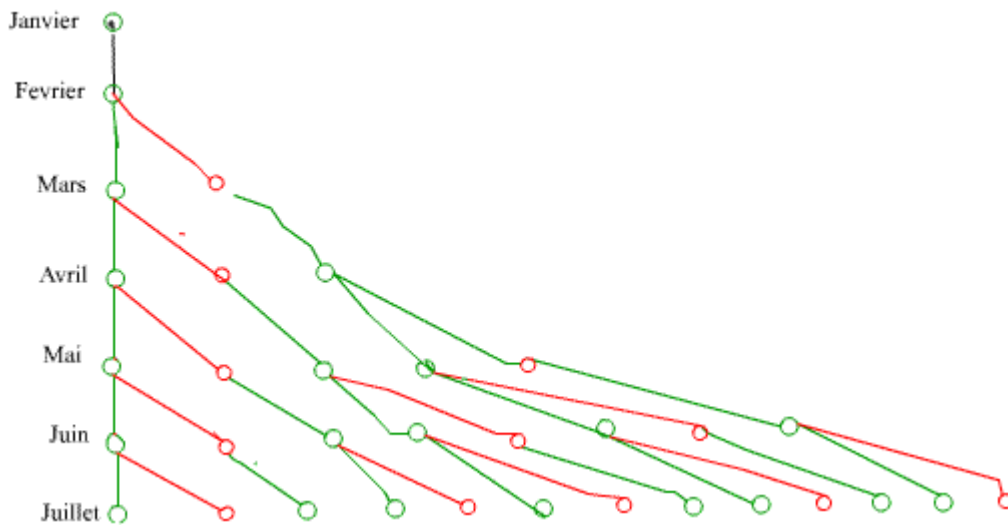


1°) Dessiner un arbre qui illustre bien la situation des 6 premiers mois en utilisant 2 types de branches : \longrightarrow (pour indiquer qu'un couple continue à vivre) et \dashrightarrow (pour indiquer qu'un couple donne naissance à un couple de bébés) .



2°)

n	Fib[n]
1	1
2	1
3	2
4	3
5	5
6	8

3°) Soit $n \geq 3$

$Fib[n]$ = le nombre de couples de lapins vivant dans la garenne au début du mois n = nombre de couples de lapins vivant le mois précédent + le nombre de couples nouveaux-nés de lapins = $Fib[n-1]$ + le nombre de couples nouveaux-nés .

Or le nombre de couples de nouveaux-nés est égal aux nombres de couples de lapins pouvant procréer c'est-à-dire le nombre de couples de lapins existant deux mois auparavant c'est-à-dire $Fib[n-2]$

Donc $Fib[n] = Fib[n-1] + Fib[n-2]$

4°)version 1 avec la structure répétitive FOR

```

program FIBONACCI1 ;
  uses WinCrt;
  var A,B,I,N, NOMBRE_DE_COUPLES : integer;
  begin
    clrscr;
    (* entrée des données *)
    write(' Veuillez taper un nombre de mois N = '); readln(N);
    (* traitement *)
    A := 1;
    B:=1 ;
    If N <= 2 then NOMBRE_DE_COUPLES := 1
  
```

```

        Else
            Begin
                For I :=3 to N do
                    Begin
                        NOMBRE_DE_COUPLES := A + B;
                        A := B;
                        B := NOMBRE_DE_COUPLES;
                    End;
                End;
            End;

(* affichage des résultats*)
writeln('au début du mois ', N, ' le nombre de couples de lapins est ',
        NOMBRE_DE_COUPLES);
end.

```

version 2 avec la structure répétitive while

```

program FIBONACCI2 ;
uses WinCrt;
var A,B,I,N, NOMBRE_DE_COUPLES : integer;
begin
    clrscr;
    (* entrée des données *)
    write('Veuillez taper un nombre de mois N = '); readln(N);
    (* traitement *)
    A := 1;
    B:=1 ;
    If N <= 2 then NOMBRE_DE_COUPLES := 1
        Else
            Begin
                I:=3 ;
                While I <= N do
                    Begin
                        NOMBRE_DE_COUPLES := A + B;
                        A := B;
                        B := NOMBRE_DE_COUPLES;
                        I:= I + 1
                    End;
                End;
            End;

(* affichage des résultats*)
writeln('au début du mois ', N, ' le nombre de couples de lapins est ',
        NOMBRE_DE_COUPLES);
end.

```

version 3 avec l'utilisation d'un tableau

```

program FIBONACCI3 ;
uses WinCrt;
const NOMBREMAXIMUMMOIS = 21;
var I,N, NOMBRE_DE_COUPLES : integer;
    FIB:array[1.. NOMBREMAXIMUMMOIS] of integer;

```

```

begin
  clrscr;

  (* entrée des données *)
  write('Veuillez taper un nombre de mois N = '); readln(N);

  (* traitement *)
  FIB[1] := 1;
  FIB[2] := 1;
  If N <= 2 then NOMBRE_DE_COUPLES := 1
  Else
    Begin
      For I := 3 to N do FIB[I] := FIB[I-1] + FIB[I-2];
      NOMBRE_DE_COUPLES := FIB[N];
    End;

  (* affichage des résultats*)
  writeln('au début du mois ', N, ' le nombre de couples de lapins est ',
  NOMBRE_DE_COUPLES);
end.

