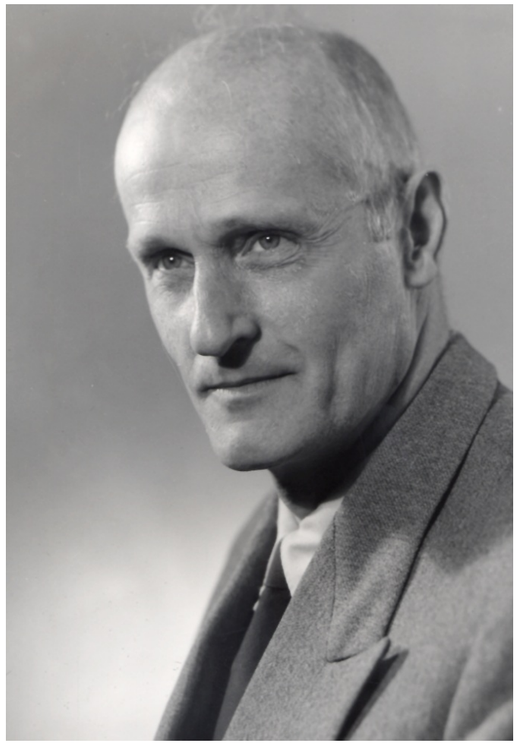
# TEORIA DA COMPUTAÇÃO

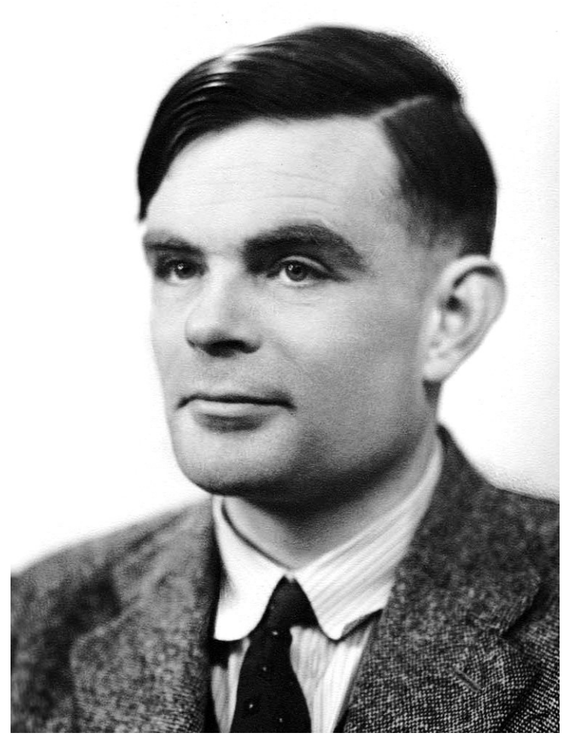
## 1. Linguagens Regulares

## 1.1. Autômatos Finitos

Alonzo Church



Stephen Kleene



Alan Turing

Prof. Abrantes Araújo Silva Filho

# O que é computar?

**Ciência da Computação:**

**É a ciência que projeta e implementa algoritmos para a resolução de problemas.**

**Algoritmo:**

**Uma coleção bem ordenada de operações efetivamente computáveis, definidas e não ambíguas que, quando executada sobre uma entrada produz uma saída e termina em uma quantidade finita de passos e de tempo.**

**É a solução ótima e perfeita para uma classe de problemas.**

**Heurística:**

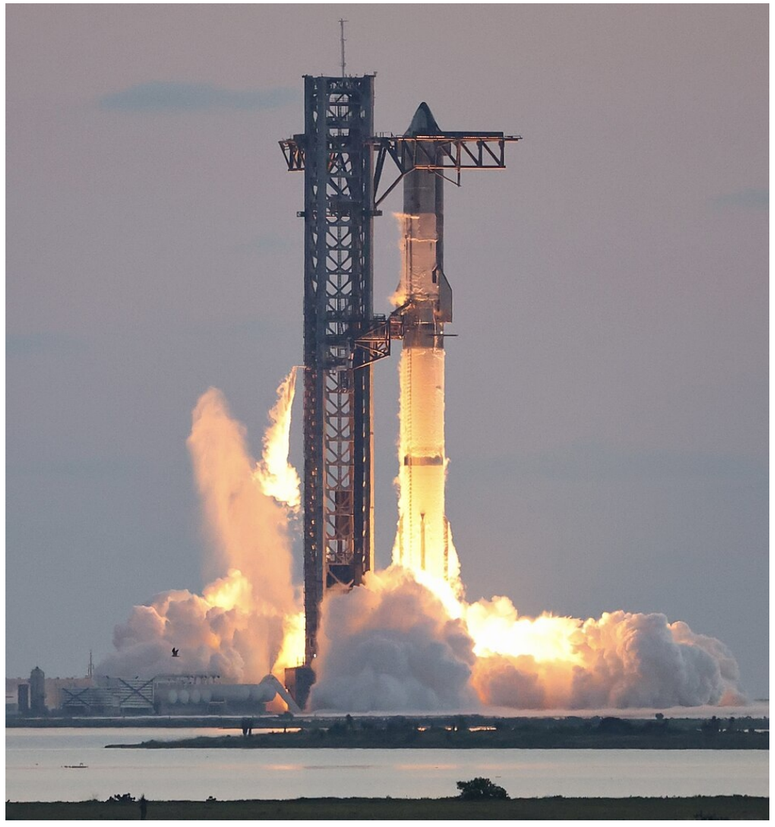
**É uma solução sem garantia de ser ótima e perfeita, mas que é boa o suficiente para nossos propósitos.**

