EMD2 d'ALGORITHMIQUE

Date : mardi 29 janvier 2013
Durée : 3 Heures

Toute circulation de supports magnétiques (flash-disc, disques externe, smartphone,…) est INTERDITE

EXERCICE

Nous disposons d'un tableau X à une dimension et contenant des nombres entiers dont certains appartiennent à la suite de Fibonacci (1, 1,2, 3, 5,…) , d'autres ont une partie centrale composée des mêmes chiffres (exples : 1722259, 56777734) et des intrus, qui n'appartiennent à aucune des deux catégories précédemment citées.

Travail à Faire :

1. **Partie A (à traiter complètement) : conception (10 pts) et réalisation (8 points)**

Nous souhaitons construire le tableau (R) à deux dimensions de la façon suivante.

- La 1ère ligne contiendra tous les éléments de X appartenant à la suite de fibonnacci
- La 2ième ligne contiendra tous les éléments dont la partie centrale est composée des mêmes chiffres
- La 3ième ligne contiendra tous les intrus

Mais **ATTENTION**, les lignes de R doivent être triées par ordre croissant et affichées selon le format suivant :

```
     1       13       89      144     6765   832040
 15559    22222   234478        0        0        0
   479     4568     4768        0        0        0
```

Jeu d'essai final : le tableau donné X

```
Donnez la taille de votre tableau : 12
t [  1] = 144
t [  2] = 479
t [  3] = 13
t [  4] = 1
t [  5] = 832040
t [  6] = 22222
t [  7] = 6765
t [  8] = 234478
t [  9] = 89
t [ 10] = 4768
t [ 11] = 15559
t [ 12] = 4568
```

2. **Partie B (2 pts)** : Si nous disposons de plusieurs tableaux de la nature de X et on voudrait associer chacun d'eux à son tableau résultat (R) et les garder dans un enregistrement, Donnez la description de cet enregistrement.

NOTA : afin de simplifier la correction de la partie programmation, construisez vos modules un à un, mais laissez-les sous forme de modules internes dans votre programme principal à l'exception des modules déjà existants qui doivent être externes. Ne mettez pas vos programmes dans vote copie.

NOUBLIEZ PAS, à la fin de l'épreuve :

1. **d'enlever les écrans de veille et SURTOUT les mots de passe**
2. **de laisser votre copie devant votre machine**
3. **d'utiliser le jeu d'essai final, donné ci-dessus , et de laisser votre écran figé sur le résultat obtenu**

ATTENTION: Votre solution doit **ABSOLUMENT** tenir compte du **formalisme étudié** en cours et utiliser la modularité ! De plus, deux (2) points seront déduits de la note pour toute copie non soignée.

CORRIGE

DECOUPAGE MODULAIRE

Le découpage ne présente aucune difficulté particulière. Il faut :

- lire la table initiale qui contient les nombres donnés (module Lect1D)
- trouver les nombres qui appartiennent à la suite de Fibonacci (module ElemFibo)
- trouver les nombres qui ont une partie centrale composée des mêmes chiffres (module PartCent)
- trier des lignes (module Tri_Sel ou n'importe quel autre module de tri disponible)
- le remplissage du tableau R , étant particulier, il se fera dans l'algorithme principal
- afficher le tableau R à 2 dimensions (Module Ecrit2d)

Les modules Lect1d, Tri_sel et ecrit2d étant disponibles, il nous faut construire les modules PartCent et ElemFibo.

Pour construire ElemFibo , il faut :

- trouver le Nième élément de la suite de Fibonacci (module Fibo)
- disposer des 45 éléments de la suite de Fibonacci en les mettant dans une table (module SuitFibo)

le module Fibo est en principe disponible (c'est l'un des premiers exos dans les séries de TD proposées), il nous faut construire le module SuitFibo.

Fibo(n :ntier) : longint *// donne le nième élément de la suite de Fibonacci*

Pour construire le module PartCent, il faut :

- connaître le nombre de positions d'un nombre N (module Nb_Pos)
- savoir comment extraire des positions d'un nombre (module Extr_Nb)

Ces 2 modules sont en principe disponibles.

