TP 2 - C

Les Tps se font sous un environnement linux, vous pouvez utiliser l'éditeur de texte que vous preferez.

Partie I : Bases

1. Moyenne

1. Ecrire un programme qui affiche la moyenne d'une suite d'entiers positifs entrés au clavier.

On arrêtera la saisie quand le nombre –1 est entré, comme dans l'exemple suivant.

Entrez un entier positif : 5

Entrez un entier positif : 2

Entrez un entier positif : 3

Entrez un entier positif :–1

La moyenne de ces 3 entiers vaut : 3.333333

2. Min Max :

1. Créez un programme qui calcule le minimum et le maximum d'un tableau d'entiers de taille 5.
2. Plutôt qu'un tableau, faites entrer le nombre d'entiers par l'utilisateur puis demandez lui de les saisir.

3. Des étoiles

1. Ecrire un programme qui affiche un triangle rempli d'étoiles, s'étendant sur un nombre de lignes entré au clavier, comme dans l'exemple suivant :

Nombre de lignes = 5

*

* *

* * *

* * * *

* * * * *

1. Demandez à l'utilisateur d'entrer une lettre ('g' ou 'd' pour droite ou gauche) permetant de changer le côté de la pointe du triangle (précédement les étoiles étaient collées à gauche):
2. Demandez à l'utilisateur de choisir si le triangle doit pointer vers le haut ou vers le bas

4. Tri a bulle

1. Créez un programme qui vérifie qu'un tableau d'entiers de longueur 6 est trié (les nombres sont dans l'ordre croissant).
2. Modifiez le programme pour qu'il inverse la position de tout élèment plus grand que l'élement précédent avec ce dernier.

Par exemple pour le tableau 1 2 4 3 6 5 on inversera le 4 et le 3 puis le 6 et le 5.

1. Modifiez ce programme pour qu'il effectue cette action jusqu'a ce que le tableau soit trié

5. Comptage de lettres

1. Ecrire un programme qui compte le nombre de chacune des lettres de l'alphabet d'un texte écrit au clavier. Pour simplifier, on ne tiendra compte que des minuscules, mais on comptera le nombre de caractère non reconnus. La saisie sera de 100 charactères maximum.

Partie II : Avancé

6. Mot de passe

On considère le code suivant qui demande un mot de passe à un utilisateur et lui propose de faire une requête à un administrateur si le mot de passe est faux :

```
#include "stdio.h"
#include "string.h"
#include "stdlib.h"

int main(){
  char message[200];
  char password[8] = "bonjour";
  char user_input[8];
  int logged_in = 0;

  while (!logged_in){
    // Ask the user for the password
    printf("please enter a password\n");
    scanf("%s", user_input);

    if (!strcmp(user_input, password)){
      logged_in = 1;
    } else {
      printf("Wrong password\nWould you like to send a request to the admin ? (y/
n)");

      // ask the user if he wants to send a permission request to the admins
      char send_msg;
      scanf(" %c", &send_msg);
      if(send_msg == 'y'){
        printf("write your message here : \n");
        scanf("%s", message);
        printf(" we will send your request : (%s)\n", message);
      }

      // ask the user if they want to try again or leave
      printf(" do you wish to try again ? (y/n)");
      char try_again;
      scanf(" %c", &try_again);
      if (!try_again){
        exit(0);
      }
    }
  }
  printf("You are logged in ! \n");
}
```

1. Ce code contient une faille de sécurité qui permet de s'identifier sans connaitre le mot de passe, trouvez la et utilisez la pour arriver à faire afficher le texte "You are logged in ! " sans taper le mot de passe.

2. Corrigez le code et vérifiez que la faille n'existe plus.

7. Lecture d'arguments et de fichier

Partie 1

Ci-dessous un code qui lit un fichier et affiche son contenu à l'ecran.

```c
#include "stdio.h"

int main(int nombre_parametres, char* parametres[]) {

    // On crée ici une variable de type FILE*
    // ce type représente un fichier
    FILE* mon_fichier;

    // On crée un tableau de 50 chars
    // Ce tableau nous permettra de lire les charactères du fichier 50 par 50
    char tableau[50];

    // On oouvre le fichier "text.txt" en mode lecture ("r" pour "read")
    mon_fichier = fopen("text.txt", "r");

    if (!mon_fichier) { // si le fichier n'est pas valide
        printf("Impossible d'ouvrir le fichier \n");
        return 1;
    }

    printf("Le contenu du fichier est :\n");

    // Tant qu'il est possible de lire 50 charactères de plus dans le fichier on les
lit
    // et on les affiche
    while ((fgets(tableau, 50, mon_fichier))) {
        printf("%s", tableau);
    }
}
```

1. Reprenez le code de l'exercice "Comptage de lettres" et modifiez ce code pour qu'il compte les lettres dans le fichier "test.txt".

Partie 2

Les arguments de la fonction `int main(int nombre_parametres, char* parametres[])` sont deux variables `nombre_parametres` et `parametres` qui permettent d'accéder aux arguments passsés au programme.

- `nombre_parametres` est un entier qui contient le nombre de paramètres passés au programme y compris la commande qui a permis de l'executer
- `parametres` est un tableau de chaines de charactères qui contient les différents paramètres

Par exemple si on appelle `./a.out -l a` le nombre d'arguments sera 3 et le tableau contiendra les chaines de charactères "./a.out", "-l" et "a".

1. Modifiez le programme pour qu'il accepte une liste de fichiers en arguments et compte les lettres de tous ces fichiers.
2. Modifiez le programme pour qu'il permette d'utiliser le flag "-m" pour compter les lettres en majuscule