```

5°) **version 4 avec l'utilisation d'une fonction récursive**

```

program FIBONACCI1 ;
  uses WinCrt;
  var A,B,I,N, NOMBRE_DE_COUPLES : integer;
  fonction FIBO(X : integer) : integer;
  begin
    if N <= 2 then FIBO := 1
    else FIBO := FIBO(N-1) + FIBO(N - 2)
  end;
begin
  clrscr;
  (* entrée des données *)
  write('Veuillez taper un nombre de mois N = '); readln(N);
  (*traitement et affichage des résultats *)
  writeln('au début du mois ', N, ' le nombre de couples de lapins est ',
  FIBO(N));
end.

```

6°) Sachant qu'un bit est soit 0 ou 1 , qu'un octet est un ensemble de 8 bits

- a) Si un entier était codé sur un bit, on peut représenter en mémoire 2 entiers.
- b) Si un entier était codé sur 2 bits, on peut représenter en mémoire $2^2 = 4$ entiers.
- c) Si un entier était codé sur 3 bits, on peut représenter en mémoire $2^3 = 8$ entiers
- d) Si un entier était codé sur 1 octet, on peut représenter en mémoire $2^8 = 256$ entiers
- e) Si un entier était codé sur 2 octets, on peut représenter en mémoire $2^{16} = 65536$ entiers

7°) En Turbo-Pascal, sachant qu'une variable entier naturel de type word est codé sur 2 octets, si l'on déclare var A,B, NOMBRE_DE_COUPLES : word dans les deux programmes précédents

- a) La fourchette exacte des valeurs disponibles pour N est : 0 .. 65535

b) les programmes précédents ne marcheraient pas pour $N > 23$ car
 $\text{Fib}(22) = 28657$ et $\text{Fib}(23) = 43368$ donc $\text{Fib}(24) = 72025 > 65535 =$ plus grand entier
représentable en mémoire.

8°) En Turbo-Pascal, sachant qu'une variable entier relatif de type integer est codé sur 2 octets,
mais que le bit situé le plus à gauche est réservé au signe (0 pour + et 1 pour -)

si l'on déclare `var A,B, NOMBRE_DE_COUPLES : integer` dans les programmes précédents

a) La fourchette exacte des valeurs disponibles pour N est : -32768 ..32767

b) les programmes précédents ne marcheraient pas pour $N > 22$ car

$\text{Fib}(21) = 14711$ et $\text{Fib}(22) = 28657$ donc $\text{Fib}(23) = 43368 > 32767 =$ plus grand entier
représentable en mémoire.

c) Pour éviter ces 2 écueils, il suffit de déclarer A,B et NOMBRE_DE_COUPLES du type real
car ce type offre une plus grande fourchette de valeurs de $2,9 \cdot 10^{-39}$ à environ $1,7 \cdot 10^{38}$ en
valeur absolue.

9°) Afin d'obliger l'utilisateur à entrer un entier compris entre 1 et 22 :

(* entrée des données *)

repeat

write('Veuillez taper un nombre de mois N = '); readln(N);

until (((N > 0) and (N < 23)) and (N = trunc(N)))

10°) Remplir le tableau suivant :

n	Fib[n]	Valeur exacte de $\frac{\text{Fib}[n+1]}{\text{Fib}[n]}$	Valeur approchée de $\frac{\text{Fib}[n+1]}{\text{Fib}[n]}$
1	1	1	1
2	1	2	2
3	2	3/2	1,5
4	3	5/3	1,666
5	5	8/5	1,6
6	8	13/8	1,625
7	13	21/13	1,615
8	21	34/21	1,619
9	34	55/34	1,617
10	55	89/55	1,618
11	89	144/89	1,617
12	144	233/144	1,618
13	233	377/233	1,618
14	377		

a) la suite (v_n) définie sur \mathbb{N}^* par $v_n = \frac{\text{Fib}[n+1]}{\text{Fib}[n]}$ semble converger vers le nombre d'or $\Phi =$

$$\frac{1 + \sqrt{5}}{2} \text{ dont une valeur approchée est } 1,618$$

b) pour tout $n \geq 2$, l'on a $v_n = \frac{\text{Fib}[n+1]}{\text{Fib}[n]} = \frac{\text{Fib}[n] + \text{Fib}[n-1]}{\text{Fib}[n]} = \frac{\text{Fib}[n]}{\text{Fib}[n]} + \frac{\text{Fib}[n-1]}{\text{Fib}[n]} = 1 + \frac{1}{v_{n-1}}$

c) En supposant que la suite (v_n) converge, alors $L = 1 + 1/L$ donc $L = L + 1 / L$ donc $L^2 - L - 1 = 0$

$$\text{donc } L = \Phi = \frac{1 + \sqrt{5}}{2} \text{ ou } L = \frac{1 - \sqrt{5}}{2}. \text{ Comme } (v_n) \text{ est positive alors } L = \Phi = \frac{1 + \sqrt{5}}{2}$$

