Université Libanaise **Final 2016/2017**
Faculté des Sciences I **Durée : 2 heures**

I1101
Algorithmes et Programmation

Problème I — 30 points

On considère la séquence suivante :

$$\begin{cases} U_0 = 1 \\ U_n = U_{n-1} + \frac{U_{n-1}}{n!} \quad si\ n > 0 \end{cases}$$

1. Écrire la fonction **int Factoriel(int n)** qui retourne le factoriel d'un entier n envoyé en paramètre.
2. Écrire la fonction **float Sequence(int n)** qui retourne la valeur de l'élément U_n de la séquence ci-dessus. Cette fonction utilise la fonction Factoriel.
3. Écrire la fonction main qui demande à l'utilisateur de donner un entier n puis calcule et affiche sur l'écran la valeur de U_n.

Problème II — 35 points

Deux nombres N1 et N2 sont dits frères si chaque chiffre de N1 apparait ou moins une fois dans N2, **et** choque chiffre de N2 apparait au moins une fois dans N1.
Exemples :

- N1 = 1162 et N2 = 612, sont frères.
- N1 = 905 et N2 = 9059, sont frères.
- N1 = 405 et N2 = 554, ne sont pas frères.

1. Écrire la fonction **Initialise** qui initialise les éléments d'un tableau d'entiers à **-1**. Le tableau et sa taille sont envoyés comme paramètres.
2. Écrire la fonction **Appartient** qui teste si un entier v existe dans un tableau d'entiers T. La fonction retourne 1 si v existe dans T et 0 sinon. L'entier v, le tableau T et sa taille sont envoyés comme paramètres.
3. Écrire la fonction **Decompose** qui prend en paramètre un entier n, un tableau d'entiers T et sa taille. Cette fonction remplit T par les chiffres de n. On suppose que la taille du tableau est plus grande que le nombre de chiffres de l'entier n.
4. En utilisant les fonctions précédentes, écrire la fonction main qui demande à l'utilisateur de donner deux entiers N1 et N2 et teste s'ils sont frères ou non.

Problème III — 35 points

1. Écrire la fonction **estAlphabet** qui teste si un caractère envoyé en paramètre est une lettre Alphabétique ou non (la fonction retourne 1 si le caractère est un alphabet et 0 sinon).
2. Écrire la fonction **EliminationNoAlphabet** qui prend une chaine de caractères comme Paramètre et élimine les caractères non alphabétiques d'elle.
   Exemple : Si S = "ab A 132?CD#"
   Après élimination, la chaine de caractères S deviendra "abACD"

Question de LAB — 15 points

3. Écrire une fonction qui prend deux chaines de caractères S1 et S2 comme paramètre et regroupe toutes les lettres alphabétiques des deux chaines (S1 et S2) dans S1 et les autres caractères des deux chaines dans S2.
   Exemple : Si S1="ab123??" et S2 = "&@Mp9C"
   Après appel à la fonction, les deux chaines deviendront : S1="abMpC" et S2 = "123??&@9"

Suggestion : On peut utiliser la fonction **strcpy**(STR1, STR2) qui copie la chaine STR2 dans la chaine STR1.

Bonne chance

Solution

1ère année – 2ème semestre 2017

Problème I

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

1.

```
int factoriel(int n)
{
	int i, fact = 1;

	for ( i = 1; i <= n; i++)
		fact *= i;

	return fact;
}
```

2.

```
float sequence(int n)
{
	float Un = 1;
	int i;

	for ( i = 1; i <= n; i++)
		Un = Un + Un / factoriel(i);

	return Un;
}
```

Ou

```
float sequence(int n)
{
	if (n == 0) return 1;

	return sequence(n - 1) + sequence(n - 1) / factoriel(n);
}
```

3.

```
void main()
{
	int n;

	printf("donner n pour calculer Un");
	scanf("%d", &n);

	printf("\nU%d =  %f", n, sequence(n));
}
```

Problème II

```
#include <stdio.h>
#define N 100
```

1.

```
void Initialize(int T[], int n)
{
	int i;
	for (i = 0; i < n; i++)
		T[i] = -1;
}
```

