

# HARDWARE E SOFTWARE

Prof. Muriel Mazzetto  
Algoritmos 1

# Algoritmo

2

- **Algoritmo** é uma sequência finita de instruções bem definidas e não ambíguas, cada uma das quais devendo ser executadas mecânica ou eletronicamente em um intervalo de tempo finito e com uma quantidade de esforço finita. (Wikipédia)

# Computador

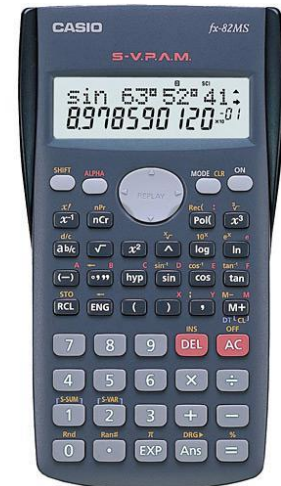
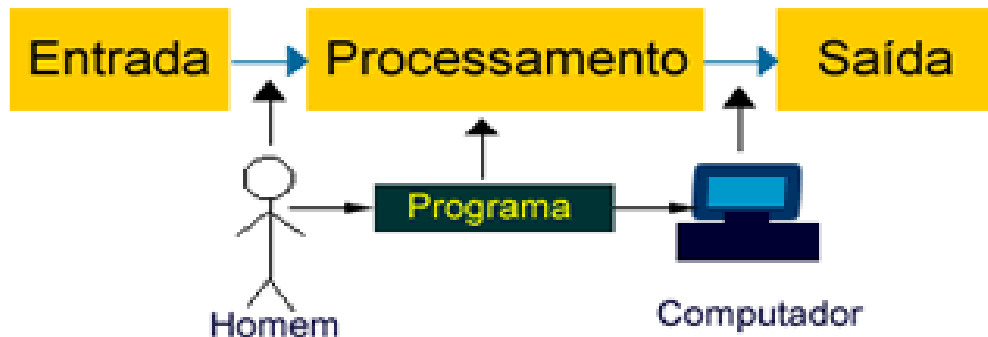
3

- Um computador recebe, manipula e armazena dados;
- É formado pelo **hardware** e pelo **software**:
  - ▣ Hardware: parte física, componentes eletrônicos;
  - ▣ Software: parte lógica, programas.
- Sua principal função é realizar o **processamento de dados**.

# Computador

4

- Processamento de dados: consiste em receber dados pelo dispositivo de entrada, realizar operações com estes dados e gerar um resultado que será exibido em um dispositivo de saída.



# Computador

5

- Entrada: periféricos (teclado, mouse, câmera), armazenamentos, etc.
- Processamento: sequência de operações lógicas e aritméticas (softwares).
- Saída: dados devolvidos aos periféricos (monitores, atuadores), comunicação de dados.

# Hardware

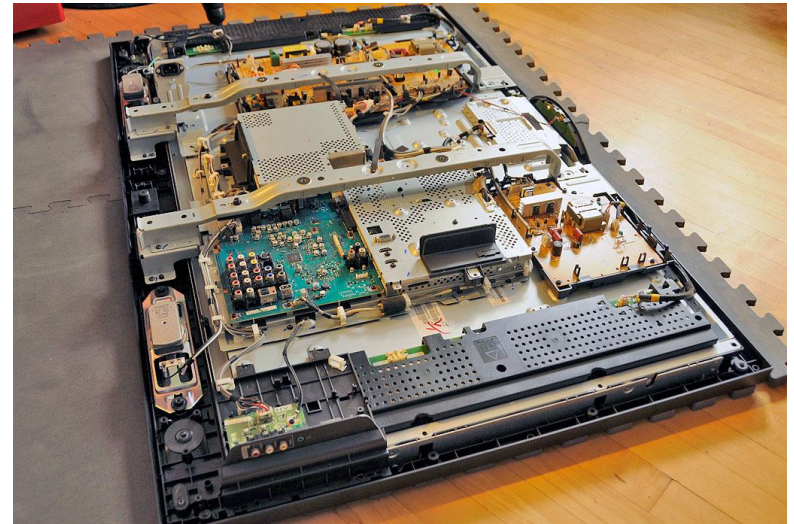
6

- Hardware em computação especifica unidades de processamento, memórias e dispositivos de entrada e saída.
- Não se refere apenas aos computadores pessoais mas também aos equipamentos embarcados.

# Hardware

7

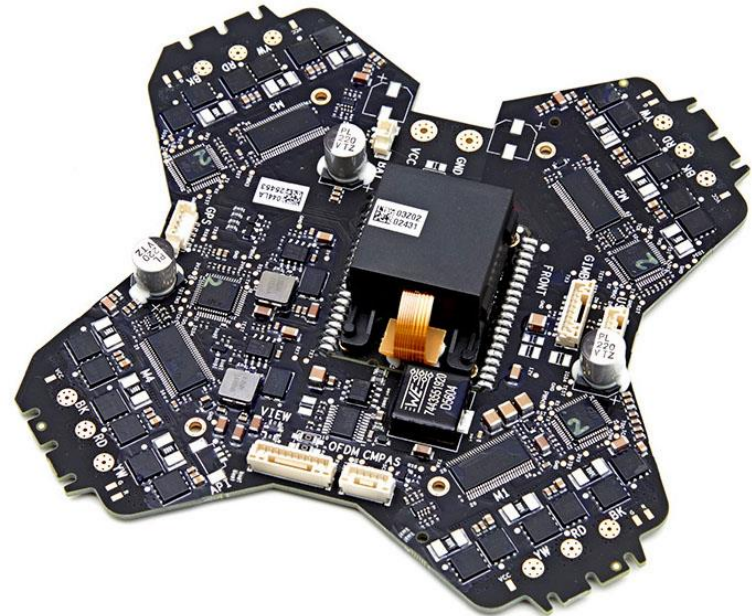
## □ Sistemas embarcados:



# Hardware

8

## □ Sistemas embarcados:

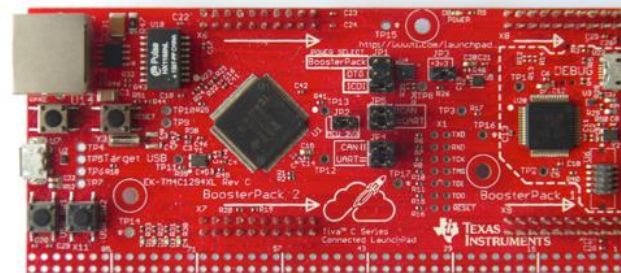
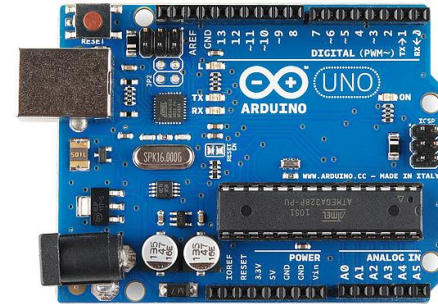




