

TD 07 les fonctions et les procédures

Exercice 01 :

Algorithme exo 01

Partie réservée à la déclaration des variables globales

Fonction Strictpos (x : entier) : booléen ;

Partie pour les variables locales

Début

Si (x < 0) alors

Retourner vrai

Sinon

retourner faux

Fin fonction ;

Algorithme Exo02

Var k : entier ;

Fonction fact (N : entier) : entier

Var

i, F : entier ;

Début

F ← 1 ;

Pour i de 2 à N pas 1 faire

F ← F * i ;

Fin pour

Retourne F ;

FinFonction ;

Début

Ecrire ("Ce programme affiche la factorielle des 10 premiers entiers") ;

Pour k de 1 à 10 pas 1 faire

Ecrire (fact(k)) ;

Ecrire ("\n") ;

Fin pour

Fin.

Algorithme Exo 03 :

Fonction ave(x,y,z : real, c1,c2,c3 : integer) ;

Begin

Return (x*c1+y*c2+z*c3)/(c1+c2+c3)

End

//Principal algorithm

Var a,b,c:real; i,c1,c2,c3:integer;

Av=array [100]of real

Begin

Write ("n?\n");

Read(n);

For(i=1 to n step 1)do

Write (" the grades and the coefficients of the student ",i);

Read(a,b,c, c1,c2,c3);

Av[i]=ave(a,b,c,c1,c2,c3);

Write (" the average is ",av[i]);

Endfor

end

Algorithme Exo04

Var

i : entier ;

Procédure disptab (N:entier)

Var

k : entier ;

Début

Ecrire ("Table de ", N, " : ") ;

Pour k de 1 à 10 pas 1 faire

Ecrire (N, " * ", k, " = ", N * k) ;

Fin pour

FinProc ;

Début

Ecrire ("Ce programme écrit la table de multiplication de 10 nombres ") ;

Pour (i 1 à 10) faire

disptab (i) ;

fin pour

Fin.

Algorithme exo5

Var

A : integer ;

Function sommoddix (n : integer) : integer ;

Var R,S2 : integer ;

Begin

S2←0;

do

N ← ndiv 10 ;

R ← n mod 10 ;

S2 ← S2+ (pow (r, 3)) ;

while (n <> 0);

return S2;

Findfunc;

Function Armstrong(n : integer): boolean;

Begin

if (n= sommoddix(n)) then

return true

else

return false ;

begin // prog principale

zrite (' give a number) ;

end (n) ;

if (Armstrong(n) = vrai) then

write ('this number is armstrong ');

else

write (' it is not ');

endif,

nd .

Exercice 06

Algo exo06 :

Var variable globale ...

Procedure saisiepos(var n : integer) ;//int *n

Var

begin

```

repeat
write ( 'give a positive number \n ' );
read ( n );//scanf( "%d",&*n );
until (n>0);
end proc ;
//var a : integer ;
    Begin
Saisiepos(a) ;//saisiepos(&a)
End.
Exercice 07 :
Algorithme exo7
Var t =array [30] of integer ;
Fonction prime(x : integer) :integer ;
Var i : integer
begin
for (I from 1 to x step 1)do
if (x mod I =0) then
p← p+1 ;
endif
if (p= 2) then
return 1 ;
else
return 0 ;
nd fonc
begin
s ←0 ;
write ( ' filling the array \n' );
for (i from 1 to 30 do)
read (t[i]);
if (prime (t[i] = 1 ) then
s ← s+1 ;
endif
end for ;
write ( ' there are ' , s , 'prime number \n' );
end.
Algorithme Exo8
var L1,L2,i,j : entier ; car : caractère ;
//variables globale
Procédure dessine_ligne (n : entier)
Var i, j : entier // Variables locales
Début
Pour i allant de 1 à n par pas de 1 faire
    Ecrire ( '*' );

Fin proc ;
Début

Ecrire ( 'Donner la longueur et la largeur de
ce rectangle ' );
Lire (L1,L2) ;
Pour (i ←1 à L1) faire
Pour (j ← 1 à L2) faire
    Dessine-ligne(L2) ;
Fin pour ; // ligne i
    Ecrire ( '\n' );

```

```

Fin pour
Fin .
Exo 09 :
Algorithme paire
Var a, b ,som: entier ;
Fonction paire(x : entier) : booléen ;
Var i : entier
Début
Si (x mod 2 = 0) alors
Retourner vrai ;
Sinon
retourner faux ;
fin fonc
début
ecrire ( 'donner deux entiers positif' );
repete
    ecrire ( ' donner a ) ;
    saiestrictpos(&a)
    ecrire ( ' donner b' ) ;
    saiestrictpos(&b) ;
jusqu'à (a<b)
pour (i de a à b pas 1 ) faire
si (paire(i)= 1 ) alors
ecrire (i, 'est un nombre paire' );
finsi ;
fin pour
fin .