Nb_pos(n : longint) : longint *// donne le nombre de positions de N*

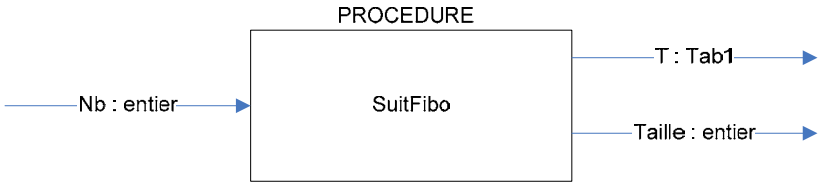
Extr_nb(n,c,p :longint) : longint *// extrait de N un nombre de c positions à partie de la position p (incluse)*

Types utilisés :

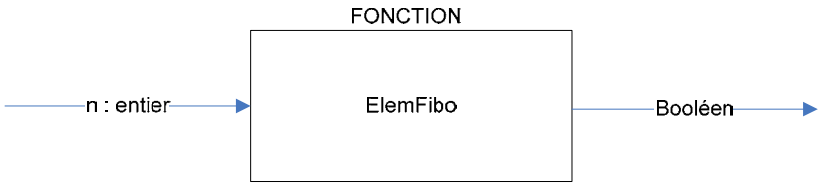
type Tab1 = Tableau[1..1000] d'entiers

Tab2 = Tableau [1..1000,1..1000] d'entiers

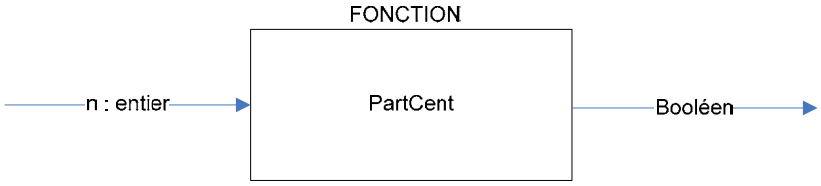
En résumé **sur les 9 modules nécessaires**, six sont disponibles en principe **il nous reste donc 3 modules à construire et qui sont : ElemFibo, suitFibo et PartCent**



Rôle :donne une table qui contient les éléments de la suite de Fibonacci
Attention Nb = 45 au maximum



Rôle : donne vrai si N se trouve parmi les 45 premiers éléments de la suite de Fibonacci



Rôle : donne vrai si la partie centrale de N est composée du même chiffre

Construction du module SuitFibo :
Idée générale : on met dans la table le 1ér élément de la suite, puis le second, puis le troisème….
On ne dépasse pas le 45ième pour la simple raison que le type Longint ne suffit plus, on pourrait utiliser le type Réel , mais le problème devient un peu plus compliqué.
Analyse

- on initialise t[1] et T[2] à 1
- on fait varier i = 3,4,5, …,45
  - t[i] = Fibo[i] *(on met dans le tableau l'élément de rang i de la suite de Fibonacci)*
- taille = n

Algorithme

```
Procedure SuitFibo( n : entier ;var t : tab1; var taille : entier)
variable i: Entier
Fonction fibo

DEBUT
t[1] ← 1
t[2] ← 1
Pour  i allant de 3 à n faire t[i] ← Fibo(i)
Taille ← n
FIN
```

Construction du module ElemFibo :
Idée générale : si l'élément donné se trouve dans la table qui contient le éléments de la suite**,** alors il appartient à la suite de Fibonacci
Analyse

- on construit la table qui contient les 45 premiers éléments de la suite de Fibonacci
  SuitFibo(45, t, tai)
- elemfibo = Faux (on prend un aiguillage qui prendra la valeur faux au départ)
- on parcourt la table avec i : 1, 2, 3 ….
  Si t[i] = n , on met vrai dans elemeFibo
  On s'arrête lorsque (elemeFibo = vrai) ou (i > tai)

Algorithme

```
Function ElemFibo( n:entier): booléen
Variables     t : tab1
              I , tai : Entier
Peocédure SuitFibo

DEBUT
SuitFibo (45, t, tai)
elemfibo ← Faux
i ← 1
Répéter
    Si  t[i] = n Alors elemfibo ←  Vrai
    i ← i+1
Jusqu'à  (elemfibo = Vrai) OU (i>tai)
FIN
```

Construction du module PartCent :
Idée générale : si le nombre de positions de N est pair, il suffit que les 2 chiffres du milieu soient identiques pour dire que la partie centrale de N est composée des mêmes chiffres. De la même manière si le nombre de positions est impair, on ne contrôlera que les 3 chiffres du milieu.
Pour vérifier si les chiffres du milieu sont identiques on pourra soit , utiliser le module MemeCh déjà utilisé en principe, soit vérifier si les 2 ou 3 chiffres du milieu sont égaux à 0 ,soient s'ils sont des multiples de 11 ou de 111 (par exemple dans le cas ou les 2 chiffres du milieu sont identiques on ne peut avoir que 11,22,33,44,….,99).
Aussi on évitera les nombres composés de 2 et de 3 chiffres parce que l'on ne peut en extraire les parties centrales.