# O que é computar?

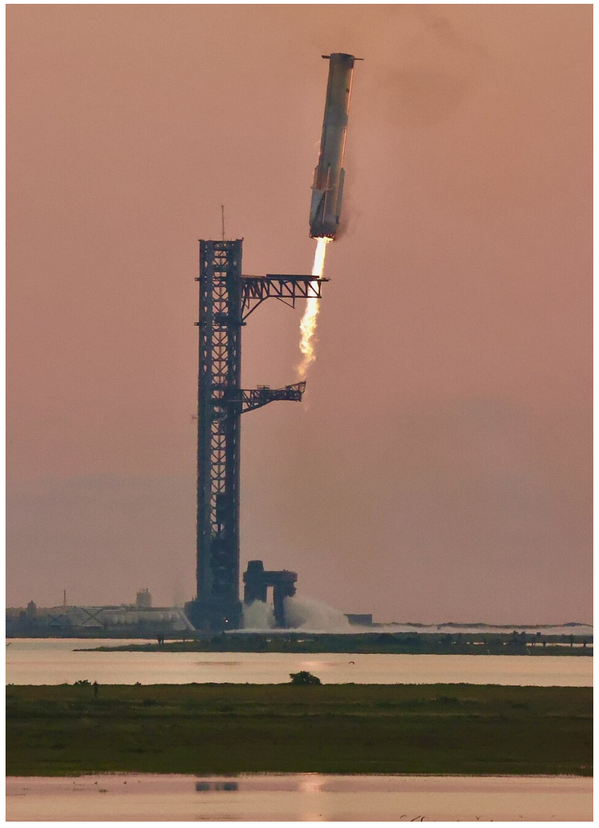
**Computar, por enquanto, é simplesmente a execução de algum algoritmo em um computador, para a resolução de algum problema.**


**Nós precisamos entender como um computador computa. Como fazer isso já que os computadores são extremamente complexos e dependem de interação entre hardware e software?**

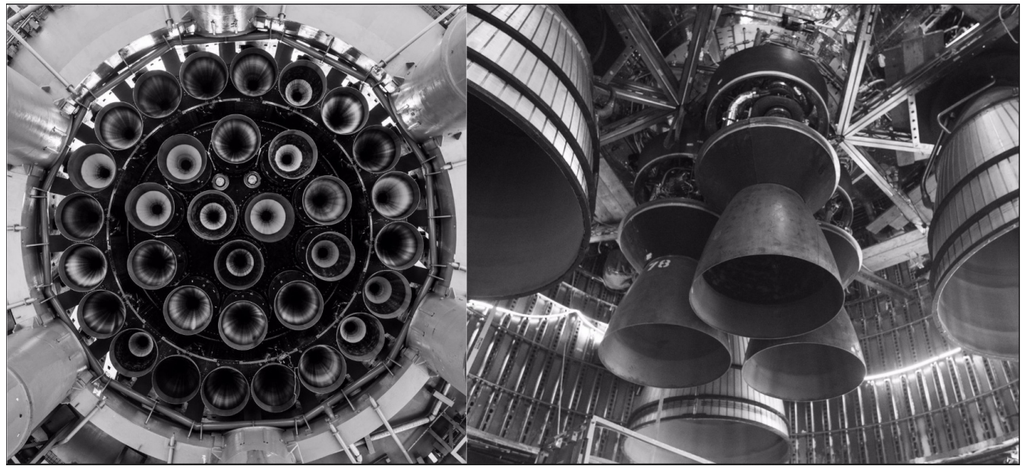
# Modelos computacionais



https://commons.wikimedia.org/wiki/
File:SpaceX_Starship_ignition_during_IFT-5.jpg



https://commons.wikimedia.org/wiki/
File:Starship_Booster_Return_on_Final_Approach_(54063904149).jpg



https://www.universetoday.com/articles/spacex-shares-an-image-of-the-super-heavy-booster-bristling-with-33-newly-installed-raptor-engines

**Seria muito difícil tentar entender como tudo funciona em um sistema complexo. Precisamos simplificar as coisas e, para isso, usamos um modelo computacional (computador simplificado). Quanto temos que simplificar até podermos entender? Modelo: transporte.**

# Modelos computacionais

https://www.infomoney.com.br/mercados/conheca-equipamento-que-o-maior-aviao-do-mundo-veio-buscar-no-brasil/

https://super.abril.com.br/tecnologia/antonov-225-destruido-na-ucrania-nasceu-para-carregar-onibus-espacial-sovietico/