Partie B

Soit la suite (u_n) définie sur \mathbb{N} par $u_0 = \frac{3}{2}$ et pour tout entier naturel $u_{n+1} = 1 + \frac{1}{u_n}$

1°) Démontrer par récurrence que pour tout entier naturel n l'on a $\frac{3}{2} \leq u_n \leq 2$

- La propriété est vraie pour $n = 0$ car $u_0 = \frac{3}{2}$ et $\frac{3}{2} \leq u_0 \leq 2$
- Soit k un entier naturel, si $\frac{3}{2} \leq u_k \leq 2$ alors $\frac{1}{2} \leq \frac{1}{u_k} \leq \frac{2}{3}$ donc $1 + \frac{1}{2} \leq 1 + \frac{1}{u_k} \leq 1 + \frac{2}{3}$
donc $\frac{3}{2} \leq u_{k+1} \leq \frac{5}{3}$ donc $\frac{3}{2} \leq u_{k+1} \leq 2$
- En conclusion pour tout entier naturel n l'on a $\frac{3}{2} \leq u_n \leq 2$

2°) Soit l'intervalle $I = [\frac{3}{2}; 2]$ et soit la fonction f définie sur I par $f(x) = 1 + \frac{1}{x}$

$$f(x) = x \Leftrightarrow 1 + \frac{1}{x} = x \Leftrightarrow \frac{1+x}{x} = x \Leftrightarrow x^2 - x - 1 = 0 \Leftrightarrow x = \alpha = \frac{1+\sqrt{5}}{2} \approx 1,618 \text{ ou } x = \beta = \frac{1-\sqrt{5}}{2}$$

Comme $\alpha \in I$ et $\beta \notin I$ donc $S = \{\alpha\}$

3°) pour tout x de I l'on a :

- $f(x) \in I$ $\frac{3}{2} \leq x \leq 2$ alors $\frac{1}{2} \leq \frac{1}{x} \leq \frac{2}{3}$ donc $1 + \frac{1}{2} \leq 1 + \frac{1}{x} \leq 1 + \frac{2}{3}$
donc $\frac{3}{2} \leq f(x) \leq \frac{5}{3}$ donc $\frac{3}{2} \leq f(x) \leq 2$ donc $f(x) \in I$
- $f'(x) = -\frac{1}{x^2}$ donc $|f'(x)| = \frac{1}{x^2}$ Or $\frac{3}{2} \leq x \leq 2$ donc $\frac{9}{4} \leq x^2 \leq 4$ donc $\frac{1}{4} \leq \frac{1}{x^2} \leq \frac{4}{9}$
donc $|f'(x)| \leq \frac{4}{9}$

4°) a) D'après l'inégalité dite des accroissements finis : pour tout $x \in I$ on a $|f'(x)| \leq \frac{4}{9}$ l'on a

$$\alpha \in I \text{ et } u_n \in I \text{ alors } |f(u_n) - f(\alpha)| \leq k |u_n - \alpha|$$

$$\text{donc pour tout entier naturel } n, |u_{n+1} - \alpha| \leq \frac{4}{9} |u_n - \alpha|$$

b) démontrer par récurrence que pour tout entier naturel n , $|u_n - \alpha| \leq (\frac{4}{9})^n |u_0 - \alpha|$

- La propriété est vraie pour $n = 0$ car $u_0 = \frac{3}{2}$ et $\frac{3}{2} \leq \alpha \leq 2$ donc $|u_0 - \alpha| \leq 1 = (\frac{4}{9})^0 |u_0 - \alpha|$

- Soit k un entier naturel, si $|u_k - \alpha| \leq (\frac{4}{9})^k |u_0 - \alpha|$ comme $|u_{k+1} - \alpha| \leq \frac{4}{9} |u_k - \alpha|$

$$\text{alors } |u_{k+1} - \alpha| \leq (\frac{4}{9})^{k+1} |u_0 - \alpha|$$

- En conclusion pour tout entier naturel n l'on a $|u_n - \alpha| \leq (\frac{4}{9})^n |u_0 - \alpha|$

Or $\lim_{n \rightarrow +\infty} (\frac{4}{9})^n = 0$ car $-1 < \frac{4}{9} < 1$ alors $|u_n - \alpha|$ tend vers 0 donc $u_n - \alpha$ tend vers 0 donc

$$\lim_{n \rightarrow +\infty} u_n = \alpha$$