- partCent =Faux (*on prend un aiguillage initialisé à faux au départ)*

- Si le nombre de positions est paire et supérieur à 2
  - On extrait le nombre composé des 2 chiffres du milieu
  - Si ce nombre est égal à 0 ou il est divisible par 11 alors PartCent = vrai

Sinon
  - Si le nombre de positions est supérieur à 2
    - On extrait le nombre composé des 3 chiffres du milieu
    - Si ce nombre est égal à 0 ou il est divisible par 111 alors PartCent = vrai

Algorithme

```
Fonction PartCent (n : entire ) : Booléen
Variables  nbext : entier
Fonctions Extr_Nb , Nb_Pos

DEBUT
partCent ← Faux
Si (nb_Pos(n) mod 2 = 0) ET (nb_pos(n) > 2) Alors
        | Dsi
        | Nbext ←  extr_Nb(n, 2, nb_pos(n) div 2)
        | Si (nbext = 0) or (nbext mod 11 = 0) Alors  PartCent ← Vrai
        | Fsi
Sinon    Si  nb_pos(n) >3 Alors
        | Dsi
        | Nbext ←  extr_Nb(n, 3, nb_pos(n) div 2)
        | if (nbext = 0) OU  (nbext mod 111 = 0) Alors  PartCent ← Vrai
        | Fsi
FIN
```

Construction de l'algorithme principal

Idée générale : comme il demandé que les lignes du tableau à 2 dimensions soient triées il suffit pour cela de trier au départ le tableau donné X . Puis on prend les éléments du tableau trié X, un à un on va voir :

- Si cet élément appartient à la suite de Fibonacci on le met dans la 1ère ligne de R
- S'il à une partie centrale composée des mêmes chiffres on le met dans la 2ième ligne de X
- S'il n'est ni l'un , ni l'autre on le met dans la 3ième ligne

A la fin on affiche X . Pour avoir le format désiré, il suffit de reprendre le module Ecrit2d , et de le modifier un peu.

Nota : pour éviter d'avoir des lignes trop longues avec des colonnes qui ne contiennent que des zéros. On affichera que les colonnes qui contiennent effectivement des éléments recherchés.

- On lit le tableau X
  lect1d(x,tai)
- On le trie tri_sel(x,tai)
  tri_sel(x,tai)
- On initialise i à 0 (*i est l'indice de la 1ère ligne qui va contenir les éléments appartenant à la suite)*
- On initialise j à 0 (j *est l'indice de la 2ième ligne des éléments dont la partie centrale et identique)*
- On initialise k à 0 *(k est l'indice de la 3ième ligne qui va contenir les intrus)*
- On fait varier m 1, 2, 3, ….,tai et à chaque fois *(pour prendre un à un les éléments du tableau X)*
  - Si x[m] appartient à la suite de Fibonacci
    - On incrémente i
    - On met x[m] dans R[1,i]
  - Si x[m] a une partie centrale composée des mêmes chiffres
    - On incrémente j
    - On met x[m] dans R[2, j]
  - Si x[m] n'appartient à aucune des 2 catégories précédentes
    - On incremente K
    - On met x[m] dans R[3,k]
- On écrit R
  Ecrit2d(R,3,m)

```
Algorithme  ed2_1213;
type    Tab1 = Tableau [1..1000] d'entiers
         Tab2 =  Tableau  [1..1000,1..1000] d'entiers
Variables      x : tab1
               tai,i,j,k,m : entier
               R : tab2
Procédures lect1d , tri_sel , ecrit2d
Fonctions PartCent ,  ElemFibo

DEBUT
lect1d(x,tai)
tri_sel(x,tai)
I ← 0
J ← 0
k← 0
Pour  m allant de 1 à tai Faire
      Dpour
      Si  elemFibo(x[m]) Alors
                Dsi
                I ← i+1
                R[1,i] ← x[m]
                Fsi
      Sinon
           Si PartCent(x[m]) Alors
                Dsi
                J ← j+1
                R[2,j] ← x[m]
                Fsi
           Sinon
                Dsin
                K ← k+1
                R[3,k] ← x[m]
                Fsin
      Fpour
Ecrit2d(R,3,m)
FIN
```

