TP N°2 et Mini-Projet : Programmation en langage C
LE – Informatique et Mathématique

Objectif : Initiation au langage C

Prérequis : Installation du logiciel **Code::Blocks** ou **Autre** éditeur pour langage C.

Important : Les solutions des exercices du TP doivent être codées en langage C, **compilées** et **exécutées**. Ensuite, le travail à rendre sur la plateforme sera sous forme d'un **rapport** (*un fichier **word*** ou ***pdf***) contenant pour chaque exercice les captures d'écran du **code source ainsi que celles des résultats après l'exécution**.

Exercice 1:

Écrire un programme en langage C qui demande à l'utilisateur un entier ***n*** et affiche les ***diviseurs*** de ce dernier ainsi que la ***somme des diviseurs***. Exemple : Si l'utilisateur entre le nombre 10. Ses diviseurs sont : 1, 2, 5, 10. Et la somme des diviseurs est 18.

Exercice 2 :

Écrire un programme en langage C qui demande à l'utilisateur un nombre compris entre *1 et 3* ***jusqu'à ce que la réponse convienne***, avec l'affichage des messages:

- Si la valeur est inférieure à 1, on affiche "Entrez une valeur encore plus grande !".
- Si la valeur est supérieure à 3 on affiche "Entrez une valeur encore plus petite! ".

Exercice 3 :

Écrire un programme en langage C qui déclare un **tableau** de **30 notes** dont on fait ensuite saisir les valeurs par l'utilisateur. Nous souhaitons ensuite effectuer l'ensemble des manipulations suivantes toujours dans le même algorithme :

1. Afficher la valeur de la 5ème note saisie par l'utilisateur.
2. Afficher ensuite l'ensemble des notes saisies.
3. Modifier la partie de la saisie des notes de sorte à ce qu'on teste la valeur entrée par l'utilisateur, celle-ci doit être comprise entre 0 et 20.
4. Calculer et afficher la somme des notes du tableau.
5. Afficher les notes du tableaux après avoir multiplier toutes ces notes par un coefficient saisie par l'utilisateur.

   Attention : la multiplication par le coefficient doit se faire uniquement au niveau de l'affichage. Autrement dit, on ne doit perdre en au cas les notes initiales au niveau du tableau.

6. Ajouter les instructions qui permettent de calculer et afficher la moyenne de ces notes.
7. Compter et afficher ensuite le nombre des notes supérieures ou égales à 10.
8. Déterminer la note maximale, la note minimale ainsi que leurs positions dans le tableau.

Exercice 4 : Tableau à deux dimensions

Soit deux matrices A et B de taille (5,5).

1. Écrire un programme permettant de saisir les valeurs des deux matrices A et B.
2. Ensuite dans le même programme, calculer et afficher la somme de ces deux matrices.
3. Calculer et afficher ensuite le produit des deux matrices.
4. Ajouter les instructions qui permettent de retrouver et afficher la plus grande valeur au sein de la matrice A ainsi que le position de cette valeur.

Exercice 5 : Manipulation des chaînes de caractères :

Rappel sur les chaînes de caractères :

Le type **char** permet de manipuler uniquement un simple caractère. Dans l'exemple suivant, on demande à l'utilisateur d'entrer un caractère et puis on l'affiche :

```c
#include <stdio.h>

int main(int argc, char *argv[])
{
    char lettre = 0;
    printf("Entrez un caractère : \n ");
    scanf("%c", &lettre);
    printf("Le caractère que vous avez entré est : %c\n", lettre);
    return 0;

}
```

```
Entrez un caractère :
 A
Le caractère que vous avez entré est : A
Program ended with exit code: 0
```

Mais pour manipuler une chaîne de caractère on doit passer par un tableau. Il est donc important à savoir qu'une chaîne de caractères n'est rien d'autre qu'un tableau de type **char**.
Autrement dit, si nous souhaitons manipuler une chaîne de caractère, nous devons commencer par la déclaration du tableau de type **char** qui va la contenir.
La figure suivante montre le schéma représentatif de la chaîne de caractères: "Bonjour " dans un tableau nommé Tab.

