# 1. Funções Computáveis

## 1.1 Algoritmos ou Procedimentos Efectivos

## 1.2 A máquina URM - Unlimited Register Machine

## 1.3 Funções computáveis por URM

## 1.4 Predicados decidíveis e problemas

## 1.5 Computabilidade noutros domínios

# 1.1 Algoritmos ou Procedimentos Efectivos

- Quando queremos multiplicar 12 x 12 podemos desenhar no chão um quadrado de 12 por 12 e contar os pequeno quadrados dentro dele. Ou então:

$$\begin{array}{r} 12 \\ 12 \\ \hline 24 \\ 12 \\ \hline 144 \end{array}$$

- E se pretendermos multiplicar 987654321 x 123456789 ?

  Em primeiro lugar, é virtualmente impossível construir o rectângulo apropriado. Mas mesmo que fosse construído, exércitos de pessoas demorariam séculos a contar os pequenos quadrados e, no fim, quem iria acreditar no resultado?

# 1.1 Algoritmos ou Procedimentos Efectivos (2)

- O processo simbólico de manipulação de dígitos de acordo com certas regras simples é baseado em propriedades da adição e da multiplicação que se verificam para todos os números.

- Estes métodos são designados por **algoritmos** ou **procedimentos efectivos**.

  Em geral, um algoritmo é uma regra mecânica ou um procedimento automático para executar alguma operação matemática.

- Quando um algoritmo é usado para calcular o valor de alguma funç numérica, diz-se que essa função é **computável**.

# 1.1 Algoritmos ou Procedimentos Efectivo (3)

**Exemplos**:

– Somar dois números

– Decidir se um número é primo

- Considere-se agora a função:

$$g(n) = \begin{cases} 1 & \text{, se existir uma sequência de exactamente } \boldsymbol{n} \text{ 7's seguidos na expansão decimal de } \pi \\ 0 & \text{, senão} \end{cases}$$

- Do ponto de vista matemático, esta função é aceitável. Mas será computável ?

# A demonstração de Euclides

- **"Existem infinitos número primos"**

  Esta demonstração mostra que qualquer que seja o número escolhido, existe sempre um número primo maior do que ele. Escolhemos um número: N. Calculemos N!

- N! é divisível por todos os números até N. Se adicionarmos 1 a N!,

  N! + 1 não é múltiplo de 2 ( se dividirmos por 2 o resto é 1)

  N! + 1 não é múltiplo de 3 ( se dividirmos por 3 o resto é 1)

  N! + 1 não é múltiplo de 4 ( se dividirmos por 4 o resto é 1)

  …

  N! + 1 não é múltiplo de N ( se dividirmos por N o resto é 1)

- Por outras palavras, se N!+1 for divisível por algum número diferente de 1 e de si próprio, só o poderá ser por um número superior a N. Então ou el número primo, ou os seus divisores primos são superiores a N.

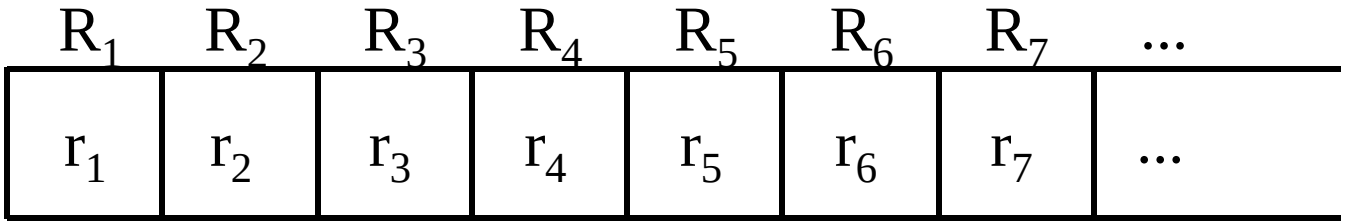
**De qualquer forma, tem que existir um número primo superior a N!**

# 1.2 A máquina URM

- **Idealização matemática de um computador**
- **Constituída por um número infinito de registos:**

  **$R_1$, $R_2$, $R_3$,…**

  **cada um dos quais contendo um número natural**



# 1.2 A máquina URM (2)

O conteúdo dos registos pode ser alterado através de **instruções**. Uma sequência finita de instruções designa-se por **programa.**