PROGRAMMES

```
program ed2_1213;
uses crt;
type    Tab1=array[1..1000] of longint;
        Tab2=array[1..1000,1..1000] of longint;
var     x:tab1;
        tai,i,j,k,m:integer;
        R:tab2;
{$i E:\Algo\modules\lect1d.pro}
{$i E:\Algo\modules\tri_sel.pro}
{$i E:\Algo\modules\ecrit2d.pro}
{$i E:\Algo\modules\PartCent.fon}
{$i E:\Algo\modules\ElemFibo.fon}
BEGIN
clrscr;
lect1d(x,tai);
tri_sel(x,tai);
i:=0;
j:=0;
k:=0;
for m:=1 to tai do
        Begin
        if elemFibo(x[m]) then
                        Begin
                        i:=i+1;
                        R[1,i]:=x[m];
                        End
        else
```

```
                if PartCent(x[m]) then
                        Begin
                        j:=j+1;
                        R[2,j]:=x[m];
                        End
                else
                        Begin
                        k:=k+1;
                        R[3,k]:=x[m];
                        End;
        End;
if i>j then
        begin
        if i > k then m:=i;
        end
else    if j>k then m:=j
        else m:=k;
writeln;
Ecrit2d(R,3,m);
readln;
END.
// ------------------------------------------------------------------------
procedure SuitFibo(n:integer;var t:tab1;var taille:integer);
//donne une table qui contient les elements de la suite de fibonacci
// Max 45 elements
var i:integer;
{$i E:\Algo\modules\fibo.fon}
BEGIN
t[1]:=1;
t[2]:=1;
for i:=3 to n do t[i]:=Fibo(i);
taille:=n;
END;
// ------------------------------------------------------------------------
Function ElemFibo(n:longint):boolean;
// donne vrai si N est un element de la suite de Fibonacci
// on ne prendra que les 45 premiers elements de la suite, donc N<=45
var     t:tab1;
        i,tai:integer;
{$i E:\Algo\modules\SuitFibo.pro}
BEGIN
SuitFibo(45,t,tai);
elemfibo := false;
i:=1;
repeat
     if t[i]= n then elemfibo:=true;
     i:=i+1;
until (elemfibo=true) or (i>tai);
END;
// ------------------------------------------------------------------------
Function PartCent(n:longint):boolean;
// donne vrai si la partie centrale de N est composee du meme chiffre
Var nbext:longint;
{$i E:\Algo\modules\Extr_Nb.fon}
{$i E:\Algo\modules\Nb_Pos.fon}
BEGIN
partCent:=false;
if (nb_Pos(n) mod 2 = 0) and (nb_pos(n)>2) then
                Begin
                nbext:= extr_Nb(n,2,nb_pos(n)div 2);
                if (nbext = 0) or (nbext mod 11 =0) then PartCent:=true
                End
else
      if nb_pos(n)>3 then
                Begin
                nbext:= extr_Nb(n,3,nb_pos(n)div 2);
                if (nbext = 0) or (nbext mod 111 =0) then PartCent:=true;
                End;
END;
```

BAREME DE NOTATION DE LA COPIE

PARTIE A	analyse	algorithme	total
recherche des nombres qui appartiennent à la suite de Fibonacci (module ElemFibo) en cas de solution différente on vérifiera la cohérence de la solution et l'utilisation de la modularité	1	2	**3**
recherche des nombres qui ont une partie centrale composée des mêmes chiffres (module PartCent) en cas de solution différente on vérifiera la cohérence de la solution et l'utilisation de la modularité	1	2	3
Algorithme principal On vérifiera l'utilisation des modules construits et ceux de lecture d'un tableau à 1 dimension, de tri, d'écriture d'un tableau à 2 dimensions. Mais aussi le remplissage correct du tableau R .	1.5	2.5	**4**
			10
Programmation			8
		TOTAL	18

PARTIE B			total
Déclaration de l'enregistrement : type Tab1 = Tableau [1..1000] d'entiers Tab2 = Tableau [1..1000,1..1000] d'entiers E = Enregistrement X : Tab1 R : tab2 Fin Tab3 = tableau [1..1000] de E Déclaration de E : 1 pt Tab3 : 1 pt Ou 0 sinon			2

NE DONNER EN AUCUN CAS LE MAXIMUM DE LA NOTE EN CAS DE COPIE NON SOIGNEE

//

BAREME DE NOTATION DE LA REALISATION

Résultats : en appliquant le jeu d'essai donné, les résultats sont ceux figurant sur le sujet.