B	o	n	j	o	u	r	\0
Tab[1]	Tab[2]	Tab[3]	Tab[4]	Tab[5]	Tab[6]	Tab[7]	Tab[8]

Remarque :

- Une chaîne de caractère **doit impérativement contenir un caractère spécial à la fin de la chaîne**, appelé *caractère de fin de chaîne* . Ce caractère s'écrit comme ceci : `'\0'`.
- Tout simplement pour que l'ordinateur sache quand s'arrête la chaîne. À chaque fois on arrive au caractère `\0`, on comprend que la chaine en question est terminée.
- Ce qui veut dire que lors de la définition de la taille du tableau, on doit toujours tenir compte de l'existence de ce caractère de fin de chaîne.
- Et donc pour stocker la chaîne de caractère "Bonjour" on doit déclarer un tableau de taille 8 et non pas 7 : `Tab[8]`.

Application : Le programme suivant vous montre comment stocker et afficher une chaine de caractère

```c
#include <stdio.h>
int main(int argc, char *argv[])
{
  char Tab[8]; // Tableau de 8 char pour stocker B-o-n-j-o-u-r +le \0
// Initialisation de la chaîne
//(on écrit les caractères un par un en mémoire)
    Tab[0] = 'B';
    Tab[1] = 'o';
    Tab[2] = 'n';
    Tab[3] = 'j';
    Tab[4] = 'o';
    Tab[5] = 'u';
    Tab[6] = 'r';
    Tab[7] = '\0';

// Affichage de la chaîne grâce au format %s du printf pour chaine de caractère
    printf("%s \n", Tab);

// Autre façon de faire, il s'agit de passer toute la chaîne au tableau d'un seul coup,
    char Tab2[] = "Re - Bonjour"; // ici la taille du tableau Tab2 est automatiquement calculée, toujours en prenant en compte le caractère de fin de la chaine \0
// Affichage de la chaîne en utilisant toujours le format %s
    printf("%s \n", Tab2);

    return 0;
}
```

```
Bonjour
Re - Bonjour
Program ended with exit code: 0
```

Exercice 5:

Écrire un programme qui demande à l'utilisateur d'entrer son nom et son prénom sous forme de deux chaînes de caractères distinctes et les affiche par la suite.

- Calculer par la suite la longueur d'une des chaînes saisies en utilisant une boucle sans compter le caractère de fin `\0`.

Ensuite, en utilisant les fonctions de la bibliothèque **<string.h>**,

- Calculer et afficher la longueur des deux chaînes de caractères, en utilisant la fonction ***strlen(chaîne)*** qui permet ceci sans compter le caractère de fin.
- Concaténer les deux chaînes de caractère dans une seule chaîne de caractère en utilisant la fonction ***strcat(chaine 1, chaine 2)***, qui permettra d'insérer ***chaine 2*** après la fin de ***chaine 1***.
- Il faut faire attention par contre à la taille de ***chaine 1*** qui doit être suffisante pour stocker les deux chaines.

Mini – Projet :

La suite de Fibonacci est définie par récurrence comme suit :

$$u_0 = 0,$$
$$u_1 = 1,$$
$$u_n = u_{n-1} + u_{n-2} \ \ pour\ n \geq 2$$

Objectif :

Écrire un programme qui permet de calculer le $n^{ième}$ terme de la suite de Fibonacci

Travail à faire :

1. *Pour ce faire, il faut tout d'abord comprendre le problème en commençant par des essaies, commencer par attribuer à n des valeurs pour bien assimiler le problème.*
2. *Prenez le temps par la suite à traduire toutes vos idées sous forme d'un algorithme.*
3. *Représentez l'algorithme sous forme d'un organigramme et ensuite sous forme de pseudo-code.*
4. *Une fois l'algorithme est complet, essayer à nouveau par des jeux de données pour tester.*
5. *Finalement traduisez votre algorithme en programme en langage C. Compilez et Exécutez.*

Note : Votre rapport doit contenir impérativement les étapes 3 et 5.