COMMENT FAIRE UNE ANALYSE STRUCTUREE ET MODULAIRE

A CHAQUE NIVEAU SE POSER LES 2 QUESTIONS SUIVANTES

- **QUELLES SONT LES ACTIONS QUE L'ON DOIT FAIRE**
  (LISTE DES ACTIONS)
- **COMBIEN DE FOIS DOIT ON LE FAIRE**
  (NOMBRE DE FOIS CONNU $\Rightarrow$ STRUCTURE POUR)
  (NOMBRE DE FOIS INCONNU $\Rightarrow$ STRUCTURE TANT QUE OU REPETER ET IL FAUT TROUVER LA CONDITION D'ARRET)

EXEMPLE

FAIRE LA SOMME DES DIVISEURS D'UN NOMBRE DONNE N

ANALYSE

PREMIER NIVEAU

- ON GENERE DES DIVISEURS d DE 1 A N DIV 2
  (ICI LE NOMBRE DE DIVISEURS A GENERER EST CONNU)
- A LA FIN ON ECRIT LA SOMME

DEUXIEME NIVEAU C-A-D POUR CHAQUE DIVISEUR d

SI N MOD d = 0 ALORS
    Som ← Som + d

ALGORITHME SomDiv

```
ALGORITHME SomDiv
VAR
        N , d , Som : ENTIER
DEBUT
    LIRE (N)
    Som ← 0
    POUR d ALLANT DE 1 A N Div 2 FAIRE
    DPOUR
        SI N Div d =0 ALORS
            DSI
                Som ← Som+d
            FSI
    FPOUR
    ECRIRE ( Som)
FIN
```

EXEMPLE

Enoncé : Trouver tous les nombres unitairement parfaits inférieurs à un nombre donné.

Un nombre n est unitairement parfait s'il est égal à la somme de ses diviseurs unitaires.

Si d est un diviseur de n et s'il est premier avec n/d, alors d est un diviseur unitaire de n.

Exemple n=60

Diviseurs de n : 1,2,3,4,5,6,10,12,15,20,30

Diviseurs unitaires de n : 1, 3, 4, 5, 12, 15, 20

Leur somme = 60 →

60 est unitairement parfait

ANALYSE

- Soit un nombre X
- On génère des Nombres N de 1 à X

  Pour chaque N généré :

  Si N est unitairement parfait on l'écrit

ALGORITHME

```
ALGORITHME UnitParf
VAR
    X, N: ENTIER
DEBUT
      LIRE (X)
      POUR N ALLANT DE 1 A X FAIRE
        DPOUR
            SAVOIR SI N EST
            UNITAIREMENT PARFAIT

            SI OUI ALORS
                ECRIRE (N)
        FPOUR
 FIN
```

ANALYSE (SUITE)

Pour savoir si un nombre N est unitairement parfait

- On génère ses diviseurs d de 1 à N DIV 2

  Pour chaque d généré :

  Si d divise N ALORS

  Si N DIV d et d sont premier entre eux ALORS

  on cumule d dans S

- Une fois tous les diviseurs générés si S = N ALORS

  N est un nombre unitairement parfait

ALGORITHME

S ← 0

POUR d ALLANT DE 1 A N DIV 2 FAIRE

DPOUR

SAVOIR SI d et N div d SONT PREMIERS ENTRE EUX

SI OUI ALORS

S ← S + d

FPOUR

SI S = N ALORS

ECRIRE (N)

ANALYSE (SUITE)

Pour savoir si 2 nombres A et B sont premiers entre eux on génère des diviseurs C de l'un (A) à partir de 2 jusqu'à ce nombre(A) au plus.

Le processus s'arrête si

(J > A) OU ((A MOD J=0) ET (B MOD J = 0))

(c.-à-d. J dépasse A OU BIEN J divise les deux)

A la fin du processus si aucun diviseur généré ne divise l'autre nombre (B) alors les deux nombres A et B sont premiers entre eux.

ALGORITHME

```
K ← N div D
J ← 2
TANT QUE  (J <= D) ET (K MOD J<> 0)  OU (D MOD J <> 0))  FAIRE
DTQ
     J ← J + 1
FTQ
SI  J > D  ALORS
   DSI
        S ← S + d
   FSI
```

```
ALGORITHME UnitParf
VAR
    X, N, S, d, J, K: ENTIER
DEBUT
        LIRE (X)
        POUR N ALLANT DE 2 A X FAIRE
        DPOUR
                S← 0
                POUR d ALLANT DE 1 A N DIV 2 FAIRE
                DPOUR
                    SI N MOD D  = 0 ALORS
                        DSI
                            K <-- N div D
                            J <--2
                            TANT QUE  (J <= D) ET (K MOD J<> 0)  OU (D MOD J <> 0))  FAIRE
                                    J← J + 1
                            SI  J > D  ALORS
                                    S<-- S + D
                        FSI
                FPOUR
                SI  S=N ALORS
                    ECRIRE (N)
        FPOUR
FIN
```

