Travaux Pratiques d'algorithmique n°1

Cours d'Informatique de Deuxième Année

—DEUG Sciences S3-MI—

Révisions de *C*

L'objet de cette séance de travaux pratiques est le rappel des concepts de base du langage *C*. Nous rappelons entre autres : les tests, les boucles, les fonctions et le passage des paramètres.
Toutes les fonctions programmées devront être commentées.

▸ **Exercice 1.** *Compiler et exécuter le programme suivant. Expliquer son fonctionnement.*

```
#include<stdio.h>

int Mystere(int);

void main(void)
{ int ent, res;

  printf("Entrez un entier: ");
  scanf("%d",&ent);
  res = Mystere(ent);
  printf("%d\n",res);
}
```

```
int Mystere(int n)
{ int resultat = 1;
  int i;

  if (n==0) return 1;
  if (n>0)
     {
       for(i=2;i<n+1;i++)
          resultat *= i;
       return resultat;
     }
  return -1;
}
```

▸ **Exercice 2. (coefficients binomiaux)** *Nous cherchons à calculer les coefficients binomiaux C_n^p définis par les relations suivantes :*

$$C_n^p = \begin{cases} \dfrac{n!}{p!(n-p)!} & \text{si } 0 \le p \le n \ , \\ 0 & \text{sinon.} \end{cases}$$

Écrire la fonction `Binomial` *qui prend comme arguments* `n` *et* `p` *et qui calcule le coefficient binomial C_n^p.*

▸ **Exercice 3.** *Nous cherchons à étudier les propriétés des coefficients binomiaux. Nous nous intéressons à la somme suivante :*

$$\sum_{i=0}^{n} C_n^i \quad .$$

Écrire la fonction `Somme` *qui prend en argument* `n` *et qui calcule cette somme. Essayer cette fonction pour $n = 2, 3, 4, 5, \ldots$. En déduire une évaluation de la somme ci-dessus.*

▸ **Exercice 4.** ***(passage par adresse)*** *Programmer et tester les trois fonctions suivantes. Détailler leurs fonctionnements. Qu'en pensez-vous ?*

```
void swap1(int a, int b)
{
  int tmp;
  tmp = a;
  a   = b;
  b   = tmp;
}
```

```
void swap2(int a, int *b)
{
  int tmp;
  tmp = a;
  a   = *b;
  *b  = tmp;
}
```

```
void swap3(int *a, int *b)
{
  int tmp;
  tmp = *a;
  *a  = *b;
  *b = tmp;
}
```

▸ **Exercice 5.** *Le but de cet exercice est d'implémenter les opérations de base sur les tableaux.*

- *Écrire une fonction permettant de saisir un tableau d'entiers. Le prototype de la fonction sera* `int SaisirTab(int *t, int *nb)` *où* `t` *est le tableau contenant les entiers et* `nb` *est l'adresse de la variable contenant le nombre d'éléments du tableau. La fonction retournera* 1 *si la saisie s'est réalisée correctement,* 0 *sinon ;*
- *écrire une fonction de prototype* `void AffichageTab(int *t, int nb)` *qui affiche le contenu du tableau* `t` *de taille* `nb` *;*
- *écrire une fonction* `Moyenne` *qui retourne la moyenne des valeurs contenues dans un tableau* `t` *de taille* `nb`*.*

▸ **Exercice 6.** *On cherche à afficher la décomposition en base 2 d'un entier* `n`*. Le principe est très simple. On effectue la division entière de* `n` *par* 2*. On trouve un quotient* q_0 *et un reste* r_0*. On recommence avec* q_0*. On construit ainsi une suite de restes* $(r_0, r_1, r_2, \ldots)$*. On arrête le processus dès que l'on trouve un quotient* q_k *égal à* 0*. La décomposition de* `n` *en base* 2 *est alors* $(r_k, r_{k-1}, \ldots, r_2, r_1, r_0)$*. Par exemple pour* $n = 10$ *:*

$$\begin{array}{rclclcl} 10 & = & 5\times 2 & + & 0 & \longrightarrow & r_0 = 0 \\ 5 & = & 2\times 2 & + & 1 & \longrightarrow & r_1 = 1 \\ 2 & = & 1\times 2 & + & 0 & \longrightarrow & r_2 = 0 \\ 1 & = & 0\times 2 & + & 1 & \longrightarrow & r_3 = 1 \end{array}$$

La représentation de 10 *en base* 2 *est donc* $0\ldots01010$*.*

Écrire une fonction qui prend en argument un entier positif `n` *(en base* 10*) et un tableau* `code` *de* 16 *emplacements. Le prototype de la fonction est* `int Base_10_vers_2(int n, char *code)`*. Elle retournera* 1 *si l'opération a pu être effectuée sans dépassement de capacité, autrement elle retournera* 0*. Puis sur le même schéma, écrire la fonction* `int Base_10_vers_k(int n, char *code, int k)` *prenant en paramètre la base* `k` *dans laquelle on souhaite décomposer l'entier* `n`*.*