**Instruções:**

- Zero ***Z(n)***: para cada *n*: $0 \longrightarrow R_n$
  atribui o valor *0* ao conteúdo do registo $R_n$ ($r_n := 0$)
- Sucessor ***S(n)***: para cada *n*: $r_n+1 \longrightarrow R_n$
  atribui o valor $r_n+1$ ao conteúdo do registo $R_n$ ($r_n := r_n+1$)
- Tranferência ***T(m,n)***: para cada *m*, *n*: $r_m \longrightarrow R_n$
  atribui o valor $r_m$ ao conteúdo do registo $R_n$ ($r_n := r_m$)
- Salto ***J(m,n,q)***: para cada *m*,*n*,*q*: se $r_m = r_n$, executa a instrução $\mathbf{I_q}$

# Exemplo

$I_1$ **J(1,2,6)**

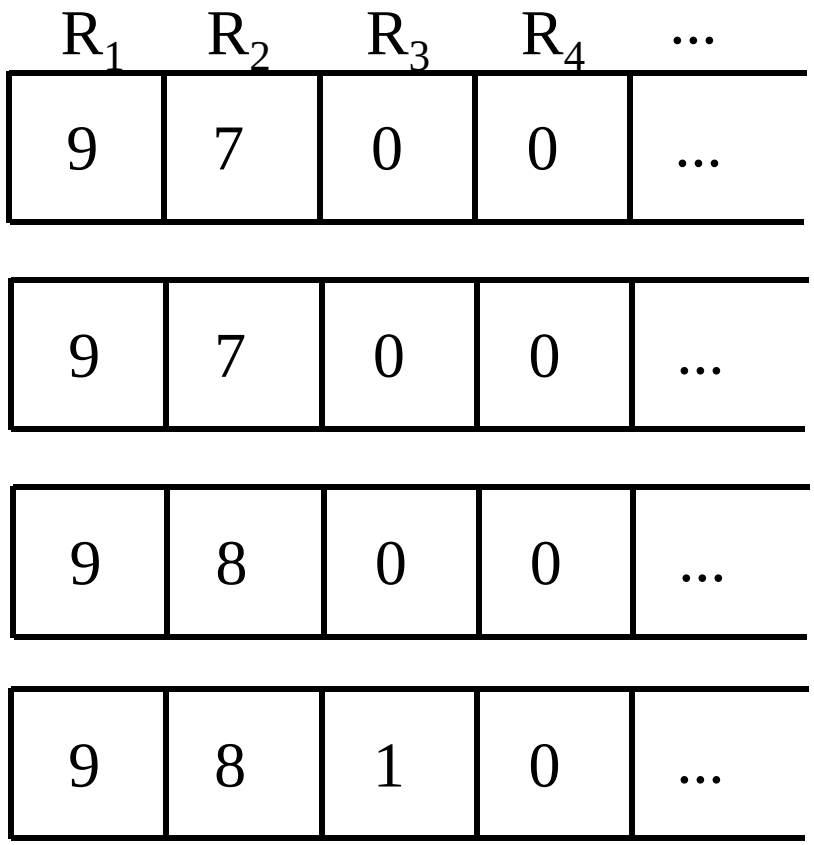
$I_2$ **S(2)**

$I_3$ **S(3)**

$I_4$ **J(1,2,6)**

$I_5$ **J(1,1,2)**

$I_6$ **T(3,1)**



# Programa URM

**P = $I_1$, $I_2$, …, $I_s$**. Seja $I_k$ uma instrução de **P**.

- Define-se a **instrução seguinte** da forma:

  Se $I_k$ não é uma instrução de salto a instrução seguinte é $I_{k+1}$

  Se $I_k$ **= J(m,n,q)**, a instrução seguinte é $\begin{cases} I_q & \text{, se } r_m = r_n \\ I_{k+1} & \text{, senao} \end{cases}$

- O programa **termina** quando não existir instrução seguinte, i.e:

  (i) se k = s e $I_s$ é uma instrução aritmética,

  (ii) se $I_k$ = J(m,n,q), $r_m = r_n$ e q > s,

  (iii) se $I_k$ = J(m,n,q), $r_m \neq r_n$ e k = s.

- A **configuração final** é a sequência $r_1, r_2, r_3, \ldots$, após a execução de P.