## Não, ainda está muito complexo...

# Modelos computacionais


https://hpg.com.br/quanto-custa-comprar-um-aviao-teco-teco.html

## Melhorou um pouco, mas ainda está muito complexo.

# Modelos computacionais


https://commons.wikimedia.org/wiki/File:Hg-trike.jpg

**Agora simplificamos bastante, mas ainda sai do chão e tem motor.**
**Temos que simplificar mais nosso modelo.**

# Modelos computacionais

https://commons.wikimedia.org/wiki/File:Hanggliding03042006.JPG

**Ah!, conseguimos simplificar o motor. Mas ainda está saindo do chão. Vamos simplificar mais...**

# Modelos computacionais

https://commons.wikimedia.org/wiki/File:Triumph_Bicycle.JPG

**Já estamos quase chegando lá, mas ainda temos catracas, correntes, freios... precisamos simplificar mais...**

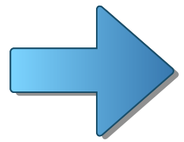
# Modelos computacionais


https://www.tramontina.com.br

## Agora sim!, temos um modelo simples o suficiente para tentarmos entender como o transporte ocorre. E na computação?

# Modelos computacionais

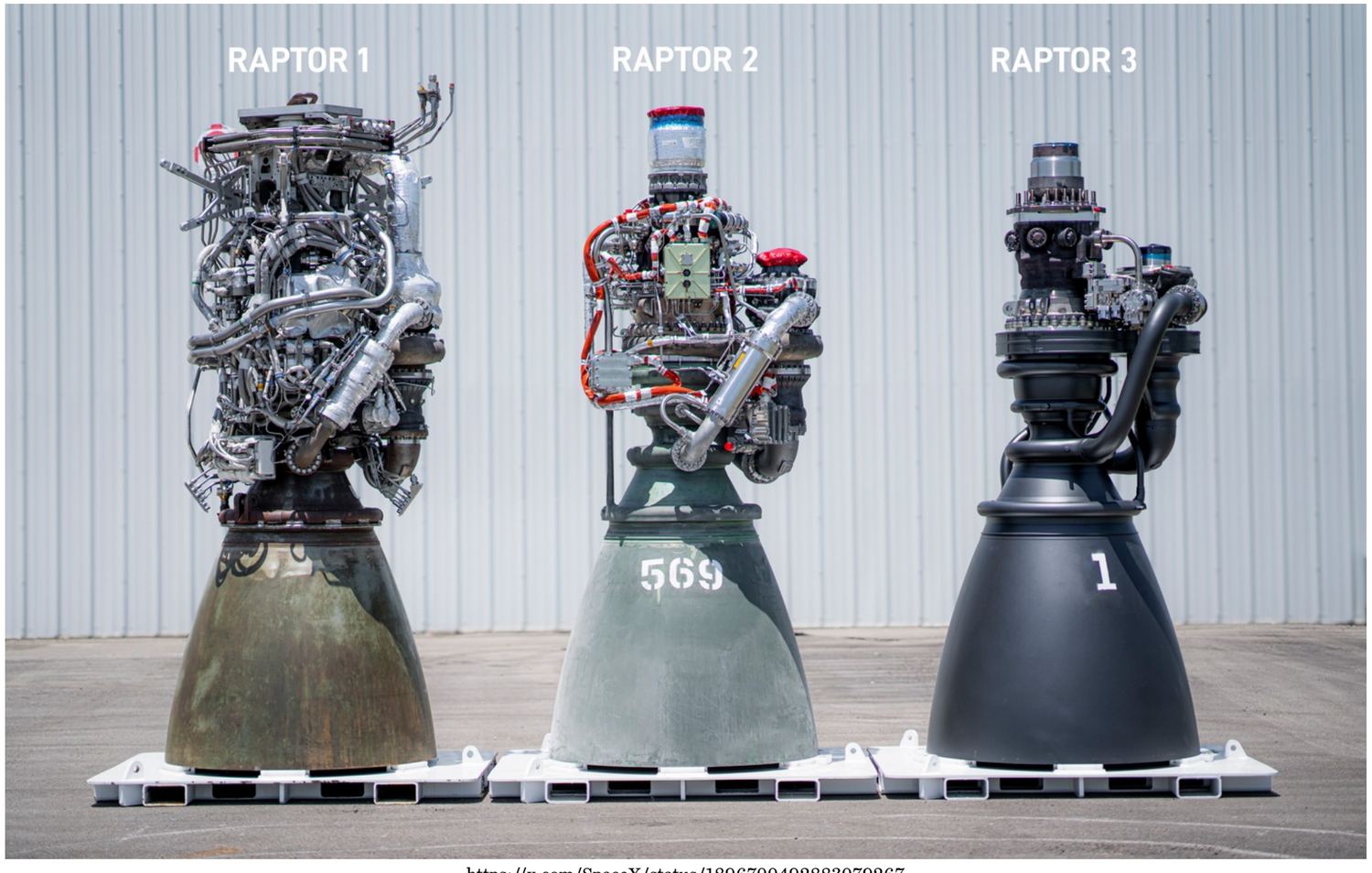
Podemos usar **diferentes modelos computacionais** dependendo das características que estamos interessados em estudar.




Imagens:
https://www.freepik.com
https://www.pichau.com.br
https://www.mercadolivre.com.br
https://www.educativosbrinquedos.com.br

# Modelos computacionais

**Essa simplificação toda não atrapalha o entendimento? NÃO!**
**Em geral, quanto mais entendemos uma coisa, mais simples ela se torna e vice-versa: entender profundamente as coisas simples nos permite entender versões complexas.**




https://x.com/SpaceX/status/1896700492883079267

# Autômatos finitos

**É o modelo computacional mais simples, também chamado de máquina de estados finitos.**

**Usado para modelar computadores com uma quantidade extremamente limitada de memória, quase nada. Esses dispositivos existem na realidade?**

