# TP8 chaines de caractères

**Programme1**

Un mot est dit palindrome si on inverse l'ordre de ses lettres on obtient le même mot. Exemple :"neveroddoreven".

Ecrire un programme qui vérifie si un mot est palindrome (prendre 80 max de character). Test d’exécution :

Donner un mot :

starcomedybydemocrats

Starcomedybydemocrats est un palindrome

**Programme2**

Soit une chaine de caractères constituée exclusivement de caractères chiffre ('0' à '9'). On souhaite écrire un programme C qui compte le nombre d'occurrences de chacun des caractères chiffres et de fournir le résultat sous forme d'un tableau où on trouvera à l'indice i le nombre de caractères i dans la chaine. Exemple : "01230012340067080" devra produire le tableau résultat suivant :

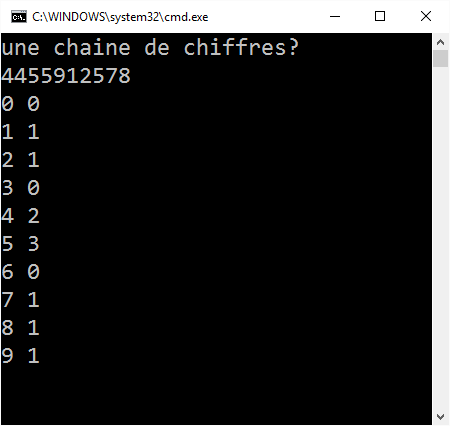
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 7 | 2 | 2 | 2 | 1 | 0 | 1 | 1 | 1 | 0 |

Ecrire le programme C qui demande la saisie des chaines de caractères contenant uniquement des caractères chiffres, remplit le tableau résultat ; Attention : pour trouver l'entier équivalent à un caractère contenant un chiffre :

char a='3' ;

int ia=a-'0' ; // donne à ia la valeur entière 3.

Votre programme doit ensuite afficher le nombre d’occurrences de chaque caractère dans la chaine saisie par l’utilisateur.



**Programme3**

Ecrire un programme C qui lit un texte S et élimine de S toute séquence de caractères formée de plus de 2 occurrences d’un même caractère. Le programme affiche S après réduction. (Prendre max du chaine 100)

*Exemple : si S est “aabbbacdddddf” ; après exécution du programme devient “aaacf".*

*Si S est “aabcabbdf” ; après l’appel de la fonction, S reste inchangée.*

**Solution**

Note :

* Les méthodes ‘expert’ sont des méthodes optionnelles à faire.

Programme 1

#include <stdio.h>

#define N 80

void main ()

{

char chaine[N];

int i, lng = 0, test = 0;

Remplir normal d’une chaine de caractère de longueur variable mais limitée par 80

printf("Entrez une chaine (max 80) :");

gets(chaine);

for (i = 0; chaine[i] != '\0'; i++)

Calcule de longueur de cette chaine dans lng

{

lng++;

}

for (i = 0; i <= lng / 2; i++)

Algorithme principal voire trace à la fin

Si Test devient = 1 implique un non symétrie de chaine c.à.d. no palindrome

{

if (chaine[i] != chaine[lng - i - 1])

{

test = 1;

break; // optionnel

}

}

if (test == 1)

{

printf("le mot n'est pas palindrome");

Affichage selon les cas…

}

else

{

printf("le mot est palindrome");

}

}

Exemple d’exécution :

**Entrez une chaine (max 80) :esttse**

**le mot est palindrome**

**Entrez une chaine (max 80) :a ca va**

**le mot n'est pas palindrome**

**Entrez une chaine (max 80) :estttse**

**le mot est palindrome**

Programme 2

**Méthode 1 (simple):**

#include <stdio.h>

#include <conio.h>

#include <string.h>

#define N 9

#define M 100

void main()

{

char texte[N];

int lng = 0, i, j, T[N], temp, counter;

printf("entrer votre nombre de 0 et 9 seulement :\n");

gets(texte);

lng = strlen(texte);

puts("votre final input texte est: ");

puts(texte);

* Conversion de la valeur de chaque caractère chiffre en son chiffre et le comparer s’il est égal à « j ».
* Calculer les redondances (répétition) de j dans le chaine texte et le mettre dans « counter » puis affecter ce dernier au tableau

for (j = 0; j <= 9; j++)

{

counter = 0;

for ( i = 0; i < lng; i++)

if (texte[i]-'0'==j)

counter++;

T[j] = counter;

}

printf("\n resultat: \n", i, T[i]);

for (i = 0; i <= N; i++)

{

printf("%d %d \n", i, T[i]);

}

}

**Méthode 2 (simple):**

#include <stdio.h>

#include <conio.h>

#include <string.h>

#define N 9

#define M 100

void main ()

{

char texte[N];

int lng = 0, i, T[N], temp, test = 1;

Annuler tout élément dans T afin d’ajouter directement après des “++” ou +1.

for (i = 0; i <= N; i++)

T[i] = 0;

Entrer d’un texte conditionné par des chiffres seulement.