# Exercício

**Execute o programa anterior para a configuração inicial:**

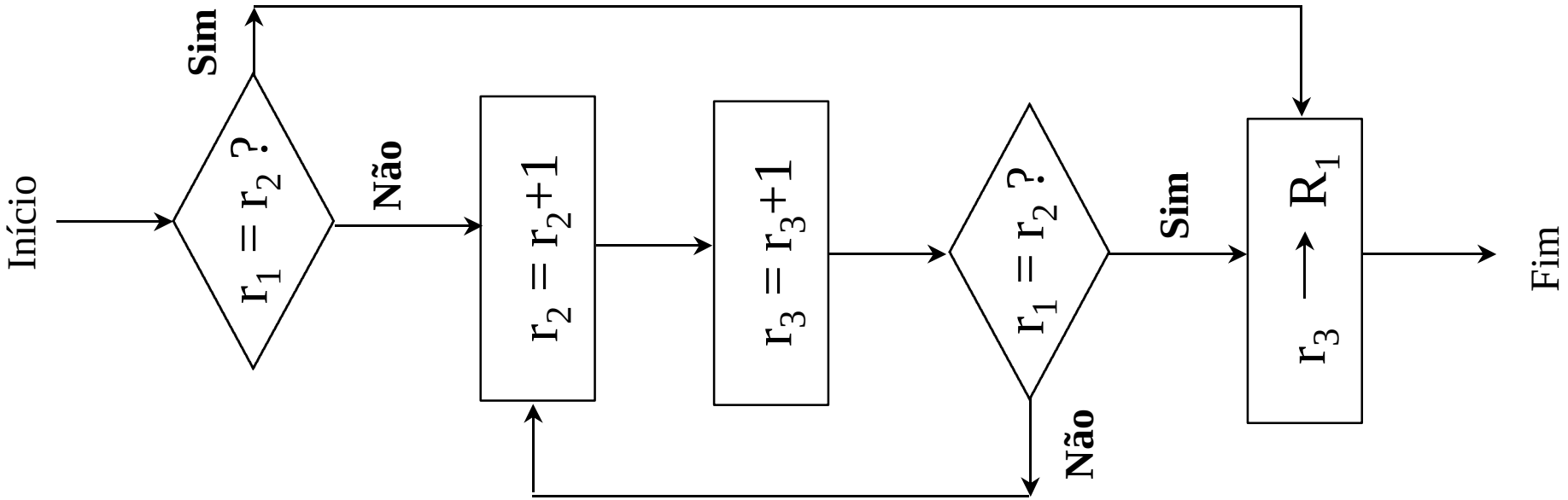
| $R_1$ | $R_2$ | $R_3$ | $R_4$ | ... |
|---|---|---|---|---|
| 8 | 4 | 2 | 0 | ... |

**E para a configuração inicial:**

| $R_1$ | $R_2$ | $R_3$ | $R_4$ | ... |
|---|---|---|---|---|
| 2 | 3 | 0 | 0 | ... |

**O que faz este programa?**

# Diagramas de Fluxo



# 1.3 Funções computáveis em URM

Seja $a_1$, $a_2$, $a_3$, … uma sequência infinita em |N e seja P um programa.

(i) $P(a_1,a_2,a_3,\ldots)$ é a execução de P para a configuração inicial $a_1,a_2,a_3,\ldots$

(ii) $P(a_1,a_2,a_3,\ldots)\downarrow$ significa que a computação $P(a_1,a_2,a_3,\ldots)$ pára (**converge**)

(iii) $P(a_1,a_2,a_3,\ldots)\uparrow$ significa que a computação $P(a_1,a_2,a_3,\ldots)$ nunca pára (**diverge**)

Considerando que todos excepto um número finito dos $a_i$ serão nulos, usaremos a seguinte notação: $P(a_1,a_2,\ldots,a_n)$.

Seja f uma função de $|N^n$ em |N.

Calcular o valor de f $(a_1,a_2,\ldots,a_n)$ pode ser interpretado como a execução do programa P com a configuração inicial $a_1,a_2,\ldots,a_n,0,0,\ldots$, ou seja, $P(a_1,a_2,\ldots,a_n)$.

O valor da função será o número $\mathbf{r_1}$ contido no registo $\mathbf{R_1}$.

# 1.3 Funções computáveis em URM

Seja *f* uma função parcial de $|N^n$ em $|N$.

**Definição**

Seja P um programa e $a_1,a_2,\ldots,a_n$, $b \in |N$.