# Hardware

9

## □ Sistemas embarcados:



# Hardware

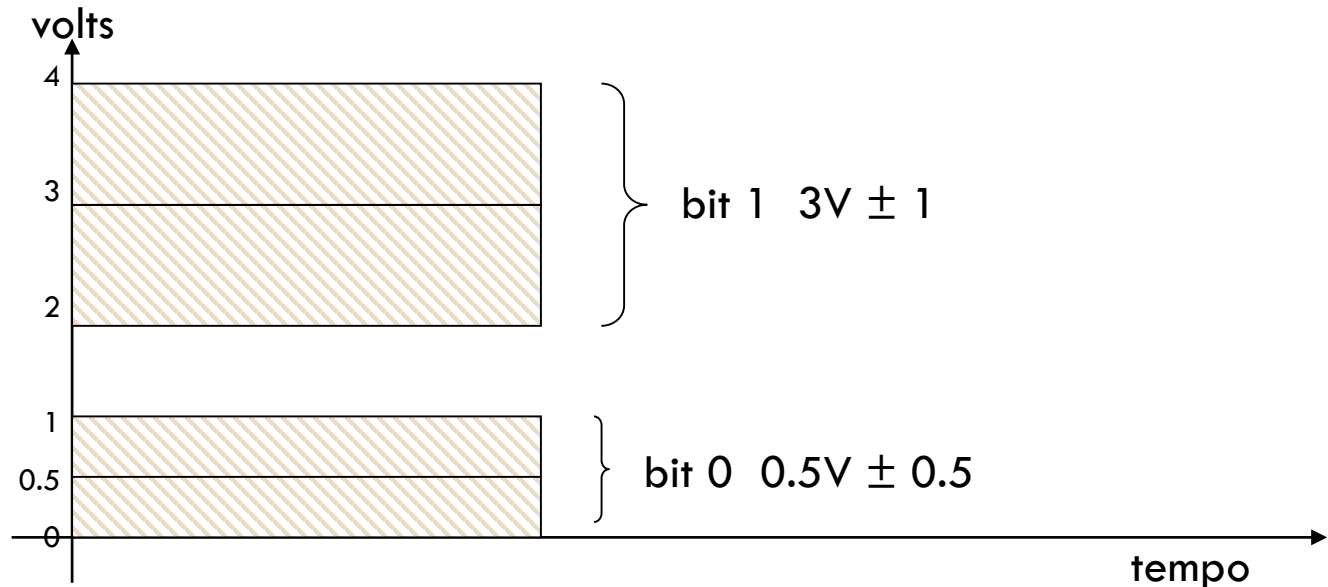
10

- Os computadores trabalham com o **sistema binário**, que utiliza dois dígitos (0 e 1).
- Cada dígito de um número na base binária é denominado **bit**.
- Bit é a menor unidade de representação de um dado, e pode assumir somente os valores 0 ou 1.

# Hardware

11

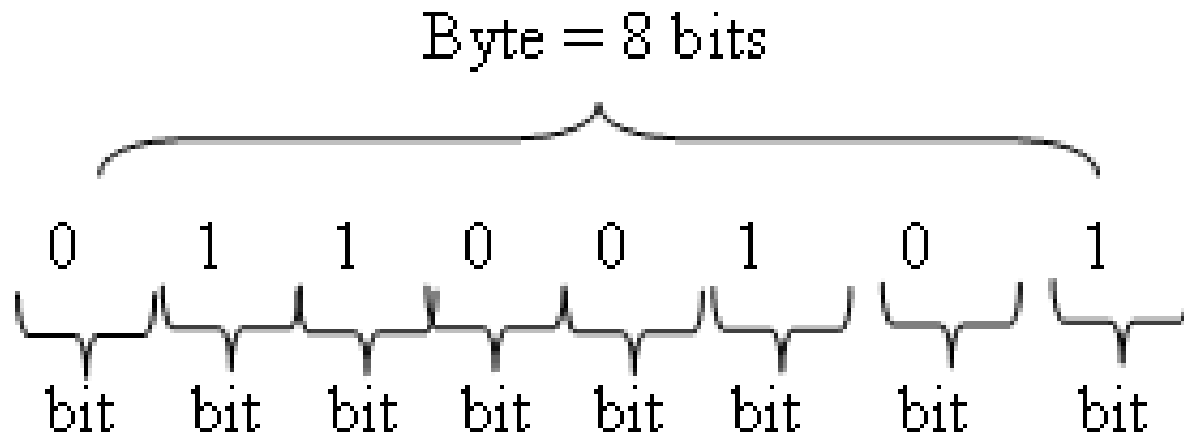
- A informação binária é representada fisicamente por sinais elétricos.
- De acordo com o nível de intensidade de tensão (volts) é determinado como 0 ou 1.



# Hardware

12

- Todos os **caracteres** derivam de uma representação binária.
- O padrão **ASCII** determina que o agrupamento de 8 bits gera um caractere.



# Hardware

13

- O Byte (Binary term) é um conjunto de 8 bits.
- Cada combinação de 8 bits forma um caractere diferente:
  - ▣ Dígitos numéricos (0 a 9);
  - ▣ Letras maiúsculas e minúsculas do alfabeto (A...Z, a...z);
  - ▣ Sinais de pontuação e símbolos (. , % \$);
  - ▣ Caracteres de controle (enter, backspace, espaço).
- São possíveis 256 combinações diferentes, que compõem a Tabela ASCII.

# Hardware

14

**Tabela ASCII -I**

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	\$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□

**Tabela ASCII -II**

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	€	160	A0	À	192	C0		224	E0	
129	81		161	A1		193	C1		225	E1	
130	82		162	A2		194	C2		226	E2	
131	83		163	A3		195	C3		227	E3	
132	84		164	A4		196	C4		228	E4	
133	85		165	A5		197	C5		229	E5	
134	86		166	A6		198	C6		230	E6	
135	87		167	A7		199	C7		231	E7	
136	88		168	A8		200	C8		232	E8	
137	89		169	A9		201	C9		233	E9	
138	8A		170	AA		202	CA		234	EA	
139	8B		171	AB		203	CB		235	EB	
140	8C		172	AC		204	CC		236	EC	
141	8D		173	AD		205	CD		237	ED	
142	8E		174	AE		206	CE		238	EE	
143	8F		175	AF		207	CF		239	EF	
144	90		176	B0		208	D0		240	F0	
145	91		177	B1		209	D1		241	F1	
146	92		178	B2		210	D2		242	F2	
147	93		179	B3		211	D3		243	F3	
148	94		180	B4		212	D4		244	F4	
149	95		181	B5		213	D5		245	F5	
150	96		182	B6		214	D6		246	F6	
151	97		183	B7		215	D7		247	F7	
152	98		184	B8		216	D8		248	F8	
153	99		185	B9		217	D9		249	F9	
154	9A		186	BA		218	DA		250	FA	
155	9B		187	BB		219	DB		251	FB	
156	9C		188	BC		220	DC		252	FC	
157	9D		189	BD		221	DD		253	FD	
158	9E		190	BE		222	DE		254	FE	
159	9F		191	BF		223	DF		255	FF	

# Hardware

15

- Partes relevantes de um computador para o entendimento de programação:
  - ▣ Disco rígido.
  - ▣ Memória RAM.
  - ▣ Processador.
  - ▣ Memória cache.



# Hardware

16

- Partes relevantes de um computador para o entendimento de programação:
  - ▣ Disco rígido: onde são instalados os programas e armazenados dados permanentemente.





# Hardware

17

- Partes relevantes de um computador para o entendimento de programação:
  - ▣ Memória RAM: onde os programas são armazenados em tempo de execução.



