

# Matemática

## *Números de Fibonacci e Catalan*

Prof. Edson Alves  
Faculdade UnB Gama

# Números de Fibonacci

O  $n$ -ésimo número de Fibonacci  $F(n)$  é definido pela recorrência

$$\begin{aligned} F(0) &= F(1) = 1 \\ F(n) &= F(n-1) + F(n-2), \quad n \geq 2 \end{aligned}$$

Os primeiros termos da sequência de Fibonacci são:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, ...

# Limites práticos dos números de Fibonacci

- Os números de Fibonacci crescem rapidamente, de modo que o número de termos que podem ser computados em tipos inteiros de C/C++ é bastante restrito
- Para variáveis de *32-bits* é possível calcular o valor exato de  $F(n)$  para  $n \leq 46$  (a saber,  $F(46) = 1836311903$ )
- Para variáveis de *64-bits*, os valores serão exatos para  $n \leq 92$  (observe que  $F(92) = 7540113804746346429$ )
- Para valores de  $n$  superiores a 92, é necessário ou trabalhar com aritmética estendida ou com aritmética modular

# Implementação recursiva dos números de Fibonacci

```
long long recursive_fibonacci(long long n)
{
    if (n == 0 or n == 1)
        return n;

    return recursive_fibonacci(n - 1) + recursive_fibonacci(n - 2);
}
```

- A implementação acima tem como vantagem a simplicidade, uma vez que corresponde à definição apresentada
- Contudo a complexidade assintótica é  $O(2^n)$

# Implementação iterativa em Python

```
def iterative_fibonacci(n):  
  
    if n < 2:  
        return n  
  
    a = 0  
    b = 1  
  
    for _ in range(n):  
        a, b = b, a + b  
  
    return a
```

- Esta versão tem complexidade  $O(n)$
- A linguagem Python implementa nativamente com aritmética estendida, de modo que esta função pode computar  $F(n)$  para  $n > 92$

# Implementação usando programação dinâmica

```
fib = [0, 1]

def fibonacci(n):

    if n < len(fib):
        return fib[n]

    next = len(fib)

    while next <= n:
        fib.append(fib[next - 1] + fib[next - 2])
        next += 1

    return fib[n]
```

# Equações de diferenças lineares

- Os números de Fibonacci podem ser definidos por meio de uma equação de diferenças lineares
- Seja  $u(n)$  um vetor cujas duas componentes são os números de Fibonacci  $F(n+1)$  e  $F(n)$
- Assim, vale que

$$u(n+1) = Au(n),$$

onde

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

# Equações de diferenças lineares

- Observe que  $u(1) = Au(0)$ ,  $u(2) = Au(1) = A^2u(0)$ , etc, e assim por diante
- De fato,

$$u(n) = A^n u(0)$$

- Usando exponenciação rápida para computar  $A^n$ , é possível computar  $F(n)$  em  $O(\log n)$
- Veja que  $F(n)$  ocupará as posições da diagonal secundária de  $A^n$



# Implementação de $F(n)$ em $O(n \log n)$

```
long long fast_fibonacci(long long n)
{
    Matrix res, A(1, 1, 1, 0);

    while (n)
    {
        if (n & 1)
            res = res * A;

        A = A * A;
        n >>= 1;
    }

    return res.b();
}
```

# Propriedades da sequência de Fibonacci

A sequência de Fibonacci tem várias propriedades interessantes:

- A razão entre dois termos consecutivos da série tende à razão áurea, isto é,

$$\lim_{n \rightarrow \infty} \frac{F(n+1)}{F(n)} = \frac{1 + \sqrt{5}}{2}$$

- A soma dos  $n$  primeiros termos da sequência pode ser computada por meio de uma soma telescópica e é igual a

$$\sum_{i=1}^n F(i) = F(n+2) - 1$$

# Propriedades da sequência de Fibonacci

- A soma dos quadrados dos  $n$  primeiros termos da sequência é igual a

$$\sum_{i=1}^n F(i)^2 = F(n)F(n+1)$$

- Para qualquer  $m > 1$  fixo, a sequência dos restos  $r(n, m)$  é cíclica, onde

$$r(n, m) = F(n) \bmod m$$

- O período de  $r(n, m)$  é denominado Período de Pisano  $\pi(m)$

# Período de Pisano

- Alguns valores comuns:
  - $\pi(2) = 3$
  - $\pi(10) = 60$
  - $\pi(100) = 300$
  - $\pi(10^k) = 15 \times 10^{k-1}, k \geq 3$
- Exceto para o caso  $m = 2$ , o período de Pisano é sempre par

# Números de Catalan

Os números de Catalan são definidos pela da recorrência

$$C(n+1) = \sum_{i=0}^n C(i)C(n-i), \quad n \geq 0,$$

e pelo caso base  $C(0) = 1$ .