Il y a 5 points de contrôle :

1. éléments de la suite de Fibonacci corrects : 1.5 pts
2. nombre ayant une partie centrale composée des mêmes chiffres : 1.5 pts
3. tris des lignes du tableau final : 2pts
4. utilisation de la modularité (modules interne, externes ou biblio) :2 pts
5. stylistique (programmes lisibles, indentation, bloc visibles, ….) : 1 pt

à l'intérieur d'un point de contrôle, la note peut être modulée en fonction des résultats observés

CPI - 1ère Année - Année Universitaire 2012/2013 - (Semestre 1)

EMD2 d'ALGORITHMIQUE

Date : mardi 29 janvier 2013
Durée : 3 Heures

Toute circulation de supports magnétiques (flash-disc, disques externe, smartphone,…) est INTERDITE

EXERCICE

Nous disposons d'un tableau X à une dimension et contenant des nombres entiers dont certains appartiennent à la suite de Fibonacci (1, 1,2, 3, 5,…) , d'autres ont une partie centrale composée des mêmes chiffres (exples : 1722259, 56777734) et des intrus, qui n'appartiennent à aucune des deux catégories précédemment citées.

Travail à Faire :

1. **Partie A (à traiter complètement) : conception (10 pts) et réalisation (8 points)**

Nous souhaitons construire le tableau (R) à deux dimensions de la façon suivante.

- La 1ère ligne contiendra tous les éléments de X appartenant à la suite de fibonnacci
- La 2ième ligne contiendra tous les éléments dont la partie centrale est composée des mêmes chiffres
- La 3ième ligne contiendra tous les intrus

Mais **ATTENTION**, les lignes de R doivent être triées par ordre croissant et affichées selon le format suivant :

```
     1       13       89      144     6765   832040
 15559    22222   234478        0        0        0
   479     4568     4768        0        0        0
```

Jeu d'essai final : le tableau donné X

```
 Donnez la taille de votre tableau : 12
t [  1] = 144
t [  2] = 479
t [  3] = 13
t [  4] = 1
t [  5] = 832040
t [  6] = 22222
t [  7] = 6765
t [  8] = 234478
t [  9] = 89
t [ 10] = 4768
t [ 11] = 15559
t [ 12] = 4568
```

2. **Partie B (2 pts)** : Si nous disposons de plusieurs tableaux de la nature de X et on voudrait associer chacun d'eux à son tableau résultat (R) et les garder dans un enregistrement, Donnez la description de cet enregistrement.

NOTA : afin de simplifier la correction de la partie programmation, construisez vos modules un à un, mais laissez-les sous forme de modules internes dans votre programme principal à l'exception des modules déjà existants qui doivent être externes. Ne mettez pas vos programmes dans vote copie.

NOUBLIEZ PAS, à la fin de l'épreuve :

1. **d'enlever les écrans de veille et SURTOUT les mots de passe**
2. **de laisser votre copie devant votre machine**
3. **d'utiliser le jeu d'essai final, donné ci-dessus , et de laisser votre écran figé sur le résultat obtenu**

ATTENTION: Votre solution doit **ABSOLUMENT** tenir compte du **formalisme étudié** en cours et utiliser la modularité ! De plus, deux (2) points seront déduits de la note pour toute copie non soignée.

CORRIGE

DECOUPAGE MODULAIRE

Le découpage ne présente aucune difficulté particulière. Il faut :

- lire la table initiale qui contient les nombres donnés (module Lect1D)
- trouver les nombres qui appartiennent à la suite de Fibonacci (module ElemFibo)
- trouver les nombres qui ont une partie centrale composée des mêmes chiffres (module PartCent)
- trier des lignes (module Tri_Sel ou n'importe quel autre module de tri disponible)
- le remplissage du tableau R , étant particulier, il se fera dans l'algorithme principal
- afficher le tableau R à 2 dimensions (Module Ecrit2d)

Les modules Lect1d, Tri_sel et ecrit2d étant disponibles, il nous faut construire les modules PartCent et ElemFibo.

Pour construire ElemFibo , il faut :

- trouver le Nième élément de la suite de Fibonacci (module Fibo)
- disposer des 45 éléments de la suite de Fibonacci en les mettant dans une table (module SuitFibo)

le module Fibo est en principe disponible (c'est l'un des premiers exos dans les séries de TD proposées), il nous faut construire le module SuitFibo.

Fibo(n :ntier) : longint *// donne le nième élément de la suite de Fibonacci*

Pour construire le module PartCent, il faut :

- connaître le nombre de positions d'un nombre N (module Nb_Pos)
- savoir comment extraire des positions d'un nombre (module Extr_Nb)

Ces 2 modules sont en principe disponibles.