Ou

```
void Initialize(int T[], int n)
{
	while (n--)
		T[n] = -1;
}
```

2.

```
int Appartient(int v, int T[], int n)
{
	int i, test = 0;

	for (i = 0; i < n; i++)
		if (v == T[i])
		{
			test = 1;
			break;
		}

	return test;
}
```

3.

```
int Decompose(int n, int T[], int t)
{
	int i;

	for (i = 0; n != 0; i++)
	{
		T[i] = n % 10;
		n /= 10;
	}
}
```

4.

```
 void main()
{
	int i, N1, N2, T1[N], T2[N], test = 1;

	printf("donner 2 entier pour tester si freres ou non");
	scanf("%d%d", &N1, &N2);

	Initialize2(T1, N);
	Initialize2(T2, N);

	Decompose(N1, T1, N);
	Decompose(N2, T2, N);

	for (i = 0; T1[i] != -1; i++)
		if (Appartient(T1[i], T2, N) == 0)
			test = 0;

	for (i = 0; T2[i] != -1; i++)
		if (Appartient(T2[i], T1, N) == 0)
			test = 0;

	if (test)
		printf("\n %d et %d sont freres", N1, N2);
	else
		printf("\n %d et %d sont non freres", N1, N2);
}
```

Problème III

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#define N 100
```

1.

```
int estAlphabet(char c)
{
	if ((c >= 'a'&&c <= 'z') || (c >= 'A'&&c <= 'Z'))
		return 1;

	return 0;
}
```

2.

```
void EliminationNoAlphabet(char S[])
{
	int i, j;
	for (i = 0; S[i] != '\0'; i++)
		if ( !estAlphabet(S[i]))
			for (j = i--; S[j] != '\0'; j++)
				S[j] = S[j + 1];

}
```

3.

Méthode 1 :

```
void EliminationAlphabet(char S[])
{
	int i, j;

	for (i = 0; S[i] != '\0'; i++)
		if (estAlphabet(S[i]))
			for (j = i--; S[j] != '\0'; j++)
				S[j] = S[j + 1];
}
void arrange(char S1[], char S2[])
{
	char alpha[N], other[N], Stemp1[N], Stemp2[N];

	strcpy(Stemp1, S1);
	strcpy(Stemp2, S2);

	EliminationNoAlphabet(S1);
	EliminationNoAlphabet(S2);

	strcat(S1, S2);

	/* strcat (destination, source); Elle ajoute le contenu d'une
chaîne(source) à la suite d'une autre.*/

	EliminationAlphabet(Stemp1);
	EliminationAlphabet(Stemp2);
	strcat(Stemp1, Stemp2);

	strcpy(S2, Stemp1);
}
```

Méthode 2 :

```
void arrange(char S1[], char S2[])
{

	int i, j = 0, k = 0;
	char alpha[N], other[N];

	for (i = 0; S1[i] != '\0'; i++)
	{
		if (estAlphabet(S1[i]))
		{
			alpha[j] = S1[i];
			j++;
			// c'est meme que alpha[j++] = S1[i];
		}
		else
		{
			other[k] = S1[i];
			k++;
		}
	}
```

```
	for (i = 0; S1[i] != '\0'; i++)
	{
		if (estAlphabet(S1[i]))
		{
			alpha[j] = S1[i];
			j++;
		}
		else
		{
			other[k] = S1[i];
			k++;
		}
	}

	alpha[j] = '\0';
	other[k] = '\0';

	for (i = 0; i <= j; i++)
		S1[i] = alpha[i];    // ou avec suggestion strcpy(S1,alpha)

	for (i = 0; i <= k; i++)
		S2[i] = other[i];

}
```

Ou

```
void arrange_helper(char Sx[], char alpha[], char other[])
{
	int i, j = strlen(alpha), k = strlen(other);

	for (i = 0; Sx[i] != '\0'; i++)
		if (estAlphabet(Sx[i]))
			alpha[j++] = Sx[i];
		else
			other[k++] = Sx[i];

	alpha[j] = '\0';
	other[k] = '\0';
}

void arrange(char S1[], char S2[])
{
	char alpha[N] = { '\0' }, other[N] = { '\0' };

	arrange_helper(S1, alpha, other);
	arrange_helper(S2, alpha, other);

	strcpy(S1, alpha);
	strcpy (S2, other);
}
```