(i) A computação $P(a_1,a_2,\ldots,a_n)$ **converge para *b*** se $P(a_1,a_2,\ldots,a_n)\downarrow$ e, na configuração final, *b* está em $R_1$.

(ii) P **URM-computa** *f* se, para qualquer $a_1,a_2,\ldots,a_n$, b , $P(a_1,a_2,\ldots,a_n)\downarrow$ *b* se e só se $(a_1,a_2,\ldots,a_n) \in$ Dom (f) e $f(a_1,a_2,\ldots,a_n) = b$

Em particular $P(a_1,a_2,\ldots,a_n)\downarrow$ se e só se $(a_1,a_2,\ldots,a_n) \in$ Dom (f) .

**Definição**

A função *f* é **computável por URM** se existe um programa P que computa *f*.

A classe das funções computáveis por designada por $\mathrm{C}$ .

# 1.3 Funções computáveis em URM

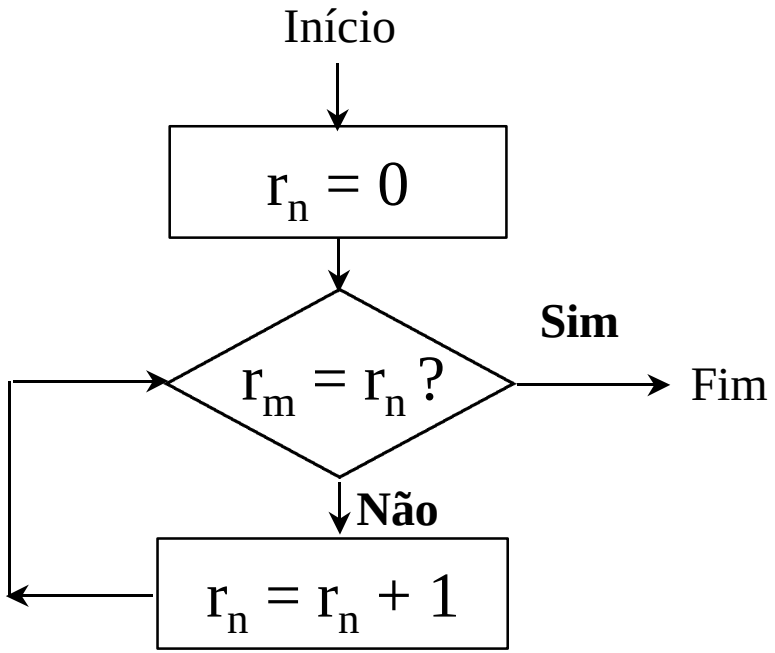
- Dado um qualquer programa P e uma qualquer configuração inicial $a_1,a_2,\ldots,a_n,0,0,\ldots$, existe uma única função n-ária que P computa, designada por $f_P^{(n)}$ :

$$f_P^{(n)}(a_1,a_2,\ldots,a_n) = \begin{cases} \text{o único b tal que } P(a_1,a_2,\ldots,a_n)\downarrow b & \textit{se } P(a_1,a_2,\ldots,a_n)\downarrow \\ \text{indefinida} & \textit{se } P(a_1,a_2,\ldots,a_n)\uparrow \end{cases}$$

# Exercício

- Mostre que, para cada instrução de transferência T(m,n), existe um programa sem instruções de transferência que tem exactamente o mesmo efeito que T(m,n).
- Ou seja, as instruções de transferência são redundantes, mas convenientes.

$I_1$ Z(n)
$I_2$ J(m,n,5)
$I_3$ S(n)
$I_4$ J(1,1,2)



