# TP9 – Fonctions

**Programme 1**

1. Ecrire une fonction qui teste si une chaine est un palindrome ou pas. On appelle palindrome un mot qui se lit de la même façon de gauche à droite. Exemple:"neveroddoreven" est un palindrome.
2. Ecrire une fonction main qui saisit une chaine de caractères et affiche à l'écran si c'est un palindrome ou pas.

**Programme 2**

1. Ecrire une fonction int MAX (int a, int b) qui retourne le maximum de 2 entiers ;
2. Tester la fonction MAX en utilisant une instruction simple.
3. Ecrire une fonction main qui saisit N entiers, demande à l'utilisateur de saisir N entiers et calcule leur maximum en utilisant la fonction MAX (sans utiliser un tableau!)

**Programme 3**

1. Ecrire une fonction qui teste si un entier est ABONDANT (retourne 1 si oui et 0 sinon). L'entier n est abondant lorsque la somme de ses diviseurs (y compris n lui-même) est plus grand que sa double (2n). Exemple : 12 est abondant (1+2+3+4+6 +12>24)
2. Tester la fonction en utilisant une instruction simple ;
3. Ecrire une fonction main qui saisit 2 entiers a et b puis affiche les nombres abondants dans l'intervalle [a,b].

**Programme 4**

1. Ecrire une fonction char encrypt (char c, int k) qui qui crypte un caractère comme suit : *si c’est un caractère alphabétique, décaler c par k vers la droite (+k) sinon décaler c par k vers la gauche (-k)*
2. Ecrire une fonction main qui teste la fonction.

**Solution**

Programme 1

#include <stdio.h>

#define N 80

**Part 1:**

Type de la fonction ( int return un valeur int)

Nom de la function

Parameter prix par le fonction

int test(char c[])

{

int lng = 0, i, temp = 0;

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

Calcule de longueur de cette chaine dans lng

{

lng++;

}

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

{

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

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

{

temp = 1;

}

}

return temp;

}

**Part 2:**

void main()

{

char chaine[N];

Remplir normal d’une chaine de caractère de longueur variable

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

gets(chaine);

Appel de la fonction

if (test(chaine) == 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**

Programme 2

#include <stdio.h>

**Part 1:**

int MAX(int a,int b)

{

« else » ici n’est pas nécessaire à mettre

Car on sort de la fonction lors de premier return si on entre dans if :

c.à.d. « return » arrête l’exécution de la fonction car il retourne une valeur « a » et au cas contraire on continue pour retourner « b ».

if (a > b) return a;

else return b;

}

**Part 2:**

void main()

{

printf("max de 4 et 5 est %d", MAX(4, 5));

}

**Part 3:**

void main()

{

int a, max, N, i;

printf("Entrer le nombre de tes entier puis donner tes entier!\n");

scanf("%d%d", &N, &max);

On a pris un premier élément comme un max c.à.d. il reste N-1 éléments à saisir.

for (i = 1; i <= N-1; i++)

{

scanf("%d", &a);

max = MAX(a, max);

}

printf("Ton maximum est %d",max);

}

Exemple d’exécution :

**Entrer le nombre de tes entier puis donner tes entier!**

**5**

**3 20 17 6 4**

**Ton maximum est 20**

Programme 3

#include <stdio.h>

**Part 1:**

int abandant(int a)

{

Les variables locales déclarées sont indépendantes des variables présentes dans autre fonctions les valeurs par exemple « i » dans « abandant » est diffèrent que « i » dans « main » et ont des valeurs différentes.

int i, sd = 0;

for (i = 1; i <= a; i++)

if (a%i == 0)

sd = sd + i;

if (sd > 2 \* a)

return 1;

return 0;

}

**Part 2:**

void main()

{

printf("%d est abandant si cela egale a un : %d ", 12, abandant(12));

}

**Part 3:**

void main()

{

int a, b, i;

printf("donner 2 entier a puis b tel que a<b\n");

scanf("%d%d", &a, &b);

printf("les entier abandant dans l'intervalle [%d,%d] sont ", a, b);

for ( i = a; i <= b; i++)

if (abandant(i))

printf(" %d, ",i);

}

Exemple d’exécution :

**donner 2 entier a puis b tel que a<b**

**1**

**20**

**les entier abandant dans l'intervalle [1,20] sont 12, 18, 20,**

Programme 4

#include <stdio.h>

#include <conio.h>

**Part 1:**

int encrypt(char c, int k)

{

if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'))

return c + k;

return c - k;

}

**Part 2:**

void main()

{

int k;

char n1;

printf("donner ton caracter puis ton entier\n");

scanf("%c%d", &n1, &k);

printf("ton valeur encrypte est %c", encrypt(n1,k));

}

Exemple d’exécution :

**donner ton caracter puis ton entier**

**a**

3

**ton valeur encrypte est d**