Printemps 2021 Idées de correction **Examen médian- LO01**

1. Confluence de rivières digitales (7 points) Copie 1

Une rivière digitale est une suite définie de la manière suivante :

- U_0 entier strictement positif
- $U_n = U_{n-1} + \text{somme\_des\_chiffres}(U_{n-1})$

On appellera rivière N la rivière digitale commençant par N ($U_0 = N$).

Exemples :

Rivière 25	**Rivière 1**
$\boldsymbol{U_0} = 25$ $\boldsymbol{U_1} = 25 + (2 + 5) = 32$ $\boldsymbol{U_2} = 32 + (3 + 2) = 37$ $\boldsymbol{U_3} = 37 + (3 + 7) = 47$ … $\boldsymbol{U_{47}} = 620$	$\boldsymbol{U_0} = \boldsymbol{1}$ $\boldsymbol{U_1} = 1 + 1 = 2$ $\boldsymbol{U_2} = 2 + 2 = 4$ $\boldsymbol{U_3} = 4 + 4 = 8$ … $\boldsymbol{U_{57}} = 620$

L'objectif de cet exercice est d'écrire un programme permettant de calculer le point de confluence entre deux rivières K et N.

1. Écrire une fonction *sommeDesChiffres()* qui prend en paramètre un entier positif et retourne la somme de ses chiffres.
2. Écrire une fonction *faireCoulerLaRiviere()* qui prend en paramètre un entier positif U_n et retourne U_{n+1} en utilisant la définition de la rivière digitale
3. Écrire le programme principal. Votre programme doit :

- Lire les rangs de deux rivières *(inutile de vérifier les entrées, nous faisons confiance à l'utilisateur).*
- Faire couler les deux rivières jusqu'à leur confluence. Si la valeur d'une des deux rivières dépasse la constante INT_MAX (définie dans la bibliothèque limits.h), votre programme affichera «Pas de solution trouvée".
- Si une solution existe, afficher le rang (nombre d'itérations) de chaque rivière au point de confluence.

L'affichage devra se faire sous la forme suivante :

```
Point de départ de la première rivière : 25
Point de départ de la seconde rivière : 1
Les deux rivières se rencontrent en 620 après 47 itérations pour la rivière 25
et 57 itérations pour la rivière 1
```

4. Nous voulons maintenant écrire une autre version de la fonction de la question 2. Écrire la fonction *faireCoulerLaRiviere2()* qui modifie le paramètre de valeur U_n avec la valeur de rang supérieur U_{n+1} en utilisant la définition de la rivière digitale.
5. Modifier le programme principal pour utiliser maintenant la fonction précédente (question 4). Ne recopiez pas tout le programme, réécrivez seulement les lignes de code qui doivent être modifiées.

Remarque importante: cet exercice doit être réalisé sans utiliser de tableaux

1.

```
int sommeDesChiffres(int n) {
    int somme = 0;
    while(n > 0){
        somme += n%10;
        n/=10;
    }
    return somme;
}
```

2.
```c
int faireCoulerLaRiviere(int un) {
    return un + sommeDesChiffres(un);
}
```

3.
```c
#include <stdio.h>
#include <limits.h>

int main()
{
    int u0_k, u0_n, k, n, iter_n = 0, iter_k = 0;
    printf("Point de départ de la première rivière : ");
    scanf("%d", &u0_k);
    printf("Point de départ de la seconde rivière : ");
    scanf("%d", &u0_n);

    k = u0_k;
    n = u0_n;

    while(k != n && k < INT_MAX && n < INT_MAX) {
        if(k < n) {
            k = faireCoulerLaRiviere(k);
            iter_k++;
        } else {
            n = faireCoulerLaRiviere(n);
            iter_n++;
        }
    }

    if(k==n) {
        printf("Les deux rivières se recontrent en %d après %d itérations pour la rivière %d et %d
itérations pour la rivière %d", k, iter_k, u0_k, iter_n, u0_n);
    } else {
        printf("Aucune solution trouvée");
    }

    return 0;
}
```

4.
```c
void faireCoulerLaRiviere(int *un) {
    *un = *un + sommeDesChiffres(*un);
}
```

5.
remplacer les 2 appels de la fonction faireCoulerLaRiviere() par

```
faireCoulerLaRiviere2(&k)
faireCoulerLaRiviere2(&n)
```

2. Opérateurs arithmétiques (7 points) Copie 2

1.1 - Écrire quatre fonctions *plus(), moins(), mult(), div()*, permettant de représenter les opérateurs arithmétiques simples +, -, *, /. Chacune des fonctions aura comme paramètre la dimension « dim » de la figure.
Exemple : pour dim = 5

plus(5) : moins(5) multiplier(5) diviser(5)

```
  +          *     *        /
  +           *   *        /
 +++++         ---          *          /
  +           *   *        /
  +          *     *      /
```