Par défaut tout texte est formé de caractère chiffre (test=0) mais si un seul chiffre n’est entre 0 et 9 on entre dans « while » autre fois car test devient =1.

while (test)

{

printf("entrer votre nombre de 0 et 9 seulement :\n");

test = 0;

gets(texte);

for (i = 0; texte[i] != '\0'; i++)

if (texte[i]<'0' || texte[i]>'9')

test = 1;

} // note 0 : condition fausse, tout autre non nul valide

Calcule de longueur de chaine utilisant Librairie <string.h>

lng = strlen(texte);

puts("votre final input texte est: ");

puts(texte);

Conversion de la valeur de chaque caractère chiffre en son chiffre dans un type entier (« temp ») et ajouter 1 a son case correspandant dans T.

for (int i = 0; i < lng; i++)

{

temp = texte[i] - '0';

T[temp]++;

}

Affichage clair du tableau avec ses indices correspondants.

printf("\n resultat: \n", i, T[i]);

for (i = 0; i <= N; i++)

printf("%d %d \n", i, T[i]);

}

Exemple d’exécution :

**entrer votre nombre de 0 et 9 seulement :**

**453fg345**

**entrer votre nombre de 0 et 9 seulement :**

**05042017**

**votre final input texte est:**

**05042017**

**resultat:**

**0 3**

**1 1**

**2 1**

**3 0**

**4 1**

**5 1**

**6 0**

**7 1**

**8 0**

**9 0**

Programme 3

**Méthode 1 (simple):**

#include <stdio.h>

#include <conio.h>

#define N 100

void main ()

{

char chaine[N], chainecorrige[N];

int k = 0, i, i\_initial, equalnb;

puts("enter votre texte:");

gets(chaine);

for (i = 0; chaine[i] != '\0'; i++)

{

i\_initial = i;

equalnb = 1;

Compteur des nombres de répétitions des caractères consécutifs et l’ajouter a « equalnb ».

while (chaine[i] == chaine[i + 1])

{

equalnb++;

i++;

}

if (equalnb == 1)

{

chainecorrige[k] = chaine[i\_initial];

k++;

Remplir un autre chaine corrigé en utilisant un compteur indépendant « k ».

}

if (equalnb == 2)

{

chainecorrige[k] = chaine[i\_initial];

k++;

chainecorrige[k] = chaine[i\_initial + 1];

k++;

}

}

Toujours il y a « k++ »

Le chaine devient de longueur k

chainecorrige[k] = '\0';

puts("votre chaine corrige est :");

puts(chainecorrige);

}

Exemple d’exécution :

enter votre texte:

aaabaabcddaaa

votre chaine corrige est :

baabcdd

Méthode 2 (expert) :

Points of weakness of simple method:

* Definition of 2 chains when 1 is enough;
* Use “if” sequences when doing job with interval and boucle is stronger algorithm for other case more generic.

when we can define one chains?

if the first is no longer needed and when the result length is equal or less of the initial one.

#include <stdio.h>

#include <conio.h>

#define N 100

void main()

{

char chaine[N];

int k = 0, j, i, i\_initial, equalnb;

puts("enter votre texte:");

gets(chaine);

for (i = 0; chaine[i] != '\0'; i++)

{

i\_initial = i;

equalnb = 1;

while (chaine[i] == chaine[i + 1])

{

Même méthode de compteur

equalnb++;

i++;

}

for (j = i\_initial; equalnb <= 2 && j < i\_initial+equalnb; k++, j++)

{

Remplir dans la même chaine par « overwrite » utilisant un compteur k initialisé par 0 et qui n’incrémente que si les caractères à remplir sont valides, et une autre j pour copier les caractères demandés, (on a en général j>=k)

chaine[k] = chaine[j];

}

}

chaine[k] = '\0';

puts("votre chaine corrige est :");

puts(chaine);

}

enter votre texte:

aaabaabcddaaa

votre chaine corrige est :

baabcdd