Nb_pos(n : longint) : longint *// donne le nombre de positions de N*

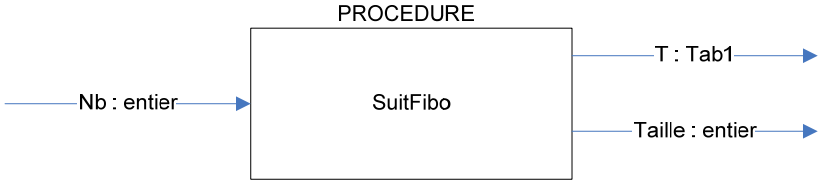
Extr_nb(n,c,p :longint) : longint *// extrait de N un nombre de c positions à partie de la position p (incluse)*

Types utilisés :

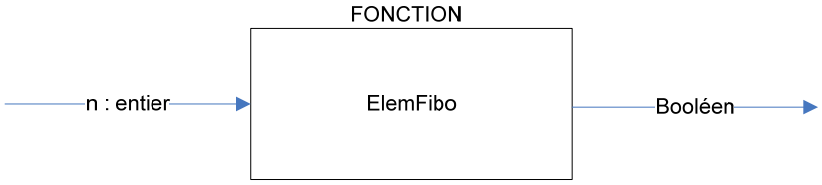
type Tab1 = Tableau[1..1000] d'entiers

Tab2 = Tableau [1..1000,1..1000] d'entiers

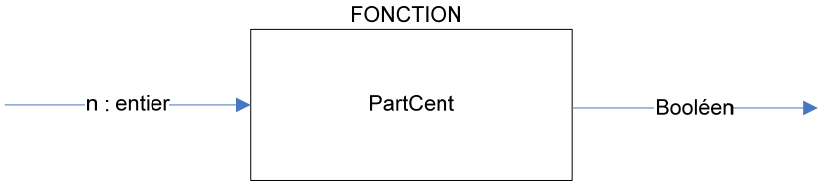
En résumé **sur les 9 modules nécessaires**, six sont disponibles en principe **il nous reste donc 3 modules à construire et qui sont : ElemFibo, suitFibo et PartCent**



Rôle :donne une table qui contient les éléments de la suite de Fibonacci
Attention Nb = 45 au maximum



Rôle : donne vrai si N se trouve parmi les 45 premiers éléments de la suite de Fibonacci



Rôle : donne vrai si la parti centrale de N est comosée du même chiffre

Construction du module SuitFibo :
Idée générale : on met dans la table le 1ér élément de la suite, puis le second, puis le troisème….
On ne dépasse pas le 45^{ième} pour la simple raison que le type Longint ne suffit plus, on pourrait utiliser le type Réel , mais le problème devient un peu plus compliqué.
Analyse

- on initialise t[1] et T[2] à 1
- on fait varier i = 3,4,5, …,45
  - t[i] = Fibo[i] *(on met dans le tableau l'élément de rang i de la suite de Fibonacci)*
- taille = n

Algorithme

```
Procedure SuitFibo( n : entier ;var t : tab1; var taille : entier)
variable i: Entier
Fonction fibo

DEBUT
t[1] ← 1
t[2] ← 1
Pour  i allant de 3 à n faire t[i] ← Fibo(i)
Taille ← n
FIN
```

Construction du module ElemFibo :
Idée générale : si l'élément donné se trouve dans la table qui contient le éléments de la suite **,** alors il appartient à la suite de Fibonacci
Analyse

- on construit la table qui contient les 45 premiers éléments de la suite de Fibonacci
  SuitFibo(45, t, tai)
- elemfibo = Faux (on prend un aiguillage qui prendra la valeur faux au départ)
- on parcourt la table avec i : 1, 2, 3 ….
  Si t[i] = n , on met vrai dans elemeFibo
  On s'arrête lorsque (elemeFibo = vrai) ou (i > tai)

Algorithme

```
Function ElemFibo( n:entier): booléen
Variables     t : tab1
              I , tai : Entier
Peocédure SuitFibo

DEBUT
SuitFibo (45, t, tai)
elemfibo ← Faux
i ← 1
Répéter
    Si  t[i] = n Alors elemfibo ←  Vrai
    i ← i+1
Jusqu'à  (elemfibo = Vrai) OU (i>tai)
FIN
```

Construction du module PartCent :
Idée générale : si le nombre de positions de N est pair, il suffit que les 2 chiffres du milieu soient identiques pour dire que la partie centrale de N est composée des mêmes chiffres. De la même manière si le nombre de positions est impair, on ne contrôlera que les 3 chiffres du milieu.
Pour vérifier si les chiffres du milieu sont identiques on pourra soit , utiliser le module MemeCh déjà utilisé en principe, soit vérifier si les 2 ou 3 chiffres du milieu sont égaux à 0 ,soient s'ils sont des multiples de 11 ou de 111 (par exemple dans le cas ou les 2 chiffres du milieu sont identiques on ne peut avoir que 11,22,33,44,….,99).
Aussi on évitera les nombres composés de 2 et de 3 chiffres parce que l'on ne peut en extraire les parties centrales.

