# Recursão e Teoria dos Números

Rogerinho

MaratonUSP

2020

# Sumário

# O que é recursão?

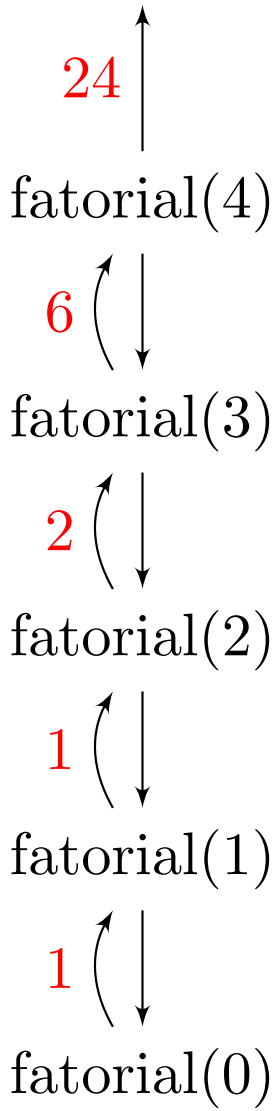
- Recursão, ou coisas recursivas, são coisas que são definidas em termos delas mesmas.

- Exemplo: Fatorial.

$$n! = \begin{cases} 1, & \text{se } n = 0 \\ n \cdot (n-1)!, & \text{se } n \geq 1 \end{cases}$$

Também podemos utilizar recursão em programação. Como construiríamos um programa que calcula fatorial recursivamente?

# O que o programa realmente está fazendo?



# Consumo de tempo

- Uma boa estimativa para o consumo de tempo do algoritmo é contar a quantidade de chamadas da função fatorial em função de $n$.

- Analisando vemos que o algoritmo é $O(n)$.

# Será que conseguimos definir potências de forma recursiva?

Dada uma pontência $a^n$, será que conseguimos definir $a^n$ de forma recursiva?

1. Podemos definir $a^n$ em termos de pontências menores?

2. Temos um caso base?

# Será que conseguimos definir potências de forma recursiva?

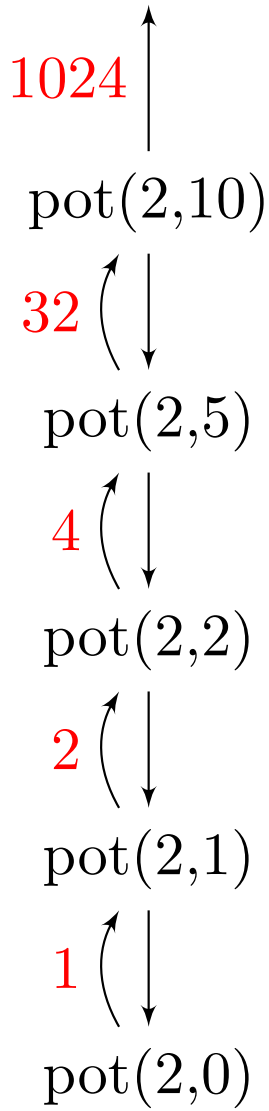
1. Duas opções imediatas:

$$a^n = a \cdot a^{n-1} \tag{1}$$
$$= a^{n/2} \cdot a^{n/2} \tag{2}$$

2. $a^0 = 1$

Como construiríamos um programa recursivo para calcular potências?

# O que o programa realmente está fazendo?



# Consumo de tempo

- Uma boa estimativa para o consumo de tempo do algoritmo é calcular da altura figura anterior em função de $n$.
- Pode-se ver que a altura para a figura é o número $k$ de vezes que temos que dividir $n$ por 2 para que $n$ seja igual a 1, mais a altura entre as chamadas para $n = 1$ e $n = 0$, que é 1. Para $k$ temos:

$$\frac{n}{2^k} = 1 \implies n = 2^k \implies k = \lg n$$

- Portanto a altura da arvore é algo da ordem de $\lg n + 1$ e portanto o algoritmo é $O(\lg n)$

# Maior divisor comum (Greatest common divisor)

- Maior divisor comum ou *gcd* entre dois números $a$ e $b$ é o maior número que divide $a$ e $b$.
- Há mais de 2000 anos ja temos um algoritmo eficiente para isso, o algoritmo de Euclides:

$$gcd(a, b) = \begin{cases} 0, & \text{se } b = 0 \\ gcd(b, a \mod b), & \text{se } b > 0 \end{cases}$$

Como implementar esse algoritmo?

# Consumo de tempo

Esse algoritmo consome tempo de $O(\lg n)$

# Teste de primalidade

- Se um número não é primo, seu menor divisor é um primo

- O menor divisor primo de um número composto $n$ é menor ou igual $\sqrt{n}$

**Prova.**

Seja $n$ um número composto. Seja $k$ o seu menor divisor e suponha que $k > \sqrt{n}$. Então podemos escrever $n = k \cdot q$, para algum $q$. Mas $q \geq k$, o que implica $q \cdot k > \sqrt{n} \cdot \sqrt{n} = |n| = n$, um absurdo. $\square$

Como implementar um algoritmo que utiliza essas duas informações para decidir se um número é primo?

# Consumo de tempo

Como só percorremos possíveis divisores de $n$ até $\sqrt{n}$, então o algoritmo é $O(\sqrt{n})$.

# Todos os divisores inteiros de um número

- Para encontrar todos os divisores inteiros de um número podemos utilizar uma estrutura muito próxima do algoritmo anterior

- Convém notar que os divisores sempre aparecem aos pares

Como implementar um algoritmo que obtém todos os divisores inteiros de um número?

# Consumo de tempo

Assim como o algoritmo anterior, o consumo de tempo deste é $O(\sqrt{n})$.

# Fatorização em números primos

- Ainda utilizando uma estrutura muito próxima, podemos encontrar a fatorização de um número $n$ em números primos

- Utilizaremos o fato de que o menor divisor de um número é um número primo

Como implementar um algoritmo que obtém a fatoração em números primos de um inteiro $n$?

# Consumo de tempo

Note que só procuramos fatores primos até $\sqrt{n}$ e, além disso efetuamos divisões sucessivas em $n$ até que $n$ se torne 1, possivelmente. Vimos anteriormente que o número de divisões sucessivas que fazemos em um número até que ele se torne 1 consome tempo proporcional a $\lg n$. Portanto o consumo e tempo desse algoritmo é proporcional a $\sqrt{n} + \lg n = O(\sqrt{n})$

# Crivo de Eratóstenes (Sieve of Eratosthenes)

- E se quisermos todos os números primos entre 1 e $n$?

- Podemos utlizar o **Crivo de Eratóstenes**.

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

# Crivo de Eratóstenes (Sieve of Eratosthenes)

Seja $n = 30$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |

Como implementaríamos um algoritmo que segue a mesma ideia do crivo?

O consumo de tempo do algoritmo é $O(n \lg \lg n)$.