https://www.assaabloyentrance.com

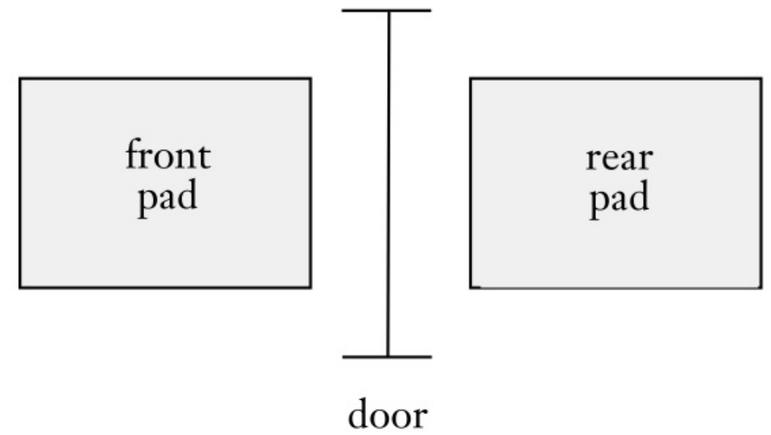
# Autômatos finitos

**Como uma porta automática de entrada realiza uma computação?**

**→ O computador (a porta) está em um de dois estados:**
- **aberto (OPEN)**
- **fechado (CLOSED)**

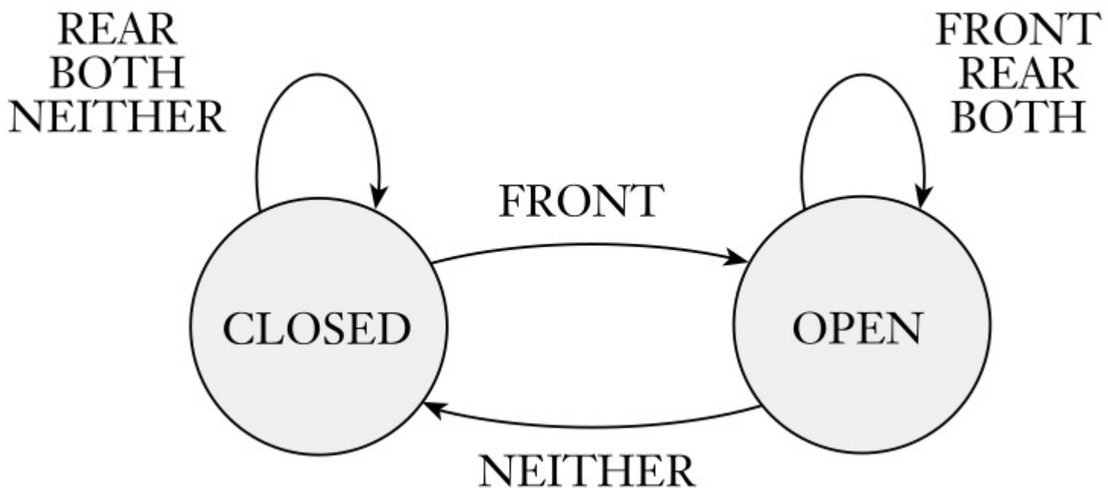
**→ Existem 4 inputs possíveis:**
- **frente (FRONT): uma pessoa está no piso frontal**
- **traseiro (REAR): uma pessoa está no piso traseiro**
- **ambos (BOTH): existem pessoas nos dois pisos**
- **nenhum (NEITHER): não há pessoas nos pisos**

**→ O computador (porta) faz transições entre os estados:**
- **de aberto para fechado**
- **de fechado para aberto**



# Autômatos finitos

## Como uma porta automática de entrada realiza uma computação?




Quanta memória esse computador precisa?

Que outros computadores podem ser modelados por autômatos finitos?

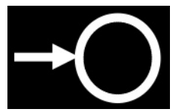
input signal

| state | NEITHER | FRONT | REAR | BOTH |
|---|---|---|---|---|
| CLOSED | CLOSED | OPEN | CLOSED | CLOSED |
| OPEN | CLOSED | OPEN | OPEN | OPEN |

# Autômatos finitos

**Definições matemáticas fundamentais:**

**→ Autômato** $M_1$

**→ Estado inicial** 

**→ Estados** $q_1\ q_2\ q_3$

**→ Transições** 


**→ Estado(s) aceito(s)** 

**→ Input: string finita**

**→ Output: aceitar ou rejeitar**

Obs.: o output pode ser bem mais complexo do que só SIM ou NÃO, mas vamos começar com esse modelo simples.

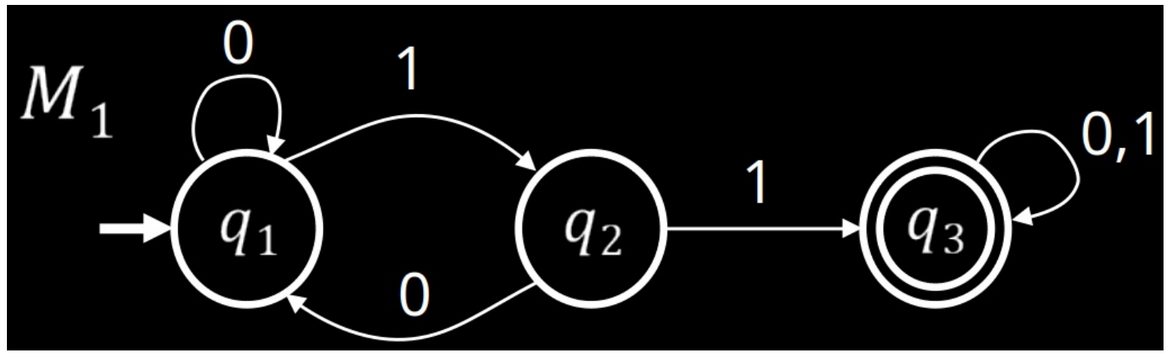



**Diagrama de Estados**

**Modo de computação: a partir do estado inicial, ler os símbolos do input, seguir as transições correspondentes, e aceitar a string se chegamos a um estado aceito, ou rejeitar a string se não chegamos a um estado aceito.**