File Edit Search Run Compile Debug Tools Options Window Help

UniP.pas

```
Program UnitParf;
VAR X,N,S,d,J,K : longint;
Begin
    write ('Donnez une limite : ');
    Readln (X);
    For N:= 1 to x  do
    begin
        s:= 0;
        for d:= 1 to n div 2  do
        begin
            if n mod d = 0  then
            begin
                j := 2;
                while (j <= d ) and ((d mod j <> 0) or ((n div d) mod j <> 0)) do
                    j := j+1;
                if (j > d)  then
                    s:= s + d;
            end ;
        end;
        if s=n then
            writeln (n);
    end ;
    writeln ('Fini!!');
    readln;
end .
```

ANALYSE AVEC MODULARITE

- Soit un nombre X
- On génère des Nombres N de 1 à X

  Pour chaque N généré :

  Si N est unitairement parfait on l'écrit

ALGORITHME

```
ALGORITHME UnitParf
VAR
    X, N: ENTIER
FONCTION UNIT_PARFAIT (N : ENTIER) : BOOLEEN
DEBUT
      LIRE (X)
      POUR N ALLANT DE 2 A X FAIRE
      DPOUR
              SI UNIT_PARFAIT (N) ALORS
                      ECRIRE (N)
      FPOUR

 FIN
```

MODULE UNIT_PARFAIT

FONCTION

N : Entier → UNIT_PARFAIT → : Booléen

Rôle : Examine si N est unitairement parfait

ANALYSE

On génère les diviseurs d de N de 1 à N DIV 2
Pour chaque d généré :
Si d divise N ALORS
Si N DIV d et d sont premier entre eux ALORS
on cumule d dans S
Une fois tous les diviseurs générés si S = N ALORS
N est un nombre unitairement parfait

MODULE UNIT_PARFAIT

ALGORITHME

```
FONCTION UNIT_PARFAIT (N:ENTIER) : BOOLEEN
VAR
      S , D : ENTIER
FONCTION PREMENTREUX (A,B : ENTIER) : BOOLEEN
DEBUT
        S← 0
        POUR d ALLANT DE 1 A N DIV 2 FAIRE
        DPOUR
                SI N Mod d = 0 ALORS
                          SI PREMENTREUX (N, N Div D) ALORS
                                        S← S + D
         FPOUR
         UNIT_PARFAIT ←  S = N
FIN
```

MODULE PREMENTREUX

FONCTION

A,B : Entier → **PREMENTREUX** → **: Booléen**

Rôle : Examine si deux entiers A et B sont premiers entre eux

ANALYSE

Pour savoir si 2 nombres A et B sont premiers entre eux
on génère des diviseurs D de l'un (A) à partir de 2 jusqu'à ce nombre(A) au plus.
si aucun diviseur généré ne divise l'autre nombre (B) alors les deux nombres A et B sont premiers entre eux.
Autrement dit , le processus de génération des diviseurs s'arrête si (A mod D = 0 ET B mod D = 0) OU (D > A) et à la fin du processus A et B sont premiers entre eux si aucun diviseur généré ne divise l'autre (B) (Cad D > A)

MODULE PREMENTREUX

ALGORITHME

```
FONCTION PREMENTREUX (A,B:ENTIER) : BOOLEEN
VAR
      D : ENTIER
DEBUT
       D← 2
       TANT QUE  (D <= A ) ET ( (B MOD D <> 0) OU (A MOD D <> 0) )  FAIRE
              D ← D+1
        PREMENTREUX ←  D > A
FIN
```

Free Pascal IDE

File Edit Search Run Compile Debug Tools Options Window Help

UniP21.pas

```pascal
program UnitParf;
USES CRT;
VAR X,N : longint;  (* Variables globales *)
function Unit_Parfait( N : Longint) : Boolean;
var d , Som : longint; (* Variables locales Unit_Parfait *)
Function PremEntreux ( A,B: longint ): boolean;
Var j : longint;  (* Variables locales PremEntreux*)
begin    (* Début de la fonction PremEntreux *)
   j := 2;
   while (j <= A  ) And (( A mod j <> 0) Or (B mod j <> 0)) do
       j := j+1;
   prementreux := j > A ; (* Si j > A Alors PremEntreux = Vrai sinon Faux*)
end ;
begin   (* Début de la fonction Unit_Parfait *)
   Som:= 0;
   for d:= 1 to n div 2  do
       if n mod d = 0  then
          if premEntreux ( d, n div d)  then
             Som := Som + d;
   Unit_Parfait := Som = N;   (* Si Som = N  Alors Unit_Parfait = Vrai sinon Faux*)
end ;

Begin (*  Début du programme principal *)
    CLRSCR;
    Write ( 'Donnez une limite : ');
    Readln (x);
    For N:= 2 to x  do
    begin
        if Unit_Parfait(N) then
           writeln (n);
    end ;
    writeln ('Fini!!');
    readln;
end .
```

35:60

Free Pascal IDE

```
Donnez une limite : 100000
6
60
90
87360
Fini!!
```