.

```
Entrez une valeur:
17
1 au carré vaut 1
2 au carré vaut 4
3 au carré vaut 9
4 au carré vaut 16
```

Choix de la boucle :

I. **Est-ce qu'on connait le nombre d'itérations à effectuer ?** Non => **`while`** ou **`do…while`**.

II. **Est-ce qu'on doit passer au moins une fois dans la boucle ? Non**. Si on prend une limite de 1, on n'entre pas dans la boucle.
Donc : une boucle **`while`**.

```
#include <stdio.h>

int carre(int n);                    // Prototype

int main()
{
  int nombre, nombreAuCarre;

  printf("Entrez une valeur :");
  scanf(" %d",&N) ;

  nombre=1;
  nombreAuCarre = carre(nombre);
  while( nombreAuCarre < N)
  {
    printf("%d\t", nombreAuCarre);
    nombre++;
    nombreAuCarre = carre(nombre);
  };
  return 0;
}

int carre(int n)
{
  return n * n;
}
```

On peut condenser le code en faisant l'affectation dans la condition

```
nombre=1;
  while( (nombreAuCarre = carre(nombre)) < N )
  {
    printf("%d\t", nombreAuCarre);
    nombre++;
  };
```

Exercice 5

Écrire un programme qui saisit, au clavier, un nombre compris entre deux valeurs et affiche la table de multiplication du nombre saisi. Le programme `main()` n'est constitué que de deux appels à des fonctions :

```
#include <stdio.h>
#include <stdlib.h>
#define MIN_TABLE 1
#define MAX_TABLE 12

int lire(int lowerLimit, int upperLimit);
void table_de_multiplication(int n);

int main(void)
{
    int chiffre, compteur;
    chiffre = lire(MIN_TABLE, MAX_TABLE);
    table_de_multiplication(chiffre);
    system("pause");
    return 0 ;
}

int lire(int lowerLimit, int upperLimit)
{
    int status, input;
    do
    {
        printf("Quelle  table  de  multiplication(%d-%d)  voulez-vous?",
    MIN_TABLE, MAX_TABLE);
        do
        {
            status = scanf("%d", &input);
            fflush(stdin);
            if (status != 1)
                printf("\a");
        }
        while (status != 1);
    }
    while (! ( (input >= lowerLimit) && (input <= upperLimit) ) );
    return input;
}

void table_de_multiplication(int n)
{
    int compteur;
    printf("TABLE DE MULTIPLICATION DE %d\n\n", n);
    for (compteur = 1; compteur <=10; ++compteur)
        printf("%3d X %3d = %3d\n", n, compteur, n * compteur);
}
```

Exercice 6

Écrire un programme qui effectue des opérations mathématiques simples : `+,-,*,/`. Ce programme doit être réalisé avec une fonction dont l'appel est le suivant :

```
calculer(2.0,5.0,'+') ;
```

L'interface utilisateur est le suivant:

```
Entrez une expression <nb op nb> :
3.5 – 2.3
Le resultat vaut 1.200
```

```c
#include <stdio.h>

float calculer(float, float, char);

int main()
{
    float nb1, nb2, resultat;
    char oper;

    printf("Entrez une expression <nb op nb> : ");
    scanf(" %f %c %f", &nb1, &oper, &nb2);

    resultat = calculer(nb1,nb2,oper);
    printf("Le resultat vaut %.3f\n", resultat);

  return 0;
}



float calculer(float n1, float n2, char op)
{
    switch(op)
    {
      case '+' :
        return (n1 + n2);   // pas besoin de break puisque return
      case '-' :
        return (n1 - n2);
      case '*' :
        return (n1 * n2);
      case '/' :
        return (n2 != 0) ? (n1 / n2) : 0;
      default  :
        return (n1 + n2);
    }
}
```