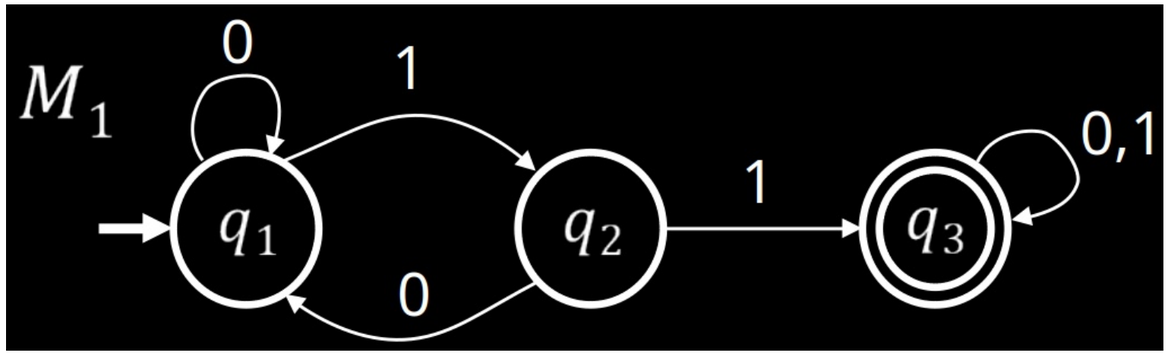
# Autômatos finitos

**Definições matemáticas fundamentais**

**→ Qual o resultado da computação com o autômato finito $M_1$ com as seguintes strings:**

**01101**

**00101**




**→ Quais são as strings que esse autômato aceita, que é capaz de processar até um estado aceito?**

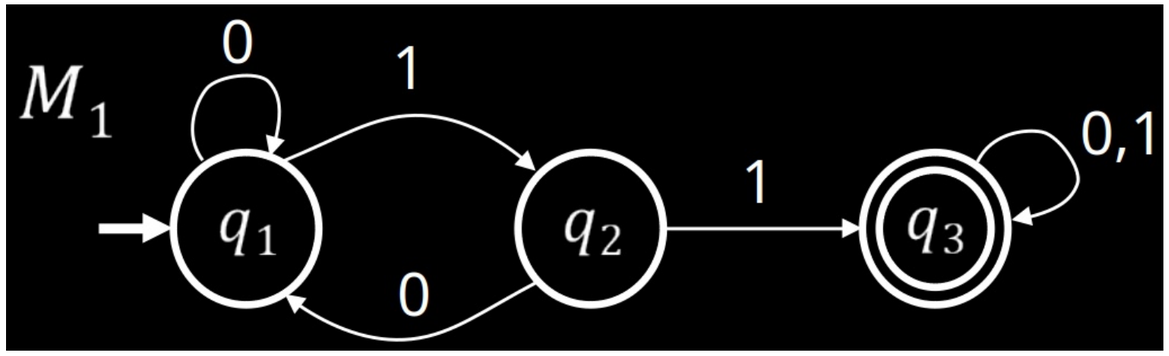
# Autômatos finitos

**Definições matemáticas fundamentais**

**→ Qual o resultado da computação com o autômato finito $M_1$ com as seguintes strings:**

**01101**

**00101**




**→ Quais são as strings que esse autômato aceita, que é capaz de processar até um estado aceito?**

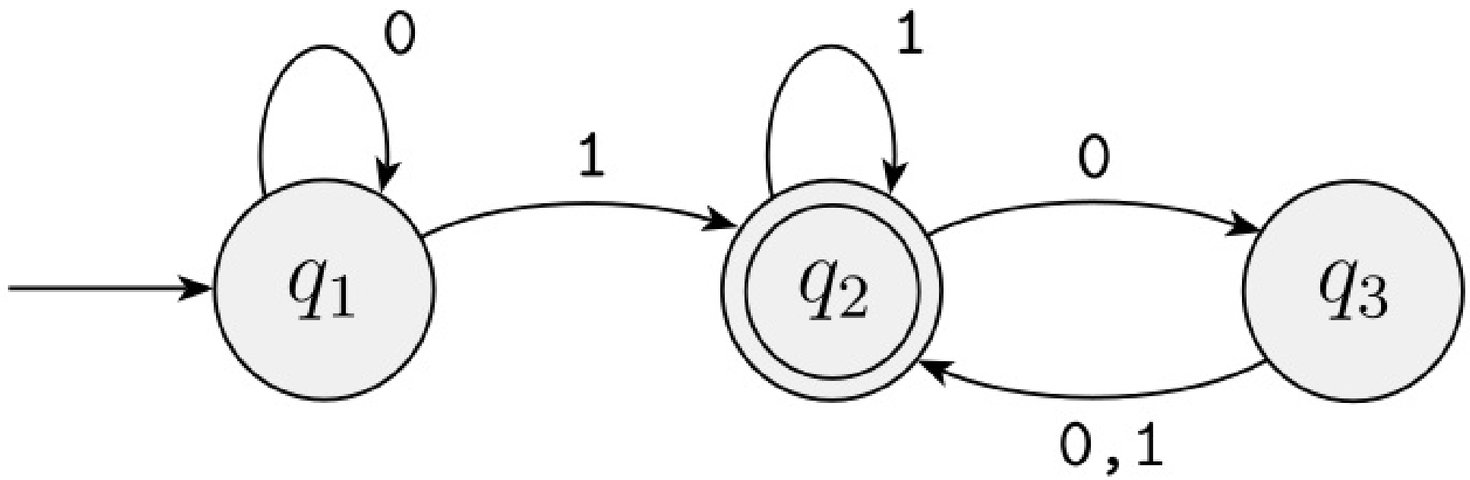
$$M_1 \text{ aceita o conjunto de strings } A \text{ onde}$$
$$A = \{w \mid w \text{ contenha a substring } 11\}$$