# Hardware

18

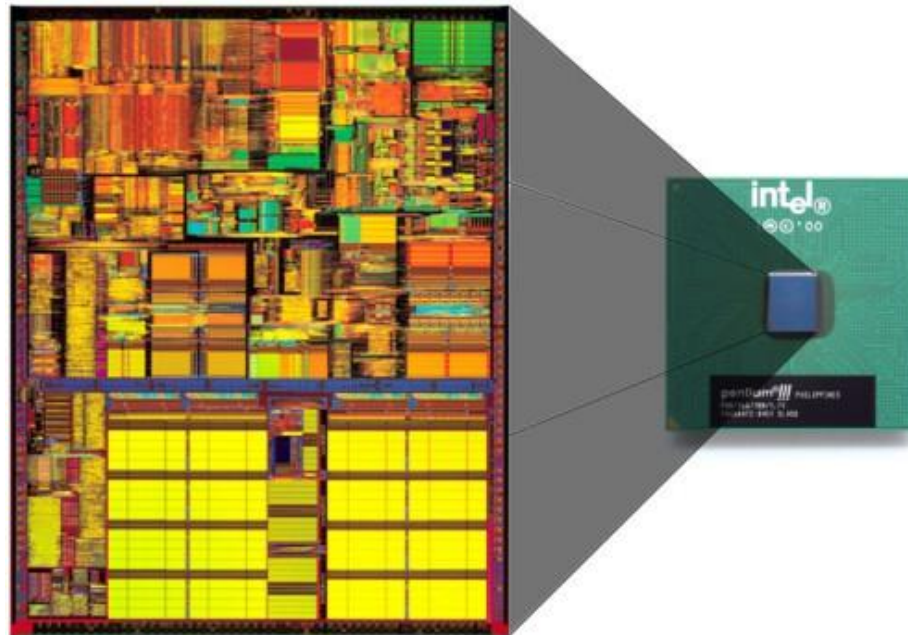
- Partes relevantes de um computador para o entendimento de programação:
  - ▣ Processador: responsável pela manipulação dos dados.



# Hardware

19

- Partes relevantes de um computador para o entendimento de programação:
  - ▣ Memória cache: memória localizada dentro do processador, para aumentar velocidade no acesso aos dados.



# Hardware

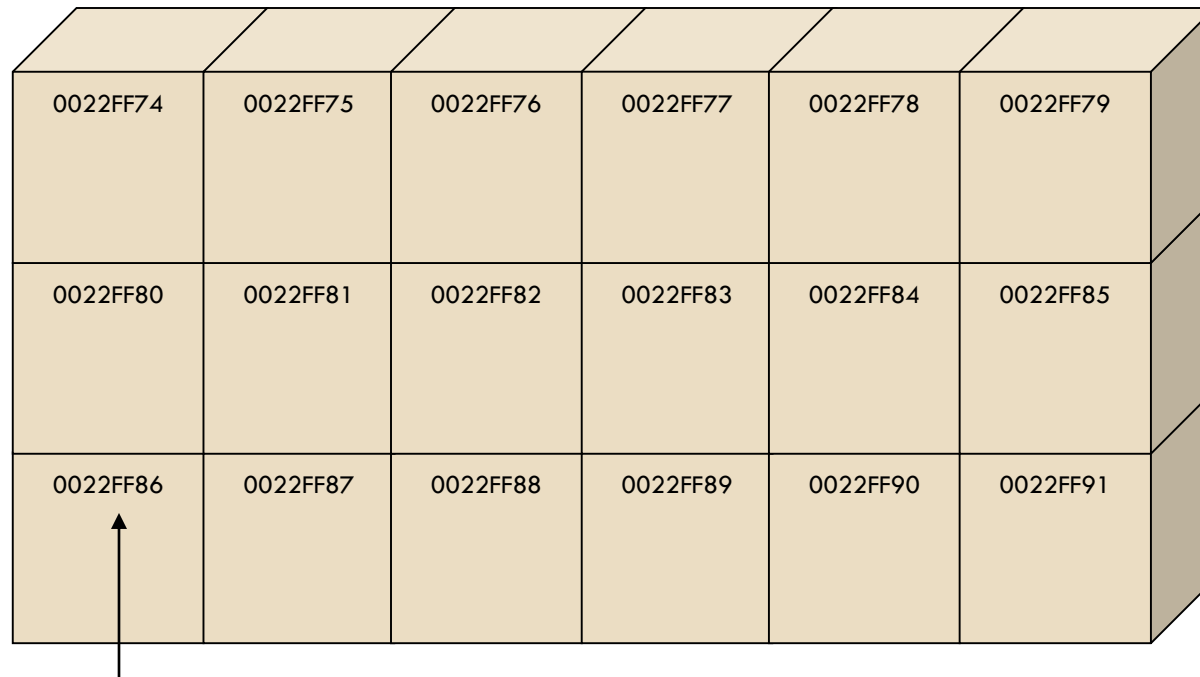
20

- ❑ Memória principal (RAM): um tipo de memória de leitura e escrita.
- ❑ Constituída por células de armazenamento para dados manipulados pelo software.
- ❑ Cada célula possui um endereço.

# Hardware

21

- Memória: armazena **bytes** em tempo de execução.
- Esses bytes compõem os programas e os dados.



Célula de memória (endereço sequencial) – local de armazenamento

# Hardware

22

- Representação esquemática das células de memória de um computador.

<i>Endereço</i>	<i>Conteúdo</i>
0022FF74	1
0022FF76	2
0022FF78	3
0022FF80	4
0022FF82	5

} = célula  
= um *byte*  
= um caractere (letra,  
dígito, símbolo)

→ localização da célula de memória

# Hardware

23

## □ Exemplo do armazenamento da palavra 'aluno':

Endereço	Conteúdo (representação binária)	código ASCII	Caractere
0040280E	0110 0001	97	a
0040180F	0110 1100	108	l
00401810	0111 0101	117	u
00401811	0110 1110	110	n
00401812	0110 1111	111	o

Endereço de memória

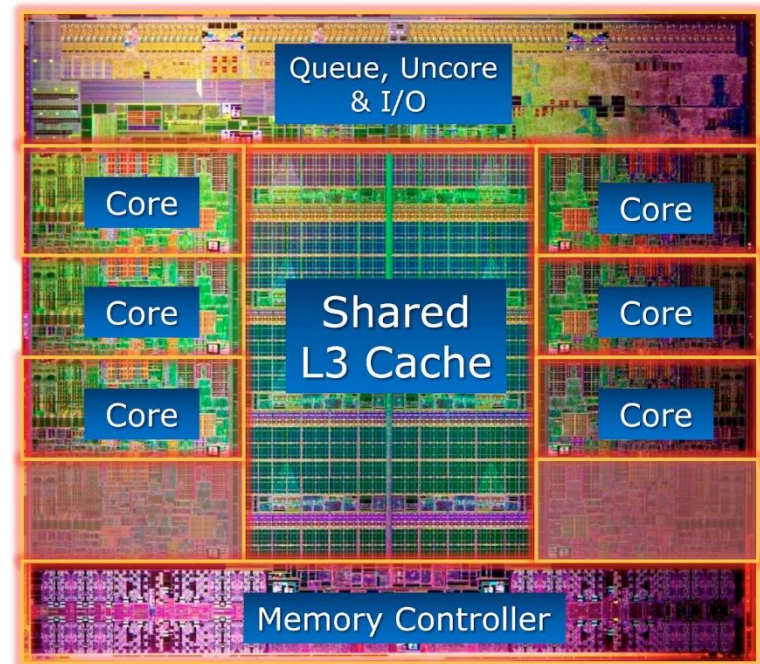
célula de memória, armazenamento de um byte (um caractere)



# Hardware

24

- Unidade Central de Processamento (CPU).
- Realiza as instruções de um programa de computador (aritmética, lógica e entrada/saída).





