# Pouvoir d’expression des équations récurrentes

- Des nénuphars
- Les tours de Hanoï
- La suite de Fibonnacci
- Le triangle de Pascal
- Dénombrer les partitions
- Equations récurrentes et programmes

# Des nénuphars



la quantité de nénuphars dans le bassin double tous les jours

$$q(n+1) = 2q(n)$$

# Les tours de Hanoï

plan

Pour déplacer une tour de $n+1$ éléments, on déplace les $n$ premiers sur le plot intermédiaire on pose le dernier sur le plot final, et on déplace les $n$ premiers sur celui-là

$$\begin{aligned} H_{n+1} &= 2H_n + 1 \\ H_1 &= 1 \end{aligned}$$

# La suite de Fibonacci

plan

$$\begin{aligned} f(n+2) &= f(n+1) + fn) \\ f(0) &= 1 \\ f(1) &= 1 \end{aligned}$$

# Le triangle de Pascal

plan

Pour faire une combinaison de $p+1$ objets parmi $n+1$, on prend une combinaison de $p$ objets parmi $n$ et on y ajoute un nouvel objet, ou on prend une combinaison de $p+1$ objets parmi $n$ et on n'ajoute pas le nouvel objet :

$$C_{n+1}^{p+1} = C_n^p + C_n^{p+1}$$

# Nombre de partitions

plan

Nombre de partitions d'un ensemble de $n$ éléments en $k$ parties

Pour faire une partition d'un ensemble de $n+1$ éléménts en $k+1$ parties, on prend une partition de $n$ éléments en $k$ parties et on ajoute une partie formée du nouvel élément, ou on prend une partition de $n$ éléments en $k+1$ parties et on ajoute le nouvel élément dans l'une des parties :

$$s_{n+1}^{k+1} = s_n^k + (k+1)s_n^{k+1}$$

# Equations récurrentes et programmes

plan

Forme canonique d'un système d´équations récurrentes :

$$
\begin{aligned}
X(-1) & \\
X(n+1) &= F(X(n), U(n+1)) \\
Y(n+1) &= G(X(n), U(n+1))
\end{aligned}
$$

avec :

- $X$, état du système
- $U$, entrées
- $Y$, sorties
- $F$, fonction de transition,
- $G$, fonction d'observation,

# Programme associé

plan

```
class system {
private :

  state x ;

public :

  system(state xinit) {x = xinit; }

  output compute(input u) {
    output y = g(x, u);
    x = f(x, u);
    return y;
  }
}
```

# Exemple : programmer Hanoï

plan

```
class hanoi {
private :

  int val ;

public :

  hanoi() { val = 0 ; }

  void reset() { val = 0 ; }

  int compute() {
    val = 2 * val + 1;
    return val;
  }
};
```

# Exemple : exécuter Hanoï



```
#include <stdio.h>
#include ‘‘Hanoi.hh’’

main () {
  hanoi h;
  int n = 0 ;

  while (getchar() == ’\n’)
    printf ("hanoi( %d ) = %d \n", ++n, h.compute());
}
```

# Résultats

plan

```
rateau% ./hanoi

hanoi( 1 ) = 1

hanoi( 2 ) = 3

hanoi( 3 ) = 7

hanoi( 4 ) = 15

hanoi( 5 ) = 31

hanoi( 6 ) = 63

hanoi( 7 ) = 127
q
rateau%
```

Systèmes d'équations récurrentes

$\approx$

Programmes objets