**→ O conjunto de strings aceitas é a linguagem desse autômato.**

# Autômatos finitos

**Que linguagem o autômato abaixo aceita?**
**Dica: teste com**

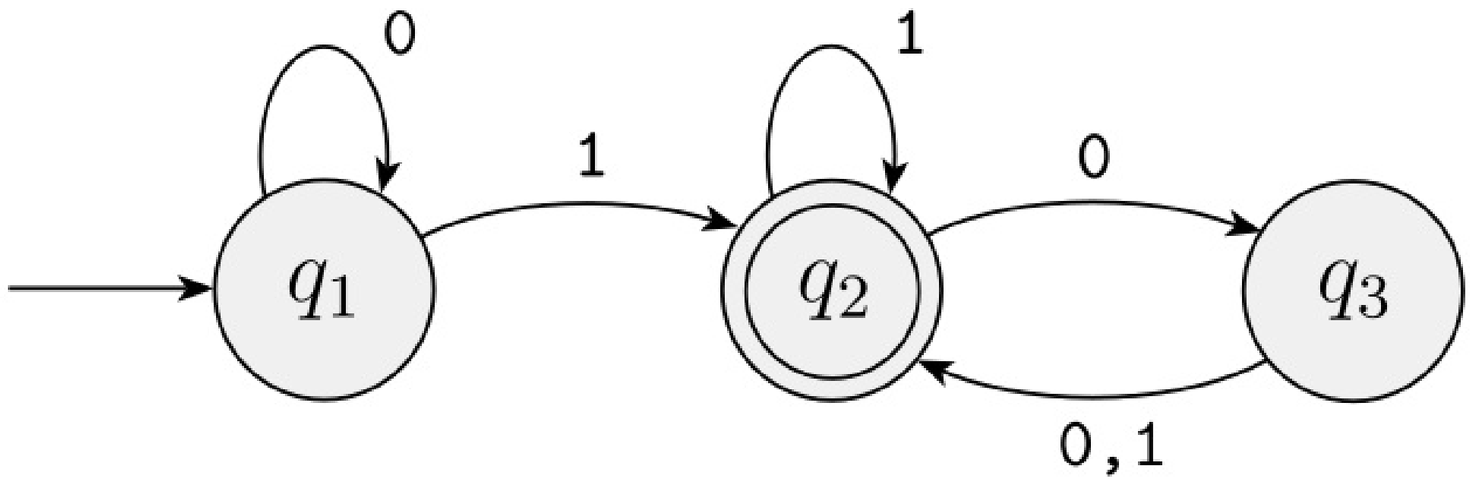
**0**
**1**
**01**
**10**
**11**
**100**
**1000**
**0100**
**0101**
**0001**
**11000**
**110000**
**0101000000**
**01010101011**



# Autômatos finitos

**Que linguagem o autômato abaixo aceita?**
**Dica: teste com**

**0**
**1**
**01**
**10**
**11**
**100**
**1000**
**0100**
**0101**
**0001**
**11000**
**110000**
**0101000000**
**01010101011**




**Aceita a linguagem formada por:**

→ **strings que terminam por 1; ou por**

→ **strings com número par de zeros após o último 1.**

# Autômatos finitos

**Definição matemática formal: um AF tem as seguintes "partes":**

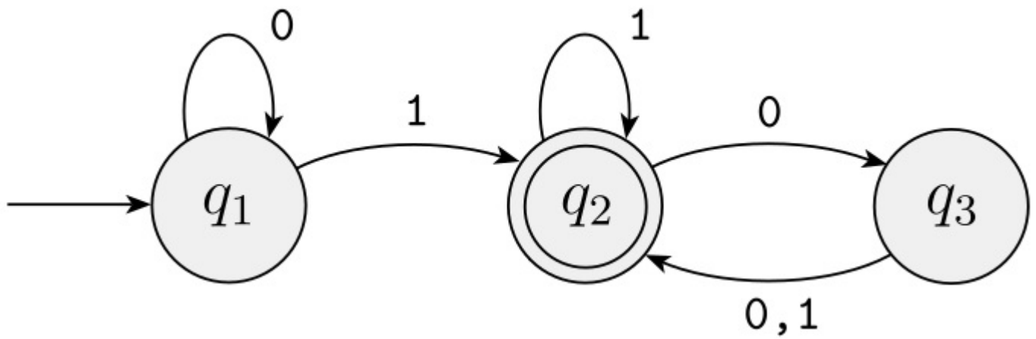
- **→ Estados**
- **→ Alfabeto de input**
- **→ Um ou mais estados finais (aceitos)**
- **→ Regras de transição entre os estados**
- **→ Estado inicial**

Um **autômato finito** é a 5-tupla $(Q, \Sigma, \delta, q_0, F)$ onde:

1. $Q$ é o conjunto finito chamado de **estados**
2. $\Sigma$ é o conjunto finito chamado **alfabeto**
3. $\delta : Q \times \Sigma \to Q$ é a **função de transição**
4. $q_0 \in Q$ é o **estado inicial**
5. $F \subseteq Q$ é o conjunto de **estados aceitos**

# Autômatos finitos

**Exemplo: defina formalmente o autômato $M_1$ abaixo:**




$M_1 = (Q, \Sigma, \delta, q_1, F)$ onde:

1. $Q = \{q_1, q_2, q_3\}$
2. $\Sigma = \{0, 1\}$
3. $\delta$ está descrita na tabela ao lado
4. $q_1$ é o estado inicial
5. $F = \{q_2\}$

| | 0 | 1 |
|---|---|---|
| $q_1$ | $q_1$ | $q_2$ |
| $q_2$ | $q_3$ | $q_2$ |
| $q_3$ | $q_2$ | $q_2$ |

Obs.: se $x$ e $y$ forem estados e $i$ for um símbolo, podemos descrever cada transição como $\delta(x, i) = y$.
Exemplo: $\delta(q_1, 0) = q_1$; $\delta(q_2, 0) = q_3$.