# Exercícios

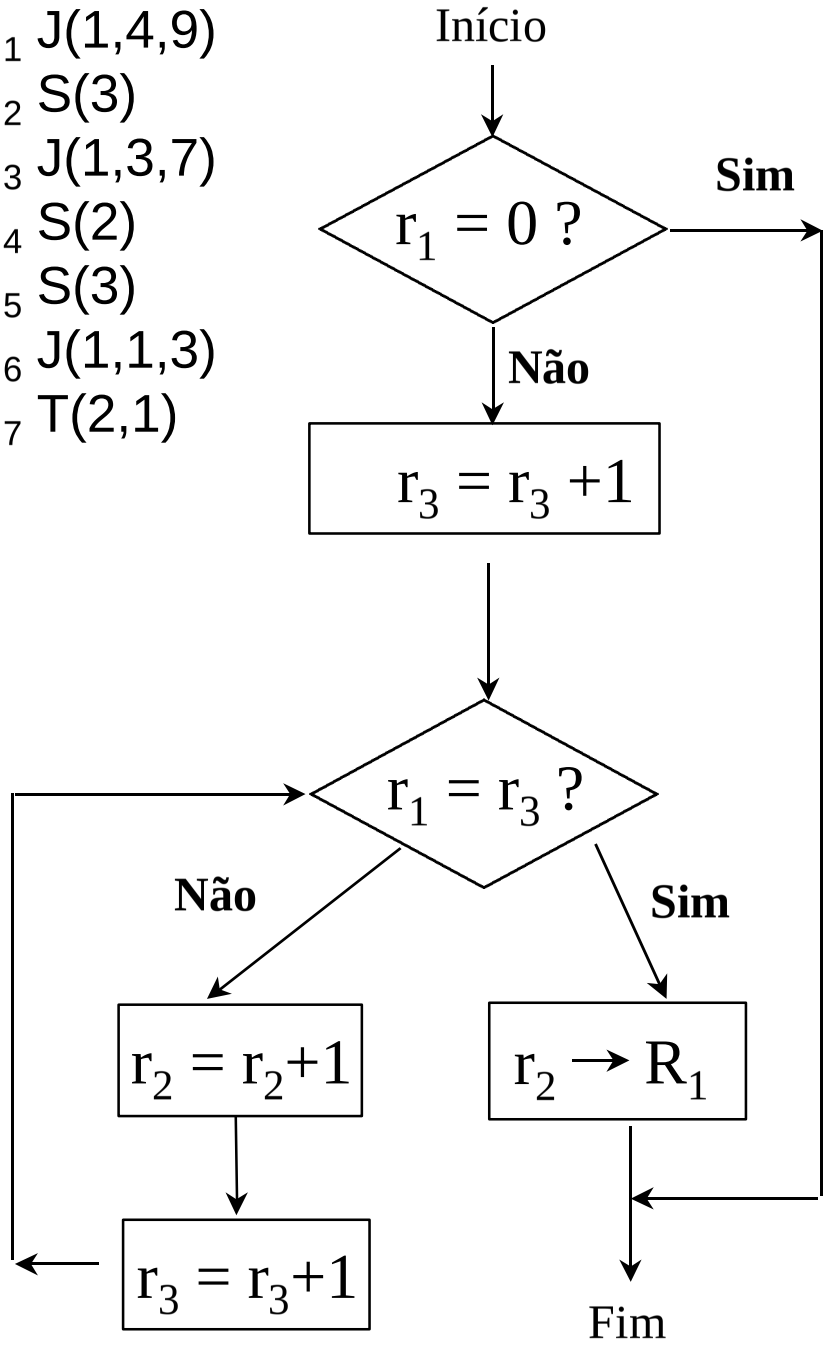
Mostre que as seguintes funções são computáveis, apresentando programas URM que as calculem:

$$x \dot{-} 1 = \begin{cases} x - 1 & \text{se x} > 0 \\ 0 & \text{se x} = 0 \end{cases}$$