1.2 – Écrire le programme principal qui demande à l'utilisateur quel opérateur il veut afficher (+, -, x, /) et vérifie si celui-ci est correct. On demandera également la dimension « dim » de la figure, on vérifiera qu'elle est comprise entre 5 et 25 et qu'elle est impaire (sinon les figures ne seraient pas symétriques). Le programme affichera la figure correspondante.

```c
#include <stdio.h>

#define MIN 5
#define MAX 25


void plus(int nb)
{
int i,j;

for (i=1; i<=nb; i++){
   for (j=1; j<=nb; j++){
        if(i==nb/2+1 || j==nb/2+1)
            printf("+");
        else
            printf(" ");
        }
   printf("\n");
}

}

 void moins(int nb)
{
int i,j;

for (i=1; i<=nb; i++){
   for (j=1; j<=nb; j++){
        if(i==nb/2)
            printf("-");
        else
            printf(" ");
        }
   printf("\n");
}
}

void mult(int nb)
{
int i,j;

for (i=1; i<=nb; i++){
   for (j=1; j<=nb; j++){
        if(i==j || j==(nb-i+1))
            printf("*");
        else
            printf(" ");
        }
   printf("\n");
}
}


 void divi(int nb)
{
int i,j;

for (i=1; i<=nb; i++){
   for (j=1; j<=nb; j++){
        if(j==(nb-i+1))
            printf("/");
        else
            printf(" ");
        }
   printf("\n");
```

```
}
}

int main()
{

   int dim;
   char operateur;

// Test sur la dimension et la parité
   do{
       printf("Entrez la dimension : ");
       scanf("%d",&dim);
   } while ((dim<MIN) || (dim>MAX) || (dim%2 !=1));


// Test sur la validité de l'opérateur

   do{
       fflush(stdin);
       printf("Entrez l'operateur (+,-, *, /) : ");
       scanf("%c", &operateur);
   } while( (operateur !='+') && (operateur !='-') && (operateur !='*') && (operateur !='/') );

   switch(operateur) {

       case '+' : {plus(dim); break;}
       case '-' : {moins(dim); break;}
       case '*' : {mult(dim); break;}
       case '/' : {divi(dim); break;}
   }

}
```

3. Nombres premiers, caractères et pointeurs (6 points) Copie 3

3.1 Nombres premiers

On veut écrire un programme qui demande à l'utilisateur de saisir un nombre entier supérieur ou égal à 2 et qui affiche sa décomposition en facteurs premiers. Si le nombre est premier, il affiche uniquement un message annonçant que le nombre est premier.

Exemples :

la décomposition en facteurs premiers de 24 est 2*2*2*3
la décomposition en facteurs premiers de 26 est 2*13
7 est un nombre premier
la décomposition en facteurs premiers de 36 est 2*2*3*3

Écrire le programme correspondant, le programme s'assurera que le nombre entré est bien supérieur ou égal à 2.

3.2 Suite de caractères

Écrire un programme qui lit une phrase terminée par un point (.), et calcule et affiche la fréquence des caractères alphabétiques contenus dans cette phrase (les autres caractères étant la ponctuation, les espaces ou tout autre caractère non alphabétique).

Remarque importante : cet exercice doit se faire sans utiliser de chaîne de caractères ni de tableau.

3.3 Pointeurs

Qu'affiche le `printf` ?

```
float x , y;
float *pt1, *pt2;

pt2 = &x ;
*pt2 = 5.0 ;
y = x + 3.0;
pt1 = pt2;
*pt1 = *pt2 * 2.0;
pt2= &y;
(*pt2)-- ;
printf(" x= %f, y= %f, *pt1= %f, *pt2= %f\n", x, y, *pt1, *pt2);
```

1. sur (3,5 points)

```c
#include <stdio.h>
#include <math.h>


int main()
{

    int nombre;     /* nombre entier reconstruit à partir de le suite de caractères lus */
    int nb;         /* copie du nombre donné par l'utilisateur pour etre modifié dans la boucle */
    int i;          /* diviseurs successifs */
    int n=0;        /* nombre de facteurs premiers trouvés */


    do {
        printf("donner un nombre (>1)  SVP  : ");
        scanf("%d", &nombre);
    }
    while (nombre<=1);

    i=2;    // initialisation du premier diviseur
    n=0;    // nombre de diviseurs trouvés initialisé à 0
    nb=nombre;
    while (i<=sqrt(nombre)) {
        if (nb%i== 0){  /* i est un diviseur donc il ne sera pas incrémenté pour pouvoir etre re-testé */
            n++;
            nb/=i;    /* on divise le nombre pour le test suivant*/
            if (n==1)
                printf("la décomposition en facteurs premiers de %d est %d", nombre ,i );
            else
                printf("*%d",i );
        }
        else
            i++; /* i n'est pas un diviseur, on l'incrémente pour tester la valeur suivante */
    }
    if (n==0)
        printf("%d est un nombre premier", nombre);
    printf("\n");
    return 0;
    }
```

2. sur (2 points)

```c
#include <stdio.h>
int main()
{

    char car;
    int alph, total;
    float freq;
    alph=0;
    total=0;
    printf("Donnez votre phrase terminée par un point(.):");
    while ((car=getchar())!='.')
    {
       if ((car>='A' && car<='Z') || (car>='a' && car<='z'))
           alph++;
       total++;
    }
    total++; // pour le point
    freq=(float)alph/ (float)total;
    printf("\n La fréquence des caractères alphabétiques  = %f ", freq);

    return 0;
}
```

3. (sur 0,5 point)

x= 10.000000, y= 7.000000, pt1= 10.000000, pt2= 7.000000