```

Solution en C :

Exercice 04 :

```

#include<stdio.h>
/* declaration de procedure*/
void saiestrictpos(int *n)
{
    do
    {
        printf("donner un entier positif");
        scanf("%d",&*n);
    } while(n<=0);
}
/* fonction factorielle */
int fact (int n) {
    int i,f=1;
    for(i=2; i<=n; i++)
        f=f*i;
    return f;
}
int main()
{
    int n, p;
    float a;
    do
    {
        printf(" donner n \n");
        saiestrictpos(&n);
        printf(" donner p \n");
    }
}

```

```

    saisiestrictpos(&p);
} while(n<=p);
    a = fact (n)/fact(n-p);
printf("le nombre d'arrangements est%f:",a);
return 0;
}

```

Exercice 06 :

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

```
int armstrong(int x)
```

```

{
    int a, tmp, ar, somme=0;
    tmp=x;
    while(x>0)
    { a=x%10;
      somme=somme+(a*a*a);
      x=x/10;
    }
    if(tmp==somme)
    { ar =1;
    }
    else
    { ar=0;
    }
    return ar;
}

```

```
int parfait (int x)
```

```

{
    int somme=0, f, i;
    for(i = 1; i <=x/2; ++i)
    {
        if(x %i == 0)
        { somme = somme + i;
        }
    }
    if(somme == x)
    {
        f=1;
    }
    else
    { f=0;
    }
    return f;
}

```

```
int main()
```

```

{
    int nbr, arm,p;
    printf(" Entrez un nombre: ");
    scanf("%d", &nbr);
    arm=armstrong(nbr);
    printf("%d resultats est\n",arm);
    p=parfait(nbr);
    printf("%d\n",p);
    if ((arm ==1)&&(p==1))

```

```

        printf(" %d est un nombre parfait et
Armstrong ",nbr);
    else printf(" %d n'est pas un nombre parfait
et Armstrong ",nbr);
    return 0;
}

```

Exercice 07 :

```
#include<stdlib.h>
```

```
#include<stdbool.h>
```

```
int premier( int x)
```

```

{
    int i, f, p=0;
    for (i=1; i<=x; i++)
    {
        if (x%i==0)
        { p++;
        }
    }
    if (p ==2)
    { f=1;
    }
    else
    { f=0;
    }
    return f;
}

```

```
int main()
```

```

{
    int x,n, i, s=0, t[30];
    int f;
    printf("remplir le tableau\n");
    printf("donner un entier inférieure à 30
\n");
    scanf("%d",&n);
    for(i=0; i<=n; i++)
    {
        printf("entrer un élément \n");
        scanf("%d",&t[i]);
        if (premier (t[i])==1)
        {
            s=s+1;
        }
    }
    printf("le nombre des nombres premiers est
%d \n",s);
    return 0;
}

```

Exercice 09

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

```
int pair (int nbr)
```

```

{
    int p;
    if (nbr %2==0)
    { p=1;
    }
    else
    { p=0;
    }
    return p;
}

```

```

}
int main()
{
    int x,y, i, pr;
    do
    {
        printf("donner deux entiers \n");
        scanf("%d%d",&x,&y);
    }
    while(x>=y);
    for(i=x;i<=y;i++)
    {
        pr =pair (i);
        if (pr==1)
        { printf("%d est un nombre pair \n",i);
        }
    }
    return 0;
}

```

Exercice triangle des étoiles :

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

```

int main()
{ int i,j;
  char c='*';
  printf ("ce programme dessine des étoiles \n
");
    for (i=0;i<=9;i++)
    { for (j=0;j<=i;j++)
      printf (" %c", c);
      printf ("\n" );
    }
    return 0;
}

```