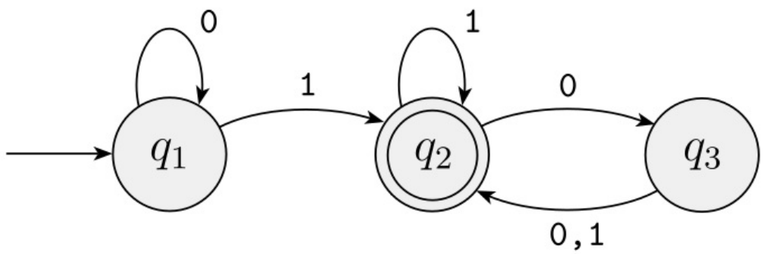
# Autômatos finitos

**A linguagem de um autômato finito:**

**Se A é o conjunto de todas as strings que o autômato $M_1$ aceita, nós dizemos que A é a linguagem do autômato $M_1$ e escrevemos:**

$$L(M_1) = A$$

**Também é comum dizer que $M_1$ reconhece A.**




$A = \{w \mid w \text{ contém pelo menos um 1 ou um número par de 0s após o último 1}\}$

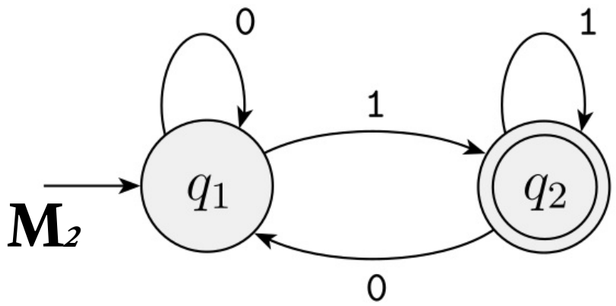
**Importante:**

**Um autômato pode aceitar diversas strings mas ele só reconhece uma única linguagem! Se um autômato não aceita strings, ele reconhece alguma linguagem? SIM, a linguagem vazia.**

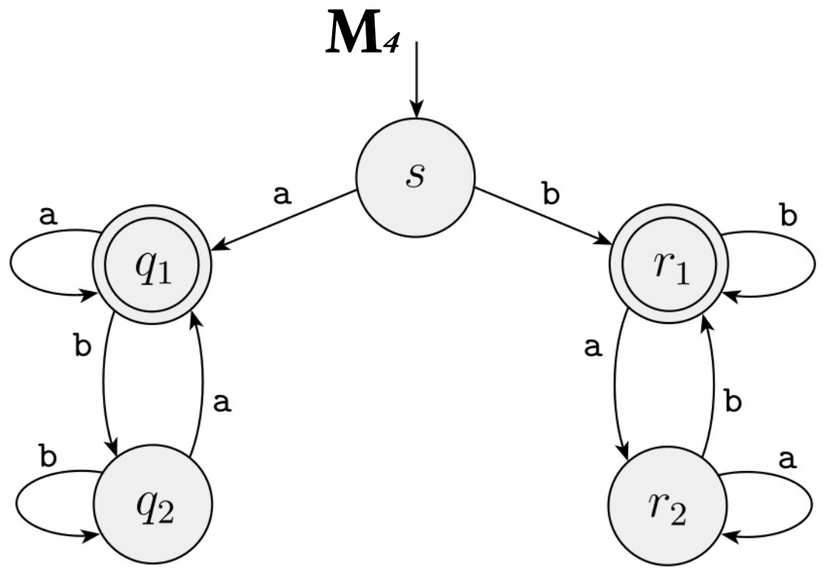
# Autômatos finitos

**Defina formalmente e descubra a linguagem dos autômatos $M_2$, $M_3$, $M_4$, $M_5$, abaixo.**

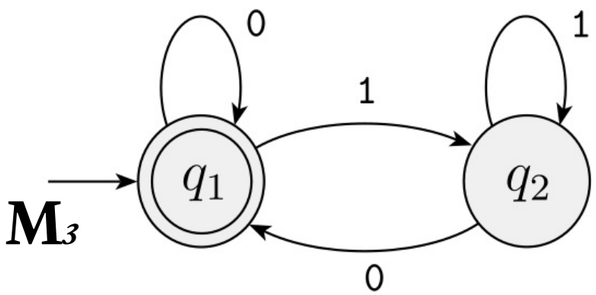
$M_2$



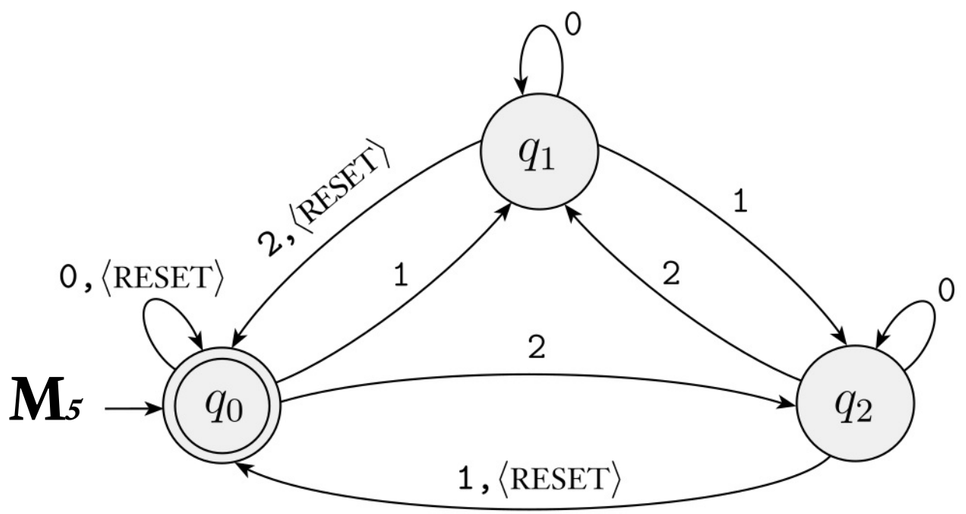

$M_4$



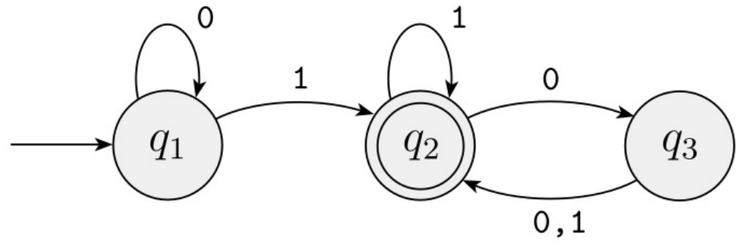

$M_3$




$M_5$



# Autômatos finitos




**Definição formal de computação:**

**Até agora a noção de computação que usamos foi a de avaliar "com os olhos" os estados, inputs e transições e ver se a string foi aceita. Para formalizar:**

