O COMPUTADOR POR DENTRO

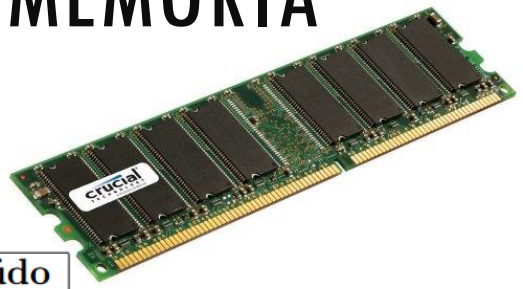
Prof. André Vignatti – DINF - UFPR

AULA PASSADA: Como **0's** e **1's** se tornam **números**, **palavras**, **imagens**, **sons**, **vídeos**?

AULA DE HOJE:

COMO 0's E 1's SE TORNAM PROGRAMAS?

“FOTOGRAFIA” DE UM PROGRAMA NA MEMÓRIA



Endereço	Conteúdo
0	1
1	54
2	2
3	1
4	50
5	4
6	7
7	46
8	4
9	47
10	46
11	46
12	7
13	48
14	4
15	49
16	50
17	48
18	3
19	51
20	47

Endereço	Conteúdo
21	49
22	6
23	52
24	51
25	3
26	53
27	46
28	52
29	5
30	55
31	53
32	54
33	8
34	55
35	2
36	56
37	46
38	52
39	5
40	57
41	56

Endereço	Conteúdo
42	54
43	8
44	57
45	9
46	33
47	2
48	76
49	67
50	76
51	124
52	14
53	47
54	235
55	35
56	23
57	78
58	243
59	27
60	88
61	12
62	12

ENDEREÇO X CONTEÚDO DO ENDEREÇO

A memória é **endereçada**

- Dado um endereço, podemos obter o **conteúdo** armazenado naquele endereço

Notação:

- **p** é um endereço
- **[p]** é o conteúdo do endereço **p**

Exemplo:

- [0] = 1
- [[0]] = [1] = 54
- [[[0]]] = [[1]] = [54] = 235
- [0] + 1 = 1 + 1 = 2

INSTRUÇÕES ACEITAS PELA CPU FICTÍCIA

Código	Mnemônico	Descrição	Notação
1	`load`	escreva em $[p+1]$ o valor do número em $[p+2]$ e some 3 em p	$[p+1] \leftarrow [p+2]$.
2	`add`	escreva em $[p+1]$ o valor da soma dos números em $[[p+2]]$ e $[[p+3]]$ e some 4 em p	$[p+1] \leftarrow [[p+2]] + [[p+3]]$.
3	`sub`	escreva em $[p+1]$ o valor da subtração do número em $[[p+2]]$ pelo número em $[[p+3]]$ e some 4 em p	$[p+1] \leftarrow [[p+2]] - [[p+3]]$.
4	`mult`	escreva em $[p+1]$ o valor do produto dos números em $[[p+2]]$ e $[[p+3]]$ e some 4 em p	$[p+1] \leftarrow [[p+2]] \times [[p+3]]$.
5	`div`	escreva em $[p+1]$ o valor da divisão do número em $[[p+2]]$ pelo número em $[[p+3]]$ e some 4 em p	$[p+1] \leftarrow \frac{[[p+2]]}{[[p+3]]}$.
6	`sqrt`	escreva em $[p+1]$ o valor da raiz quadrada de $[[p+2]]$ e some 3 em p	$[p+1] \leftarrow \sqrt{[[p+2]]}$.
7	`read`	leia um número do teclado, escreva-o em $[p+1]$ e some 2 em p	$[p+1] \leftarrow \underline{\vee}$.
8	`write`	escreva $[[p+1]]$ na tala e some 2 em p	$\square \leftarrow [[p+1]]$.
9	`stop`	pare	$\bullet$

Quase 1000 instruções

> 1000 instruções

CICLO DE EXECUÇÃO DAS INSTRUÇÕES

1. comece com $p = 0$

2. interprete [p] de acordo com a tabela de instruções

3. pare somente se a instrução for uma ordem de parar (instrução 9, stop)

SEPARANDO AS INSTRUÇÕES

Endereço	Instrução	Operando	Operando	Operando
0	1	54	2	
3	1	50	4	
6	7	46		
8	4	47	46	46
12	7	48		
14	4	49	50	48
18	3	51	47	49
22	6	52	51	
25	3	53	46	52
29	5	55	53	54
33	8	55		
35	2	56	46	52
39	5	57	56	54
43	8	57		
45	9			

ELIMINANDO ENDEREÇO DAS INSTRUÇÕES

Se sabemos a sequência de instruções, então o endereço agora é desnecessário

Instrução	Operando	Operando	Operando
1	54	2	
1	50	4	
7	46		
4	47	46	46
7	48		
4	49	50	48
3	51	47	49
6	52	51	
3	53	46	52
5	55	53	54
8	55		
2	56	46	52
5	57	56	54
8	57		
9			

USANDO NOMES COM SIGNIFICADO

Mnemônicos: nomes que significam algo para os humanos

Instrução	Operando	Operando	Operando
`load`	54	2	
`load`	50	4	
`read`	46		
`mult`	47	46	46
`read`	48		
`mult`	49	50	48
`sub`	51	47	49
`sqrt`	52	51	
`sub`	53	46	52
`div`	55	53	54
`write`	55		
`add`	56	46	52
`div`	57	56	54
`write`	57		
`stop`			