# Hardware

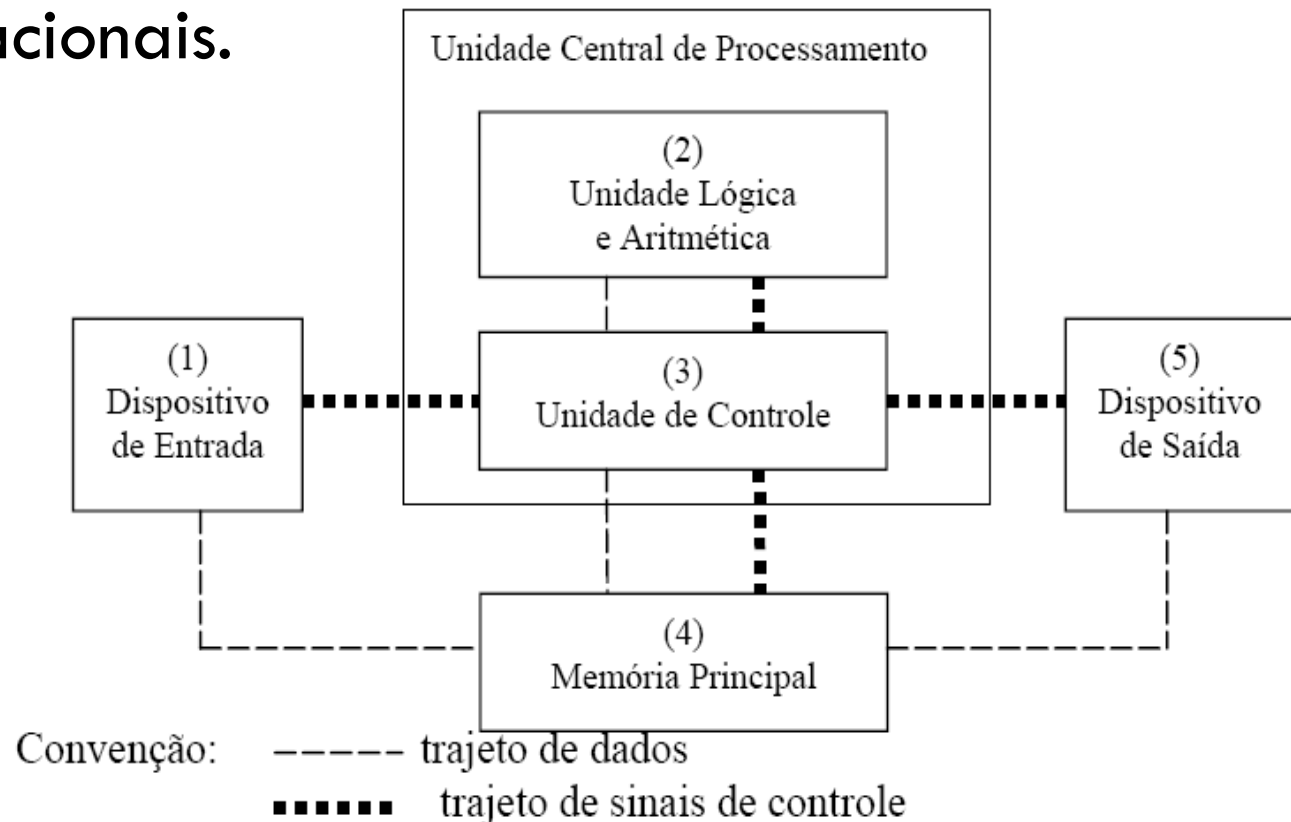
25

- Existem outros componentes que fazem parte de computadores e sistemas embarcados, para torná-los mais robustos e de aplicações específicas.
  - ▣ Placas gráficas;
  - ▣ Memórias secundárias;
  - ▣ Sensores de posicionamento.

# Hardware

26

- A Arquitetura de Computadores é a área responsável pelo estudo do projeto de sistemas computacionais.



# Software

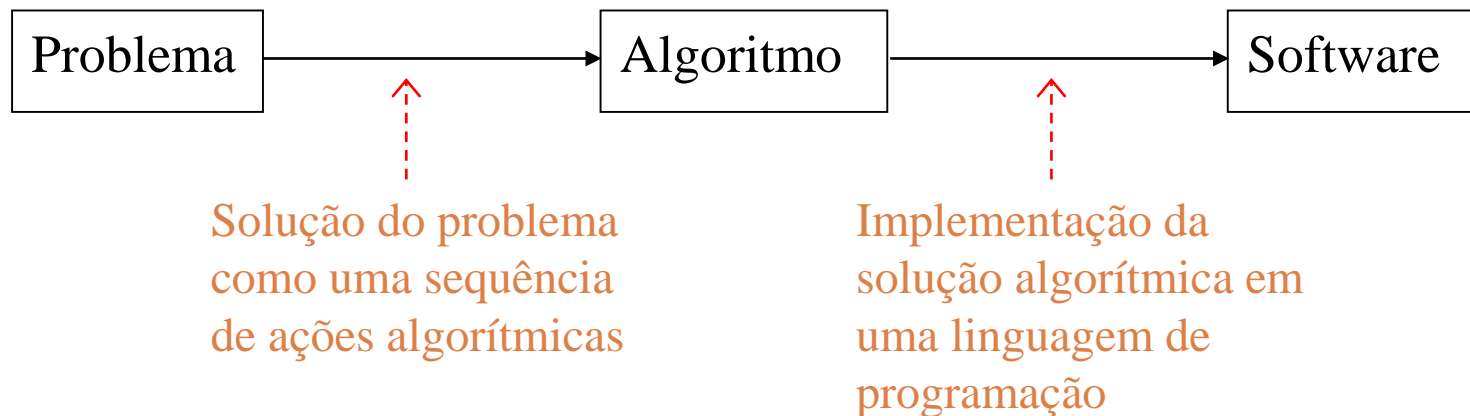
27

- Instruções organizadas em uma sequência lógica predefinida, executadas em um dispositivo eletrônico.
- Denominado também de programa, aplicativo e sistema operacional.

# Software

28

- A criação de um software passa pelas seguintes fases:
  - ▣ Análise.
  - ▣ Algoritmo.
  - ▣ Codificação.



# Software

29

- A criação de um software passa pelas seguintes fases:
  - ▣ Análise:
    - Estudo do enunciado do problema para definir os dados de entrada, o processamento e a saída.



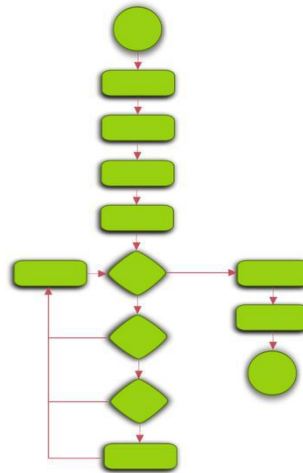
# Software

30

- A criação de um software passa pelas seguintes fases:
  - ▣ Algoritmo:
    - Utilização de ferramentas para descrever os passos para o processamento de dados.