- partCent =Faux (*on prend un aiguillage initialisé à faux au départ)*

- Si le nombre de positions est paire et supérieur à 2
  - On extrait le nombre composé des 2 chiffres du milieu
  - Si ce nombre est égal à 0 ou il est divisible par 11 alors PartCent = vrai

Sinon
  - Si le nombre de positions est supérieur à 2
    - On extrait le nombre composé des 3 chiffres du milieu
    - Si ce nombre est égal à 0 ou il est divisible par 111 alors PartCent = vrai

Algorithme

```
Fonction PartCent (n : entire ) : Booléen
Variables  nbext : entier
Fonctions Extr_Nb , Nb_Pos

DEBUT
partCent ← Faux
Si (nb_Pos(n) mod 2 = 0) ET (nb_pos(n) > 2) Alors
        | Dsi
        | Nbext ←  extr_Nb(n, 2, nb_pos(n) div 2)
        | Si (nbext = 0) or (nbext mod 11 = 0) Alors  PartCent ← Vrai
        | Fsi
Sinon    Si  nb_pos(n) >3 Alors
        | Dsi
        | Nbext ←  extr_Nb(n, 3, nb_pos(n) div 2)
        | if (nbext = 0) OU  (nbext mod 111 = 0) Alors  PartCent ← Vrai
        | Fsi
FIN
```

Construction de l'algorithme principal

Idée générale : comme il demandé que les lignes du tableau à 2 dimensions soient triées il suffit pour cela de trier au départ le tableau donné X . Puis on prend les éléments du tableau trié X, un à un on va voir :

- Si cet élément appartient à la suite de Fibonacci on le met dans la 1ère ligne de R
- S'il à une partie centrale composée des mêmes chiffres on le met dans la 1ième ligne de X
- S'il n'est ni l'un , ni l'autre on le met dans la 3ième ligne

A la fin on affiche X . Pour avoir le format désiré, il suffit de reprendre le module Ecrit2d , et de le modifier un peu.

Nota : pour éviter d'avoir des lignes trop longues avec des colonnes qui ne contiennent que des zéros. on affichera que les colonne qui contiennent effectivement des éléments recherchés.

- On lit le tableau X
  lect1d(x,tai)
- On le trie tri_sel(x,tai)
  tri_sel(x,tai)
- On initialise i à 0 (*i est l'indice de la 1ère ligne qui va contenir les éléments appartenant à la suite)*
- On initialise j à 0 (j *est l'indice de la 2ième ligne des éléments dont la partie centrale et identique)*
- On initialise k à 0 *(k est l'indice de la 3ième ligne qui va contenir les intrus)*
- On fait varier m 1, 2, 3, ….,tai et à chaque fois *(pour prendre un à un les éléments du tableau X)*
  - Si x[m] appartient à la suite de Fibonacci
    - On incrémente i
    - On met x[m] dans R[1,i]
  - Si x[m] a une partie centrale composée des mêmes chiffres
    - On incrémente j
    - On met x[m] dans R[2, j]
  - Si x[m] n'appartient à aucune des 2 catégories précédentes
    - On incremente K
    - On met x[m] dans R[3,k]
- On écrit R
  Ecrit2d(R,3,m)

```
Algorithme  ed2_1213;
type    Tab1 = Tableau [1..1000] d'entiers
         Tab2 =  Tableau  [1..1000,1..1000] d'entiers
Variables      x : tab1
               tai,i,j,k,m : entier
               R : tab2
Procédures lect1d , tri_sel , ecrit2d
Fonctions PartCent ,  ElemFibo

DEBUT
lect1d(x,tai)
tri_sel(x,tai)
I ← 0
J ← 0
k← 0
Pour  m allant de 1 à tai Faire
      Dpour
      Si  elemFibo(x[m]) Alors
                Dsi
                I ← i+1
                R[1,i] ← x[m]
                Fsi
      Sinon
           Si PartCent(x[m]) Alors
                Dsi
                J ← j+1
                R[2,j] ← x[m]
                Fsi
           Sinon
                Dsin
                K ← k+1
                R[3,k] ← x[m]
                Fsin
      Fpour
Ecrit2d(R,3,m)
FIN
```