UMA NOTAÇÃO PARA INSTRUÇÕES

Notação: facilita a "visualização" de coisas complicadas

$[p]$	**Instrução**	**Notação**
1	`load` $x\ y$	$x \leftarrow [y]$
2	`add` $x\ y\ z$	$x \leftarrow [y] + [z]$
3	`sub` $x\ y\ z$	$x \leftarrow [y] - [z]$
4	`mult` $x\ y\ z$	$x \leftarrow [y] \times [z]$
5	`div` $x\ y\ z$	$x \leftarrow \frac{[y]}{[z]}$
6	`sqrt` $x\ y$	$x \leftarrow \sqrt{[y]}$
7	`read` x	$x \leftarrow V$
8	`write` x	$V \leftarrow [x]$
9	`stop`	$\bullet$

$$
\begin{array}{lll}
54 & \leftarrow & 2 \\
50 & \leftarrow & 4 \\
46 & \leftarrow & \underline{\vee} \\
47 & \leftarrow & [46] \times [46] \\
48 & \leftarrow & \underline{\vee} \\
49 & \leftarrow & [50] \times [48] \\
51 & \leftarrow & [47] - [49] \\
52 & \leftarrow & \sqrt{[[51]]} \\
53 & \leftarrow & [46] - [52] \\
55 & \leftarrow & \frac{[53]}{[54]} \\
\square & \leftarrow & [55] \\
56 & \leftarrow & [46] + [52] \\
57 & \leftarrow & \frac{[56]}{[54]} \\
\square & \leftarrow & [57] \\
\bullet & &
\end{array}
$$

DANDO NOMES PARA ENDEREÇOS

Novamente: usar **mnemônicos**

Novos nomes :

Endereço	Nome
54	dois
50	quatro
46	B
47	quadradoB
48	C
49	quadruploC
51	discriminante
52	raizDiscriminante
53	dobroMenorRaiz
55	menorRaiz
56	dobroMaiorRaiz
57	maiorRaiz

Programa com novos nomes:

dois	←	2
quatro	←	4
B	←	$\underline{\vee}$
quadradoB	←	B × B
C	←	$\underline{\vee}$
quadruploC	←	quatro × C
discriminante	←	quadradoB - quadruploC
raizDiscriminante	←	$\sqrt{discriminante}$
dobroMenorRaiz	←	B - raizDiscriminante
menorRaiz	←	$\frac{dobroMenorRaiz}{dois}$
□	←	menorRaiz
dobroMaiorRaiz	←	B + raizDiscriminante
maiorRaiz	←	$\frac{dobroMaiorRaiz}{dois}$
□	←	maiorRaiz
•		

DUAS VERSÕES MAIS SIMPLES

read B
read C
discriminante $\leftarrow$ B^2 - 4 $\times$ C
raizDiscriminante $\leftarrow \sqrt{discriminante}$
menorRaiz $\leftarrow \frac{B - raizDiscriminante}{2}$
write menorRaiz
maiorRaiz $\leftarrow \frac{B + raizDiscriminante}{2}$
write maiorRaiz

read B
read C
raizDiscriminante $\leftarrow \sqrt{B^2 - 4 \times C}$
write $\frac{\text{B - raizDiscriminante}}{2}$
write $\frac{\text{C + raizDiscriminante}}{2}$

VERSÃO EM LINGUAGEM PASCAL

```
program bhaskara (input,output);
var b, c, raizdiscriminante: real;

begin
      read (b);
      read (c);
      raizdiscriminante:= sqrt(b*b - 4*c);
      write ((b - raizdiscriminante)/2);
      write ((b + raizdiscriminante)/2);
end.
```

COMPILADOR

Compilador: programa que faz **isso** virar **isso**

```
program bhaskara (input,output);
var b, c, raizdiscriminante: real;

begin
      read (b);
      read (c);
      raizdiscriminante:= sqrt(b*b - 4*c);
      write ((b - raizdiscriminante)/2);
      write ((b + raizdiscriminante)/2);
end.
```

Endereço	Conteúdo
0	1
1	54
2	2
3	1
4	50
5	4
6	7
7	46
8	4
9	47
10	46
11	46
12	7
13	48
14	4
15	49
16	50
17	48
18	3
19	51
20	47

Endereço	Conteúdo
21	49
22	6
23	52
24	51
25	3
26	53
27	46
28	52
29	5
30	55
31	53
32	54
33	8
34	55
35	2
36	56
37	46
38	52
39	5
40	57
41	56

Endereço	Conteúdo
42	54
43	8
44	57
45	9
46	33
47	2
48	76
49	67
50	76
51	124
52	14
53	47
54	235
55	35
56	23
57	78
58	243
59	27
60	88
61	12
62	12

Alto nível X Baixo nível

```
11010101010101001010001 0
10010111100000100010010 1
01010101010100101110001 1
01000101001010111000100 1
01010101000011111010101 0
01010101010101001000011
```

O QUE FAREMOS A PARTIR DE AGORA?

Escrever programas em **linguagem de alto nível**

CI055: Pascal
Próximos 4 anos: C, Java, Prolog, ...

Mandar um **compilador** gerar o **código de máquina**

CI055: compilador Free Pascal

Mandar a máquina **executar** o código gerado

Tarefa de casa: aprender **compilar** e **executar** um programa:
http://www.inf.ufpr.br/cursos/ci055/
(Guia rápido Pascal e Guia Rápido Linux)