Descrição narrativa



Fluxograma

```
Programa Aprovação;  
var Nota1, Nota2, Nota3, Média: real;  
início  
  leia(Nota1, Nota2, Nota3);  
  Média ← (Nota1 + Nota2 + Nota3) / 3;  
  Se Média >= 7  
    Então Escreva("Aprovado")  
  Senão Escreva("Reprovado");  
fim.
```

Pseudocódigo

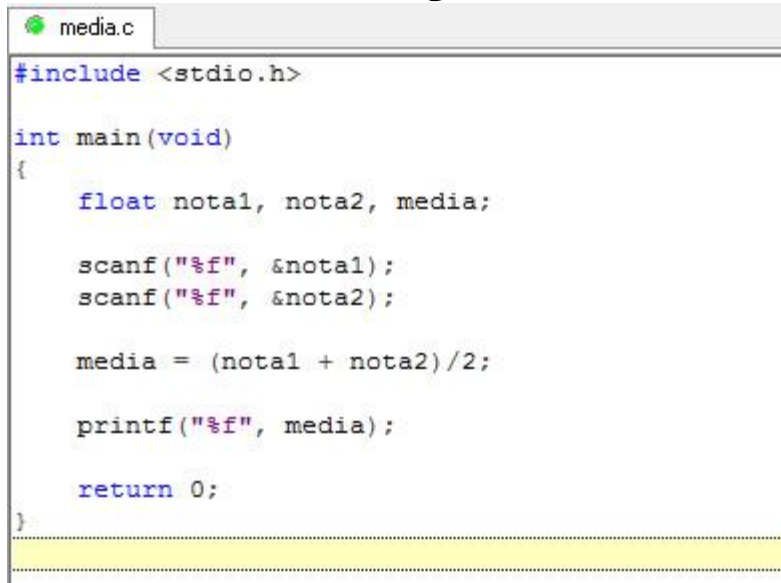
# Software

31

□ A criação de um software passa pelas seguintes fases:

▣ Codificação:

■ Transcrição do algoritmo em uma **linguagem de programação**.

A screenshot of a code editor window titled 'media.c'. The code is written in C and calculates the average of two floating-point numbers. It includes the standard input/output header, defines a main function, declares variables for the two numbers and their average, reads input from the user, calculates the average, and prints the result.

```
#include <stdio.h>

int main(void)
{
    float nota1, nota2, media;

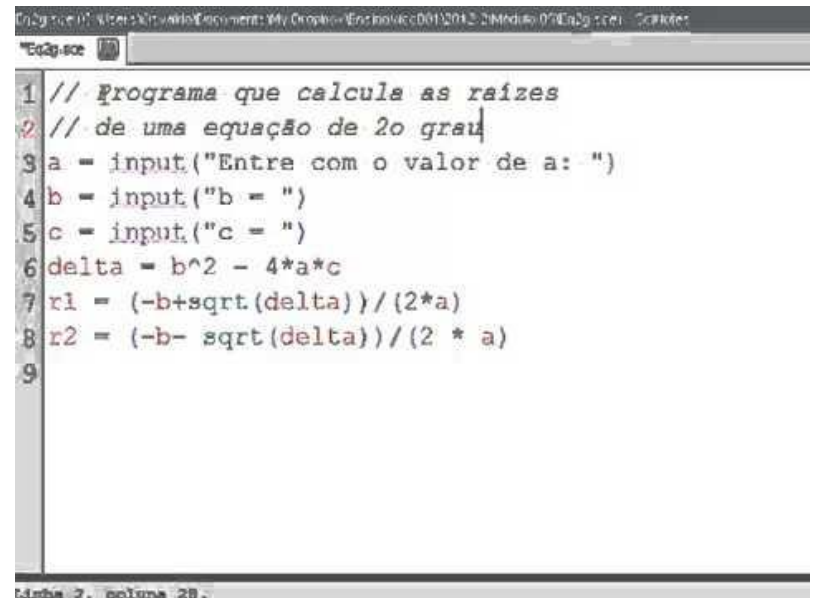
    scanf("%f", &nota1);
    scanf("%f", &nota2);

    media = (nota1 + nota2)/2;

    printf("%f", media);

    return 0;
}
```

Linguagem C

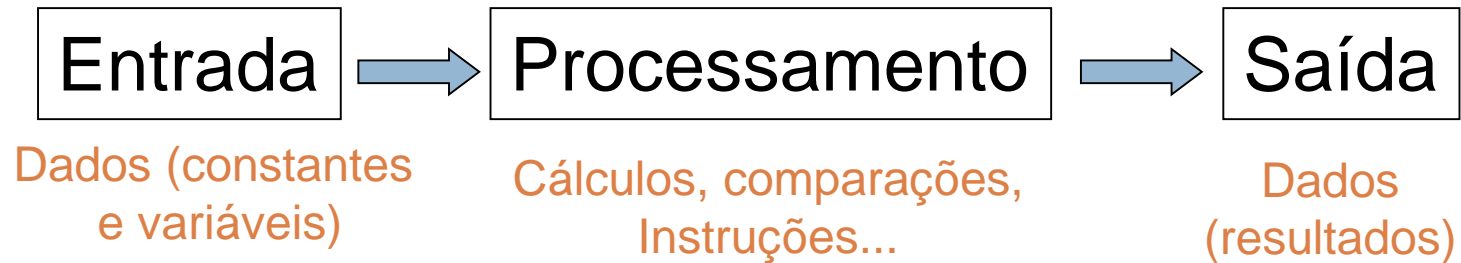
A screenshot of a Scilab script editor window. The script is written in Scilab's own language and calculates the roots of a quadratic equation. It includes comments in Portuguese, prompts the user for coefficients a, b, and c, calculates the discriminant (delta), and then calculates the two roots using the quadratic formula.

```
1 // Programa que calcula as raizes
2 // de uma equação de 2o grau
3 a = input("Entre com o valor de a: ")
4 b = input("b = ")
5 c = input("c = ")
6 delta = b^2 - 4*a*c
7 r1 = (-b+sqrt(delta))/(2*a)
8 r2 = (-b- sqrt(delta))/(2 * a)
9
```

Scilab

# Software

32



- **Entrada:**
  - ▣ Dados utilizados no processamento;
- **Processamento:**
  - ▣ Manipulação de variáveis e constantes; resolução de expressões matemáticas; estruturas de decisão e de repetição de comandos.
- **Saída:**
  - ▣ Resultados de processamento.



# Software

33

- Em geral o software é abstraído diretamente ao algoritmo.
- **Algoritmo** é uma sequência finita de instruções bem definidas e não ambíguas, cada uma das quais devendo ser executadas mecânica ou eletronicamente em um intervalo de tempo finito e com uma quantidade de esforço finita. (Wikipédia)

- Exemplo de algoritmo para troca de lâmpada:
  - ▣ Passo 1 – Pegar uma lâmpada nova;
  - ▣ Passo 2 – Pegar uma escada;
  - ▣ Passo 3 – Posicionar a escada embaixo da lâmpada queimada;
  - ▣ Passo 4 – Subir na escada com a lâmpada nova na mão;
  - ▣ Passo 5 – Retirar a lâmpada queimada;
  - ▣ Passo 6 – Colocar a lâmpada nova;
  - ▣ Passo 7 – Descer da escada com a lâmpada queimada;
  - ▣ Passo 8 – Testar o interruptor;
  - ▣ Passo 9 – Guardar a escada;
  - ▣ Passo 10 – Jogar a lâmpada velha no lixo.