PROGRAMMES

```
program ed2_1213;
uses crt;
type    Tab1=array[1..1000] of longint;
        Tab2=array[1..1000,1..1000] of longint;
var     x:tab1;
        tai,i,j,k,m:integer;
        R:tab2;
{$i E:\Algo\modules\lect1d.pro}
{$i E:\Algo\modules\tri_sel.pro}
{$i E:\Algo\modules\ecrit2d.pro}
{$i E:\Algo\modules\PartCent.fon}
{$i E:\Algo\modules\ElemFibo.fon}
BEGIN
clrscr;
lect1d(x,tai);
tri_sel(x,tai);
i:=0;
j:=0;
k:=0;
for m:=1 to tai do
        Begin
        if elemFibo(x[m]) then
                        Begin
                        i:=i+1;
                        R[1,i]:=x[m];
                        End
        else
```

```
                if PartCent(x[m]) then
                        Begin
                        j:=j+1;
                        R[2,j]:=x[m];
                        End
                else
                        Begin
                        k:=k+1;
                        R[3,k]:=x[m];
                        End;
        End;
if i>j then
        begin
        if i > k then m:=i;
        end
else    if j>k then m:=j
        else m:=k;
writeln;
Ecrit2d(R,3,m);
readln;
END.
// --------------------------------------------------------------------------
procedure SuitFibo(n:integer;var t:tab1;var taille:integer);
//donne une table qui contient les elements de la suite de fibonacci
// Max 45 elements
var i:integer;
{$i E:\Algo\modules\fibo.fon}
BEGIN
t[1]:=1;
t[2]:=1;
for i:=3 to n do t[i]:=Fibo(i);
taille:=n;
END;
// --------------------------------------------------------------------------
Function ElemFibo(n:longint):boolean;
// donne vrai si N est un element de la suite de Fibonacci
// on ne prendra que les 45 premiers elements de la suite, donc N<=45
var     t:tab1;
        i,tai:integer;
{$i E:\Algo\modules\SuitFibo.pro}
BEGIN
SuitFibo(45,t,tai);
elemfibo := false;
i:=1;
repeat
     if t[i]= n then elemfibo:=true;
     i:=i+1;
until (elemfibo=true) or (i>tai);
END;
// --------------------------------------------------------------------------
Function PartCent(n:longint):boolean;
// donne vrai si la partie centrale de N est composee du meme chiffre
Var nbext:longint;
{$i E:\Algo\modules\Extr_Nb.fon}
{$i E:\Algo\modules\Nb_Pos.fon}
BEGIN
partCent:=false;
if (nb_Pos(n) mod 2 = 0) and (nb_pos(n)>2) then
                Begin
                nbext:= extr_Nb(n,2,nb_pos(n)div 2);
                if (nbext = 0) or (nbext mod 11 =0) then PartCent:=true
                End
else
      if nb_pos(n)>3 then
                Begin
                nbext:= extr_Nb(n,3,nb_pos(n)div 2);
                if (nbext = 0) or (nbext mod 111 =0) then PartCent:=true;
                End;
END;
```

BAREME DE NOTATION DE LA COPIE

PARTIE A	analyse	algorithme	total
recherche des nombres qui appartiennent à la suite de Fibonacci (module ElemFibo) en cas de solution différente on vérifiera la cohérence de la solution et l'utilisation de la modularité	1	2	**3**
recherche des nombres qui ont une partie centrale composée des mêmes chiffres (module PartCent) en cas de solution différente on vérifiera la cohérence de la solution et l'utilisation de la modularité	1	2	3
Algorithme principal On vérifiera l'utilisation des modules construits et ceux de lecture d'un tableau à 1 dimension, de tri, d'écriture d'un tableau à 2 dimensions. Mais aussi le remplissage correct du tableau R .	1.5	2.5	**4**
			10
Programmation			8
		TOTAL	18

PARTIE B			total
Déclaration de l'enregistrement : type Tab1 = Tableau [1..1000] d'entiers Tab2 = Tableau [1..1000,1..1000] d'entiers E = Enregistrement X : Tab1 R : tab2 Fin Tab3 = tableau [1..1000] de E Déclaration de E : 1 pt Tab3 : 1 pt Ou 0 sinon			2

NE DONNER EN AUCUN CAS LE MAXIMUM DE LA NOTE EN CAS DE COPIE NON SOIGNEE

//

BAREME DE NOTATION DE LA REALISATION

Résultats : en appliquant le jeu d'essai donné, les résultats sont ceux figurant sur le sujet.

Il y a 5 points de contrôle :

1. éléments de la suite de Fibonacci corrects : 1.5 pts
2. nombre ayant une partie centrale composée des mêmes chiffres : 1.5 pts
3. tris des lignes du tableau final : 2pts
4. utilisation de la modularité (modules interne, externes ou biblio) :2 pts
5. stylistique (programmes lisibles, indentation, bloc visibles, ….) : 1 pt

à l'intérieur d'un point de contrôle, la note peut être modulée en fonction des résultats observés