**Seja:** $M = (Q, \Sigma, \delta, q_0, F)$ $\quad w = w_1 w_2 \cdots w_3$, onde cada $w_i \in \Sigma$

**Então** $M$ **aceita** $w$ **se houver uma seqüência de estados** $r_0, r_1, \ldots, r_n \in Q$ **com todas as três propriedades abaixo:**

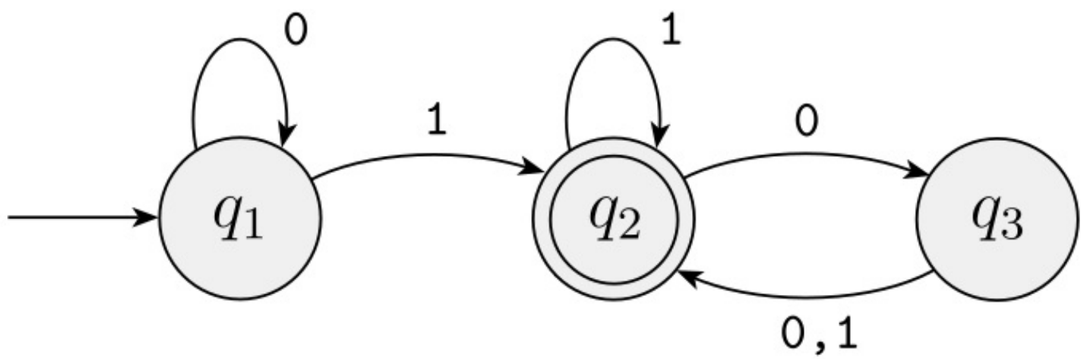
1. $r_0 = q_0$ ← a máquina começa no estado inicial
2. $\delta(r_i, w_{i+1}) = r_{i+1}$ para $i = 0, \ldots, n-1$ ← a máquina vai de estado para estado de acordo com a função de transição
3. $r_n \in F$ ← a máquina aceita o input se ela terminar em um estado aceito

$M$ **reconhece a linguagem** $A$ **se** $A = \{w \mid M \text{ aceita } w\}$.

# Autômatos finitos

**Importante:**

→ **Uma linguagem é dita linguagem regular se algum autômato finito a reconhecer.**




$A = \{w \mid w \text{ contém pelo menos um 1 ou um número par de 0s após o último 1}\}$

→ **Uma linguagem é dita não regular se nenhum autômato finito a reconhecer (a linguagem está além da capacidade dos autômatos finitos).**

# Criação de autômatos finitos

→ **Não é fácil**

→ **"Segredo" é se colocar no lugar do autômato e avaliar se, após cada símbolo recebido, a string recebida até esse momento está na linguagem. A razão para isso é que você, da mesma forma que o autômato, não vê o resto da string, só vê o que recebeu.**

→ **Descubra o que você precisa, de fato, se lembrar da string que recebeu até o momento, a medida que você a lê (lembre-se de que o autômato finito tem memória limitada, ele não pode se lembrar de toda a string), precisa apenas se lembrar do que é crucial.**

→ **Precisa de experiência.**

# Operações regulares

**Existem algumas operações que podemos realizar sobre as linguagens regulares que serão úteis para entender e criar autômatos. Essas operações são chamadas de operações regulares. Temos 3 operações regulares: união, concatenação e estrela.**

Sejam $A$ e $B$ linguagens. Nós definimos as operações regulares **união**, **concatenação** e **estrela** da seguinte forma:

- **União**: $A \cup B = \{x \mid (x \in A) \vee (x \in B)\}$
- **Concatenação**: $A \circ B = \{xy \mid (x \in A) \wedge (y \in B)\}$
- **Estrela**: $A^* = \{x_1 x_1 \dots x_k \mid k \geq 0 \text{ e cada } x_i \in A\}$

# Operações regulares

Seja o alfabeto $\Sigma = \{a, b, c, \ldots, y, z\}$.
Se $A = \{bom, mau\}$ e $B = \{menino, menina\}$, então:

- $A \cup B =$
- $A \circ B =$
- $A^* =$

# Operações regulares

Seja o alfabeto $\Sigma = \{a, b, c, \ldots, y, z\}$.
Se $A = \{bom, mau\}$ e $B = \{menino, menina\}$, então:

- $A \cup B = \{bom, mau, menino, menina\}$
- $A \circ B = \{bommenino, bommenina, maumenino, maumenina\}$
- $A^* = \{\varepsilon, bom, mau, bombom, bommau, maubom, maumau, bombombom, bombommau, \ldots\}$

# Operações regulares

**Teoremas importantes:**

**→ As linguagens regulares são fechadas sob a operação de união.**

**→ As linguagens regulares são fechadas sob a operação de concatenação.**

**→ As linguagens regulares são fechadas sob a operação de estrela.**

**Isso significa que aplicar as operações de união, concatenação e estrala, em linguagens regulares, sempre resulta em uma nova linguagem regular.**

Lembrete da matemática:
Uma coleção de objetos é fechada sob alguma operação se a aplicação da operação aos membros da coleção retorna um objeto ainda dessa mesma coleção.

# Dúvidas