- Construção de um algoritmo:
  - ▣ Entender o problema;
  - ▣ Definir os dados de entrada;
  - ▣ Definir como a entrada será utilizada (processamento);
  - ▣ Definir os dados de saída;
  - ▣ Utilizar representação ou linguagem para descrever o procedimento.

- Construção de um algoritmo:
  - ▣ Entender o problema;
  - ▣ Definir os dados de entrada;
  - ▣ Definir como a entrada será utilizada (processamento);
  - ▣ Definir os dados de saída;
  - ▣ Utilizar representação ou linguagem para descrever o procedimento.
- Cálculo do IMC baseado na fórmula:

$$IMC = \frac{peso}{altura^2}$$

# Resumo – Parte 1

37

- Computador: recebe, manipula e armazena dados.
- Hardware: parte física do computador.
  - ▣ Utiliza bits e bytes par representar dados.
  - ▣ Tabela ASCII converte bytes em caracteres.
  - ▣ Memória: células que armazenam bytes organizadas em endereços.
  - ▣ CPU: responsável pelo processamento lógico e aritmético dos dados.

# Resumo – Parte 1

38

- Software: parte lógica, programas.
  - ▣ Subdividido em entrada, processamento e saída.
  - ▣ Algoritmo: sequência bem definida de passos.
  - ▣ Implementado em diferentes linguagens de programação, dependendo do objetivo.

# Algoritmo

39

- Formas de **representação de algoritmos**:
  - ▣ Descrição narrativa:
    - Palavras em linguagem natural, lista de passos.
  - ▣ Fluxograma:
    - Diagrama de blocos conectados em sequencia.
  - ▣ Pseudocódigo:
    - Intermediário entre a linguagem natural e uma linguagem de programação.
  - ▣ Linguagem de programação:
    - Linguagem C, Matlab, Scilab, Java.

# Algoritmo: Descrição Narrativa

40

- ❑ **Descrição narrativa:** consiste no uso de frases para expressar ações a serem realizadas.
- ❑ Apresenta a facilidade da linguagem ser conhecida e o inconveniente da ambiguidade de termos.
- ❑ Procedimento para elaborar um algoritmo em descrição narrativa:
  - ▣ **Definição dos dados de entrada**
  - ▣ **Processamento (instruções a serem realizadas)**
  - ▣ **Definição dos dados de saída**



# Algoritmo: Descrição Narrativa

41

- Problema:
  - ▣ Calcular a média de duas notas.
- Solução:
  - ▣ Somar as duas notas e dividir a soma por dois.
- Dados de entrada:
  - ▣ Duas notas.
- Processamento (instruções realizadas):
  - ▣ Somar as duas notas.
  - ▣ Dividir a soma das notas por 2.
- Dados de saída:
  - ▣ Média.




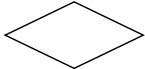


# Algoritmo: Fluxograma

42

- ❑ **Fluxograma:** diagrama de fluxo ou diagrama de blocos.
- ❑ Representação gráfica que utiliza formas geométricas ligadas por setas para indicar sequencia de instruções.
- ❑ Facilita a visualização da sequencia de instruções.
- ❑ Segue-se a norma ISO 5807 para os símbolos utilizados.

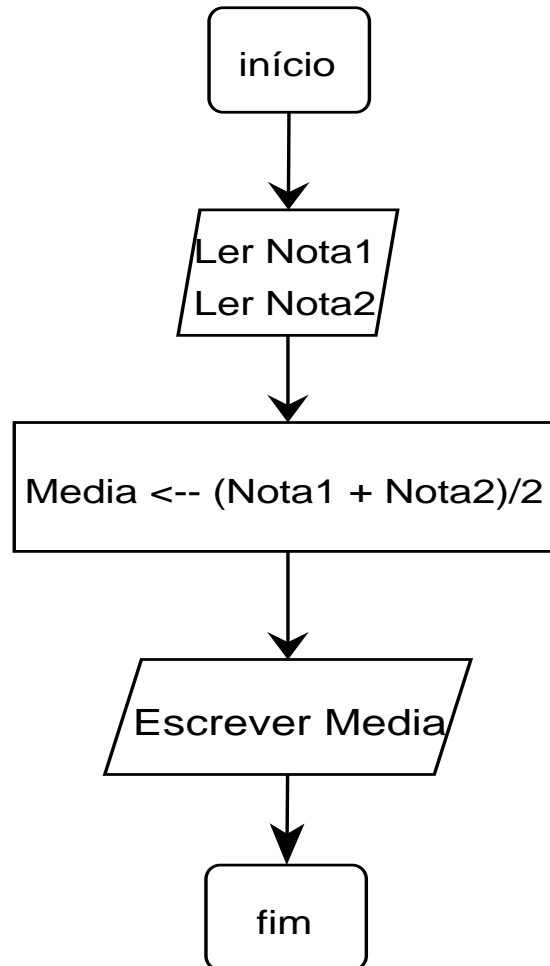
# Algoritmo: Fluxograma

43

Símbolo	Descrição
	Terminal: indica o início e fim do algoritmo.
	Entrada e saída. Receber e mostrar informações. Ler dados para armazenar em variáveis e mostrar dados contidos em variáveis. Escrever texto.
	Processamento: realizar operações com variáveis e constantes; executar as instruções contidas em estruturas de decisão e de repetição.
	Representação de decisão, divisão do fluxo em caminhos diferentes.
	Conector para agrupar fluxos.
	Indica o sentido do fluxo de dados.

# Algoritmo: Fluxograma

44



# Algoritmo: Pseudocódigo

45

- ❑ **Pseudocódigo:** também conhecido por português estruturado, é uma forma genérica de expressar um código utilizando termos da língua natural.
- ❑ Semelhante ao formato dos códigos em linguagem de programação.
- ❑ Formato geral:  
    **Algoritmo** <nome\_do\_algoritmo>  
        <declaração\_de\_variáveis>  
    **Início**  
        <instruções>  
    **Fim**

# Algoritmo: Pseudocódigo

46

```
Algoritmo media;  
Declare N1,N2 : inteiro;  
        media : real;  
Inicio  
    Ler (N1);  
    Ler (N2);  
    media  $\leftarrow$  (N1+N2) / 2;  
    SE (media >= 60) ENTÃO  
        Escrever ("Aluno aprovado com média: ", media)  
    SENÃO  
        Escrever ("Aluno reprovado com média: ", media)  
    FIMSE  
Fim.
```

# Algoritmo: Linguagem de Programação

47

- **Linguagem de programação:** define o conjunto de símbolos e as regras para expressar instruções computacionais.
  - ▣ Léxico: conjunto de palavras especiais.
  - ▣ Sintaxe: concordância e ordem das palavras.
  - ▣ Semântica: significado da sequência de palavras.
- O conjunto de palavras, seguindo as regras, constitui o **código fonte**, que será traduzido e executado pelo processador.

# Algoritmo: Linguagem de Programação

48

```
media.c
#include <stdio.h>

int main(void)
{
    float nota1, nota2, media;

    scanf("%f", &nota1);
    scanf("%f", &nota2);

    media = (nota1 + nota2)/2;

    printf("%f", media);

    return 0;
}
```



# Resumo – Parte 2

49

- Formas de **representação de algoritmos**:
  - ▣ Descrição narrativa:
    - Palavras em linguagem natural, lista de passos.
  - ▣ Fluxograma:
    - Diagrama de blocos conectados em sequência.
  - ▣ Pseudocódigo:
    - Intermediário entre a linguagem natural e uma linguagem de programação.
  - ▣ Linguagem de programação:
    - Linguagem C, Matlab, Scilab, Java.