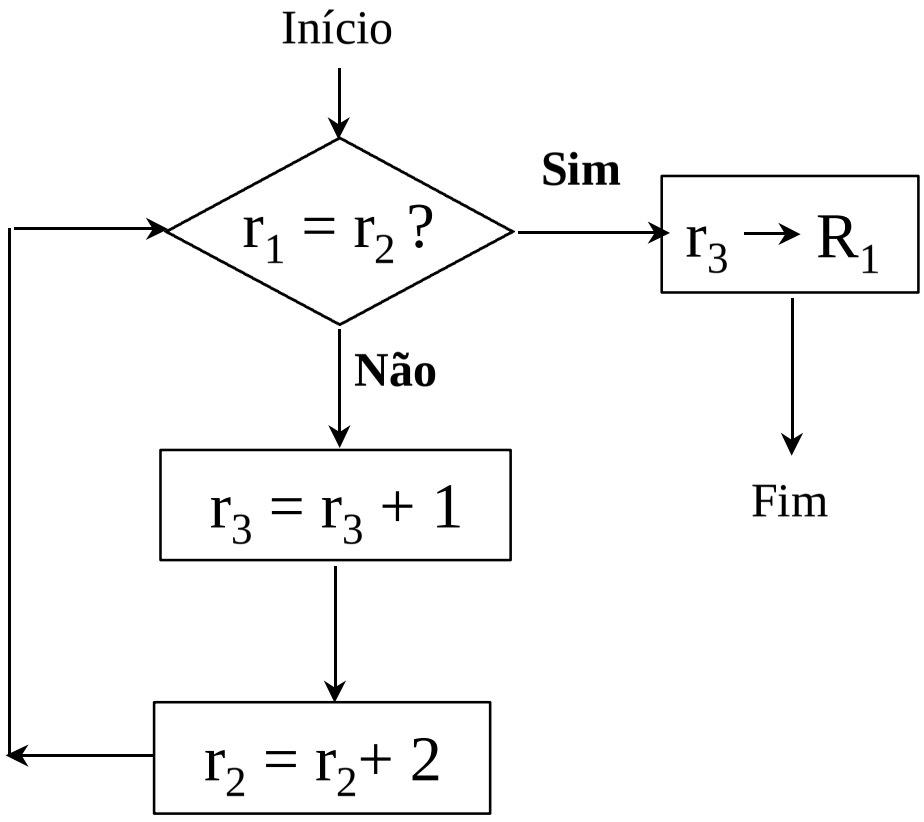
$$f(x) = \begin{cases} \frac{1}{2}x & \text{se x par} \\ \text{indefinida} & \text{se x impar} \end{cases}$$

$$x \dot{-} 1 = \begin{cases} x - 1 & \text{se } x > 0 \\ 0 & \text{se } x = 0 \end{cases}$$

$I_1$ J(1,4,9)
$I_2$ S(3)
$I_3$ J(1,3,7)
$I_4$ S(2)
$I_5$ S(3)
$I_6$ J(1,1,3)
$I_7$ T(2,1)



$$f(x) = \begin{cases} \frac{1}{2}x & \text{se x par} \\ \text{indefinida} & \text{se x impar} \end{cases}$$




$I_1$ J(1,4,6)
$I_2$ S(3)
$I_3$ S(2)
$I_4$ S(2)
$I_5$ J(1,1,1)
$I_6$ T(3,1)

# 1.4 Predicados e problemas decidíveis

- Dados dois números *x* e *y*, queremos saber se verificam a propriedade "*x* é múltiplo de *y*"

$$f(x,y) = \begin{cases} 1 & \text{se x é multiplo de y} \\ 0 & \text{se x não é múltiplo de y} \end{cases}$$

- O predicado (propriedade) "*x* é múltiplo de *y*" é efectivamente decidível ou decidível se a função f é computável.

# 1.4 Predicados e problemas decidíveis

Seja M (**x**) = M ($x_1$,$x_2$,…,$x_n$) um predicado n-ário de números naturais.

$$M(\mathbf{x}) : |\mathrm{N}^{n} \rightarrow |\mathrm{B} \qquad |\mathrm{B} = \{\tilde{\mathrm{n}}\mathrm{ao}, \mathrm{sim}\}$$

Defina-se **função característica**: $c_M(\mathbf{x}) : |\mathrm{N}^{n} \rightarrow \{0,1\}$

$$c_M(\mathbf{x}) = \begin{cases} 1 & \text{se M}(\mathbf{x}) \text{ se verifica} \\ 0 & \text{se M}(\mathbf{x}) \text{ não se verifica} \end{cases}$$

sim $\approx 1$

não $\approx 0$

**Definição**

O predicado M(x) é **decidível** se a função $c_M(\mathbf{x})$ é computável;

M(x) é **indecidível** se M(x) não é decidível.