Os primeiros números de Catalan são

1, 1, 2, 5, 14, 42, 132, 429, 1430, 4862, 16796, 58786, 208012, ...

# Cálculo

- A soma que define a recorrência tem uma fórmula fechada, de modo que

$$C(n) = \frac{1}{n+1} \binom{2n}{n}$$

- Outra recorrência, com o mesmo caso base da recorrência original, decorre desta forma fechada:

$$C(n+1) = \frac{2(2n+1)}{n+2} C(n)$$

# Implementação dos números de Catalan em $O(n)$

```
long long catalan(int n)
{
    if (n == 0)
        return 1;

    if (C[n] != -1)
        return C[n];

    C[n] = (2*(2*n - 1)*catalan(n - 1))/(n + 1);

    return C[n];
}
```

Com variáveis do tipo `long long` é possível computar até o 33º número de Catalan sem *overflow*.

# Aplicação: sequências válidas de pares de parêntesis

- A primeira aplicação notável dos números de Catalan  $C(n)$  é a contagem do número de sequências corretas formadas por  $2n$  pares de parêntesis
- Para  $n = 0$  há uma única sequência: a sequência vazia
- Para  $n = 1$  também existe uma única sequência: `()`
- Para  $n = 2$  há duas sequências possíveis: `()()`, `(( ))`
- Para  $n = 3$  há  $C(3) = 5$  sequências:

`()()()`, `(( ))()`, `()(( ))`, `(( ( )))`, `(( ))()`



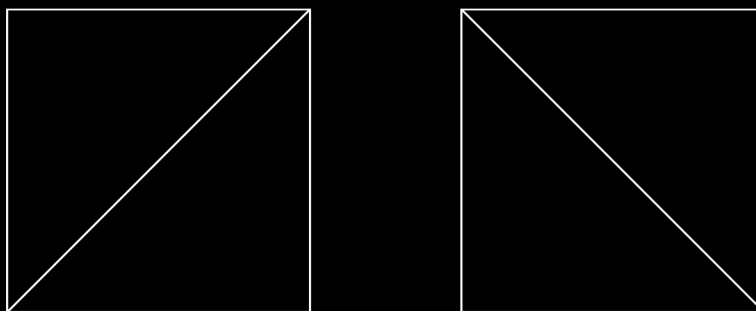
# Aplicação: contagem de árvores binárias completas

- A segunda aplicação notável é a contagem de árvores binárias completas, isto é, cada nó tem ou dois filhos ou nenhum
- Há  $C(n)$  árvores binárias completas com  $n + 1$  folhas
- Para  $n = 3$  são  $C(2) = 2$  árvores:



# Aplicação: triangularização de um polígono convexo

- Uma terceira aplicação seria a contagem de triangularizações de um polígono convexo de  $n + 2$  lados
- Existem  $C(n)$  triangularizações possíveis
- Por exemplo, para um quadrado ( $n = 2$ ) são duas triangularizações distintas:



# Problemas

## 1. OJ

1. [763 - Fibinary Numbers](#)
2. [948 - Fibonaccimal Base](#)
3. [10303 - How Many Trees?](#)
4. [10312 - Expression Bracketing](#)
5. [10689 - Yet another Number Sequence](#)

# Referências

1. **HALIM**, Felix; **HALIM**, Steve. *Competitive Programming 3*, 2010.
2. Wolfram Math World. [Pisano Period](#). Acesso em 28/09/2017.
3. Wikipédia. [Catalan Numbers](#). Acesso em 05/10/2017.
4. Wikipédia. [Pisano Period](#). Acesso em 28/09/2017.
5. Wikipédia. [Sequência de Fibonacci](#). Acesso em 28/09/2017.