# Visão lúdica da execução de uma instrução

50

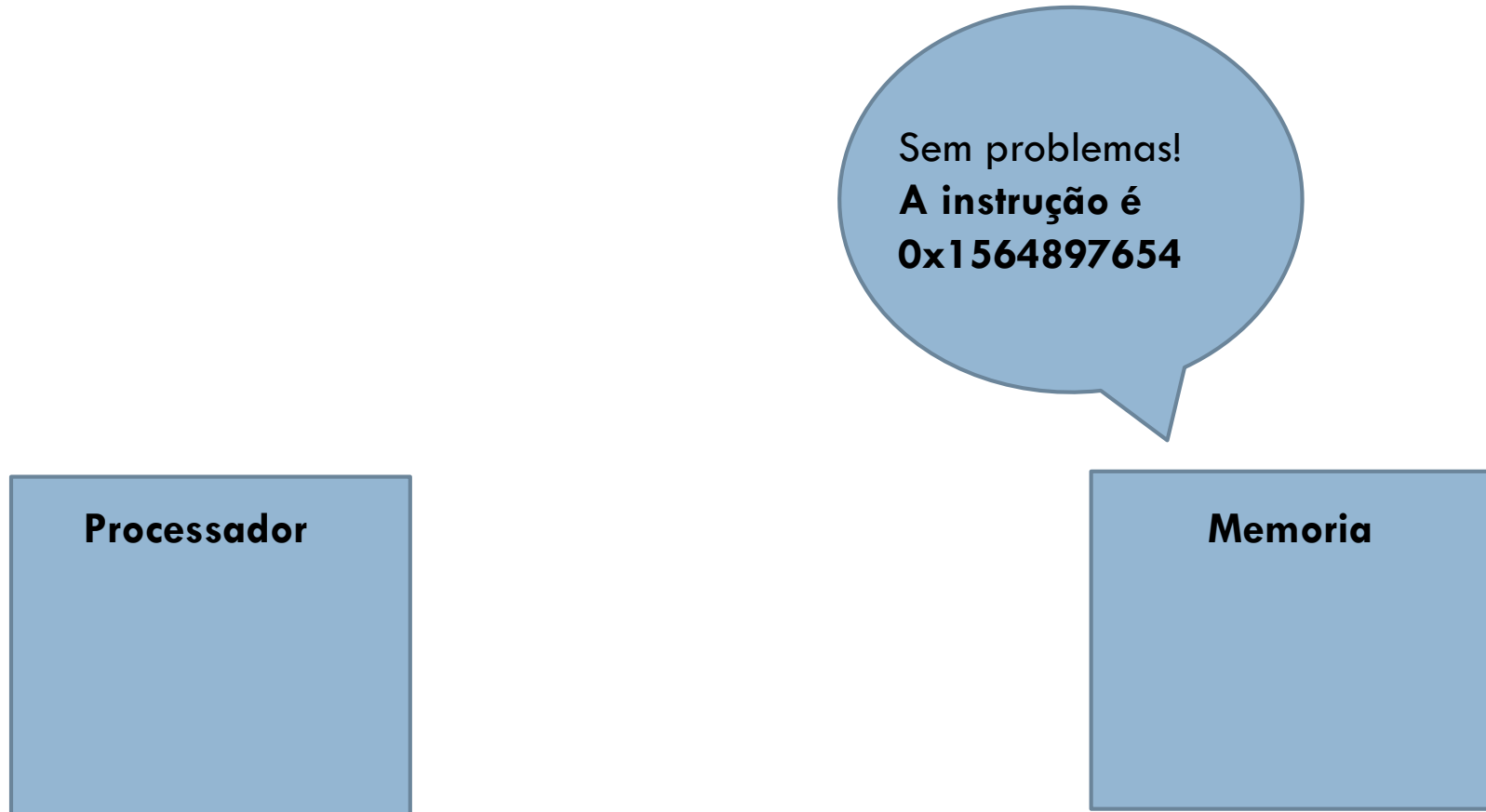
Memória, meu registrador **Program Counter** me diz que a próxima instrução que eu devo executar tem endereço **0x90907070**  
**Me diga qual é a instrução!**

**Processador**

**Memoria**

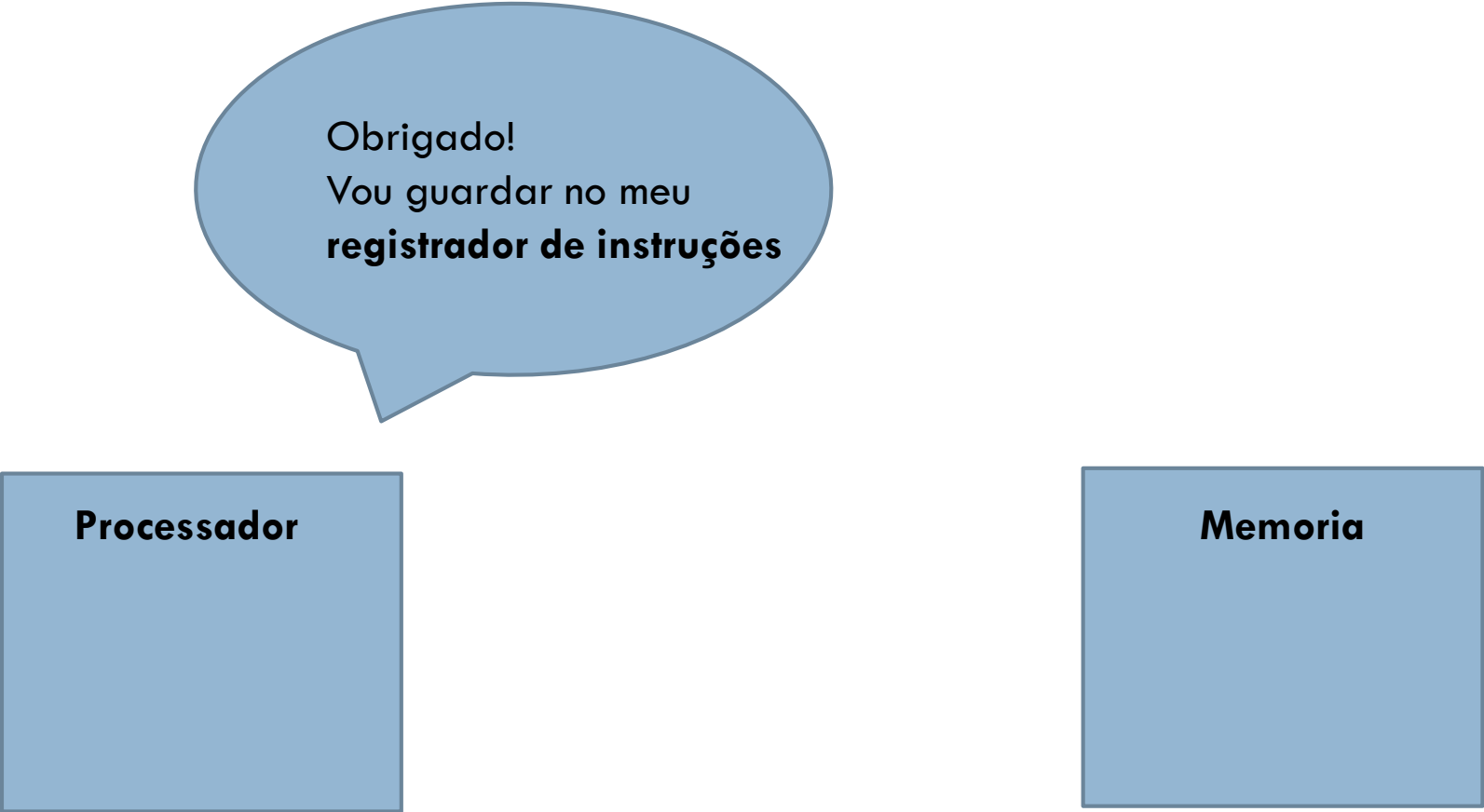
# Visão lúdica da execução de uma instrução