# Exemplos

Os seguintes predicados são decidíveis:

$"x \neq y"$ $\quad c_M(x,y) = \begin{cases} 0 & \text{se x = y} \\ 1 & \text{se x} \neq \text{ y} \end{cases}$

$I_1$ Z(3)
$I_2$ J(1,2,4)
$I_3$ S(3)
$I_4$ T(3,1)

$"x = 0"$ $\quad c_M(x) = \begin{cases} 1 & \text{se x = 0} \\ 0 & \text{se x} \neq \text{ 0} \end{cases}$

$I_1$ J(1,2,3)
$I_2$ J(1,1,4)
$I_3$ S(2)
$I_4$ T(2,1)

# O gira-discos perfeito

*A tartaruga tinha um amigo, o caranguejo, que comprou um gira-discos convencido que tinha a capacidade de reproduzir qualquer som - era o gira-discos perfeito.*

*A tartaruga tentou-o convencer que isso não podia ser, mas não o conseguiu. Um dia levou-lhe um disco chamado "Não posso ser tocado pelo gira-discos I". Quando o caranguejo o tentou ouvir, após algumas notas o gira-discos começou a vibrar e, depois de um altíssimo "POP", desfez-se em milhares de pedaços.*

*O caranguejo, muito irritado, foi queixar-se ao vendedor do gira-discos que lhe forneceu outro mais potente e mais caro. Quando contou a novidade à tartaruga, esta escreveu ao fornecedor dos gira-discos e obteve toda a informação sobre a forma como eles eram construídos. Maldosamente, ela fabricou um novo disco chamado "Não posso ser tocado pelo gira-discos II". Mais uma vez, o gira-discos "perfeito" se desfez em mil pedaços.*

# O gira- discos perfeito (cont.)

*O caranguejo, desesperado, pensou em obter um gira-discos de baixa fidelidade, que não seria capaz de captar nem de reproduzir os sons que o iriam destruir, mas esse gira-discos estaria longe de ser perfeito.*

*Mas então, para qualquer gira-discos X a tartaruga fabricaria um disco chamado "Não posso ser tocado pelo gira-discos X".*

***Mas os "sons" encontram-se gravados no disco! Os sons existem (são verdadeiros), apesar de não poderem ser reproduzidos (provados) em nenhum gira-discos.***

# Um “isomorfismo”

| | | |
|---|---|---|
| **Gira-discos** | ⇔ | **Sistema axiomática de TN** |
| **Gira-discos de baixa fidelidade** | ⇔ | **Sistema axiomático “fraco”** |
| **Gira-discos de alta fidelidade** | ⇔ | **Sistema axiomático “forte”** |
| **Gira-discos perfeito** | ⇔ | **Sistema axiomático completo** |
| **Modelo do gira-discos** | ⇔ | **axiomas e regras do sistema** |
| **Disco** | ⇔ | **Expressão do sistema** |
| **Disco tocável** | ⇔ | **Teorema do sistema** |
| **Disco intocável** | ⇔ | **Não-teorema do sistema** |
| **Som** | ⇔ | **Afirmação verdadeira de TN** |
| **Som reproduzível** | ⇔ | **Teorema interpretado** |
| **Som irreproduzível** | ⇔ | **Afirmação verdadeira que não é um teorema** |

# 1.5 Computabilidade em outros Domínios

O modelo URM é definido apenas para números naturais.

As noções de computabilidade e decidibilidade podem ser estendidas a outros domínios através de uma **codificação**:

Uma codificação de um domínio D de objectos é uma função explícita e injectiva:

$a : \mathrm{D} \rightarrow |\mathrm{N}$

Diz-se que o objecto $d \in D$ está codificado pelo número natural $a(d)$.

Seja $f : \mathrm{D} \rightarrow \mathrm{D}$ .

$f$ é codificada pela função $f^* : |\mathrm{N} \rightarrow |\mathrm{N}$ $\qquad f^* = a \circ f \circ a^{-1}$

| $f$ é computável se $f^*$ é uma função computável de números naturais |
|---|

# Exemplo

Considere-se o domínio $\mathbb{Z}$. Uma codificação explícita é dada por:

$$a(n) = \begin{cases} 2n & \text{se } n \geq 0 \\ -2n-1 & \text{se } n < 0 \end{cases} \qquad á^{-1}(m) = \begin{cases} \frac{m}{2} & \text{se } m \text{ par} \\ -\frac{m+1}{2} & \text{se } m \text{ ímpar} \end{cases}$$

Seja $f(x) = x - 1$ em $\mathbb{Z}$ .

$f^* : |\mathrm{N} \rightarrow |\mathrm{N}$ é dada por

$$f^*(x) = \begin{cases} 1 & \text{se } x = 0 \\ x-2 & \text{se } x > 0 \text{ e } x \text{ par} \\ x+2 & \text{se } x \text{ ímpar} \end{cases}$$

$f^*$ é computável por URM. Então *x-1* é computável em $\mathbb{Z}$ .