TP 1 : Architecture des ordinateurs

Opérateurs, itérations, , représentation binaire, limites du calcul numérique, récursivité

Durée : 2 semaines

Les programmes C devront respecter les conventions d'écriture d'un programme C (commentaires, indentation, etc.), être compilés à partir de Dev c++.

Exercice 1 : Ecrire deux programmes `decroissant1.c` et `decroissant2.c` qui demandent un entier positif *n* et affichent tous les entiers de *n* à 0. Le premier programme utilisera une boucle `for`, le deuxième une boucle `while`. Que se passe-t-il si un nombre négatif est entré ?

Exercice 2 :

2.1 On veut écrire un programme en langage C qui calcule par itérations la fonction factorielle, à partir d'un entier positif saisi au clavier par l'utilisateur, puis affiche à l'écran le résultat. En fonction du type d'entier choisi, à partir de quelle valeur saisie, y a-t-il dépassement de capacité ?

2.2 Ecrire un programme en langage C qui calcule par itérations la fonction *pgcd*, à partir de deux entiers saisis au clavier par l'utilisateur, puis affiche à l'écran le résultat. Il serait bon que cette fonction donne toujours un résultat positif.

Exercice 3 : Calcul de la représentation binaire à l'aide des opérateurs `/` et `%`

Ecrire un programme en langage C qui affiche le résultat de la conversion binaire d'un entier positif en écriture décimale. Le programme demande un entier naturel à l'utilisateur, puis affiche successivement la valeur de tous les bits de sa représentation. Il sera réalisé en utilisant les opérateurs `/` (division) et `%` (modulo), sans utiliser d'appels récursifs, ni de structure tableau.

Proposer un algorithme pour lequel cet affichage de bits s'opère dans l'ordre inverse de leur sens de lecture (selon la numération en base deux), puis un autre pour lequel l'ordre de lecture est respecté après affichage.

Exercice 4:

Tester le programme suivant et donner une interprétation du résultat qu'il produit :

```
#include <stdio.h>
int main(void){
int n=0;
double epsilon=1.0;
do {
      epsilon/=10.0;
      n++;
    } while(1.0+epsilon!=1.0);
printf("epsilon = 10^-%d\n",n);
return 0;
}
```

Exercice 5 :

La suite de Fibonacci est définie par la relation de récurrence suivante :

$$\begin{cases} fib(0) = 1 \\ fib(1) = 1 \\ fib(n) = fib(n-1) + fib(n-2), \qquad n > 1 \end{cases}$$

Voici le début de cette suite ; 1 1 2 3 5 ...

- Ecrire en C un programme itératif qui calcule les 10 premiers termes de la suite de fibonacci.
- Ecrire en C un programme récursif qui calcule les 10 premiers termes de cette suite.