51



# Visão lúdica da execução de uma instrução

52



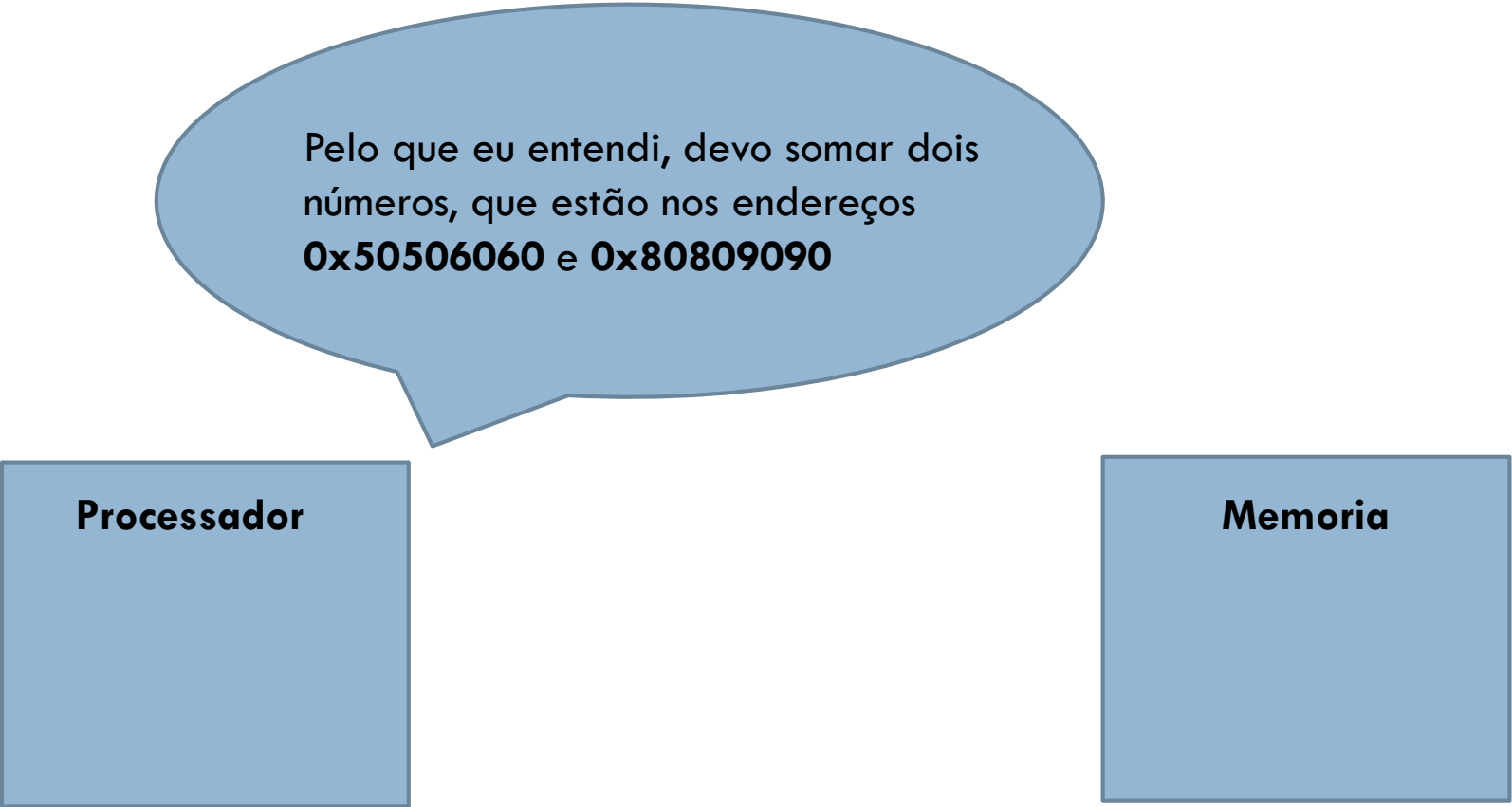
Obrigado!  
Vou guardar no meu  
**registrador de instruções**

**Processador**

**Memoria**

# Visão lúdica da execução de uma instrução

53



Pelo que eu entendi, devo somar dois números, que estão nos endereços **0x50506060** e **0x80809090**

**Processador**

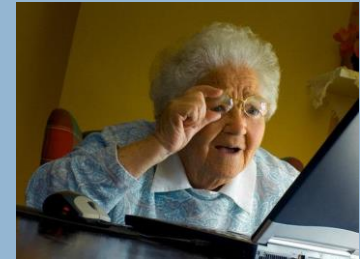
**Memoria**

# Visão lúdica da execução de uma instrução

54

**Processador**

**Memoria**



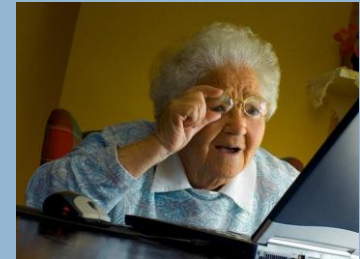
# Visão lúdica da execução de uma instrução

55

**Nossa, como você é lerda memória,  
enquanto eu trabalho em Gigahertz  
você trabalha em Megahertz, eu penso  
algumas milhares de vezes mais  
rápido que você!!!**

**Processador**

**Memoria**



# Visão lúdica da execução de uma instrução

56

É exatamente por isso que você tem memória cache!!!  
Enfim, já busquei os valores nesses endereços, que são 3 e 4.

**Processador**

**Memoria**





# Visão lúdica da execução de uma instrução

57

Certo, o resultado é 7, a instrução diz que é pra guardar esse resultado no endereço **0x45647981**

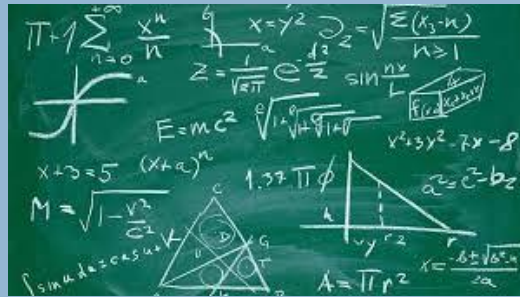
**Processador**



**Memoria**

# Visão lúdica da execução de uma instrução

58



**Processador**



**Memoria**

# Visão lúdica da execução de uma instrução

59

Falando em cache, meu algoritmo de escalonamento me diz que existe alta probabilidade de eu usar o que tem guardado desde o endereço 0x10101010 até o endereço 0x101010FF, me dê esses valores para eu guardar na minha memória cache, dessa forma não preciso de você por um tempo.

**Processador**



**Memoria**

# Visão lúdica da execução de uma instrução

60

