

# Organização Básica de computadores e linguagem de montagem

Prof. Edson Borin

2º Semestre de 2019

# Regras do Curso

- Regras em:

<http://www.ic.unicamp.br/~edson/disciplinas/mc404/2019-2s/ab/index.html>

Familiarize-se com os critérios de avaliação e as datas das provas e do exame!

# Por que Aprender Linguagem de Montagem

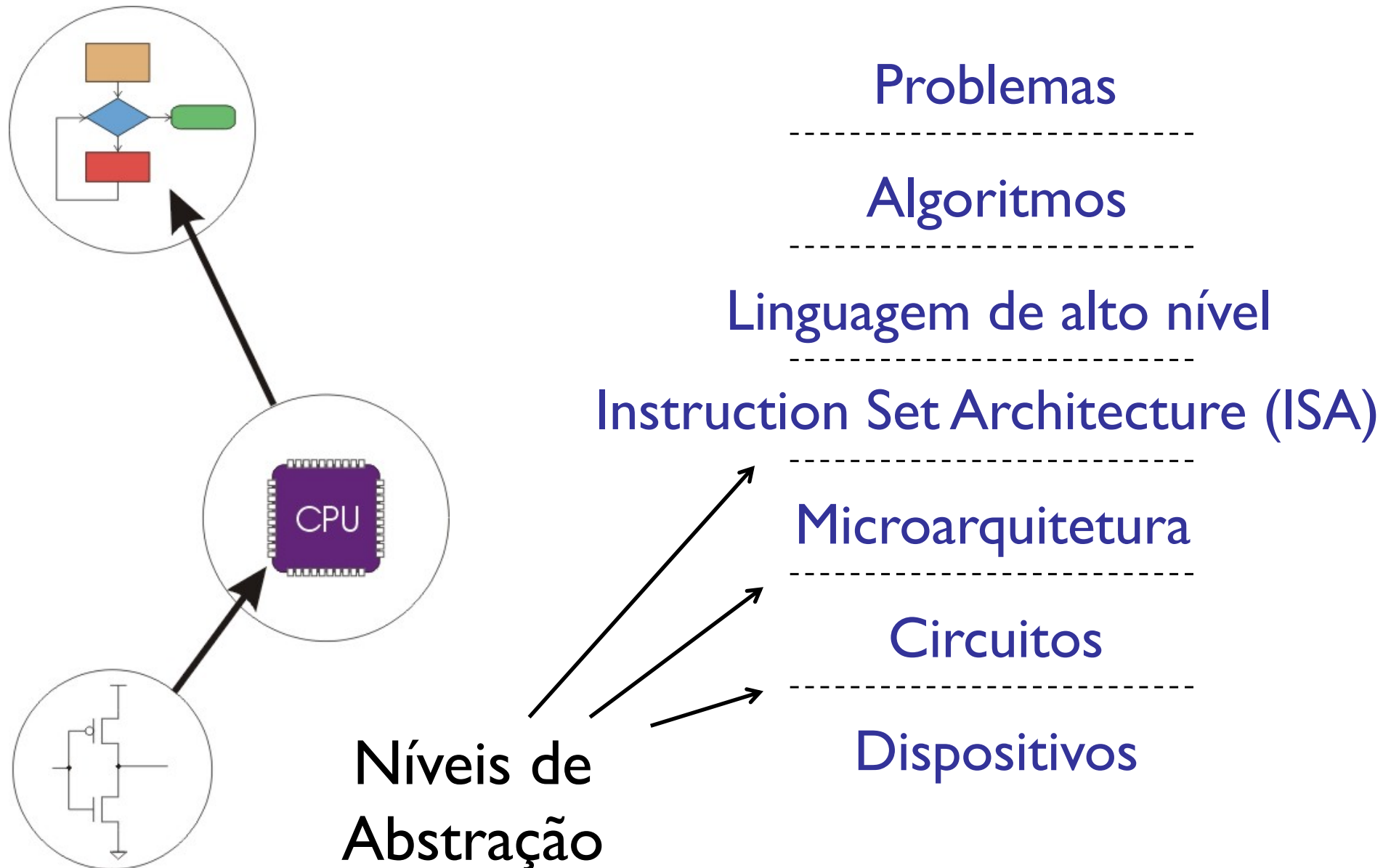
- Permite compreender o funcionamento de uma CPU
- Utilizado na:
  - Programação de máquinas baseadas em micro-controladores.
  - Programação de sistemas embarcados (*embedded systems*)
  - Programação de trechos críticos (tempo e/ou memória)
  - Acesso a recursos não disponíveis em alto nível
- OBS.:A linguagem de montagem é absolutamente ligada ao hardware, depende de cada máquina específica (diferentemente das linguagens de alto nível, como C, C++ e Java)

# Por que Aprender Linguagem de Montagem

- Permite entender como programas escritos em linguagens de alto nível, como C ou Java, são traduzidos para a linguagem de máquina.

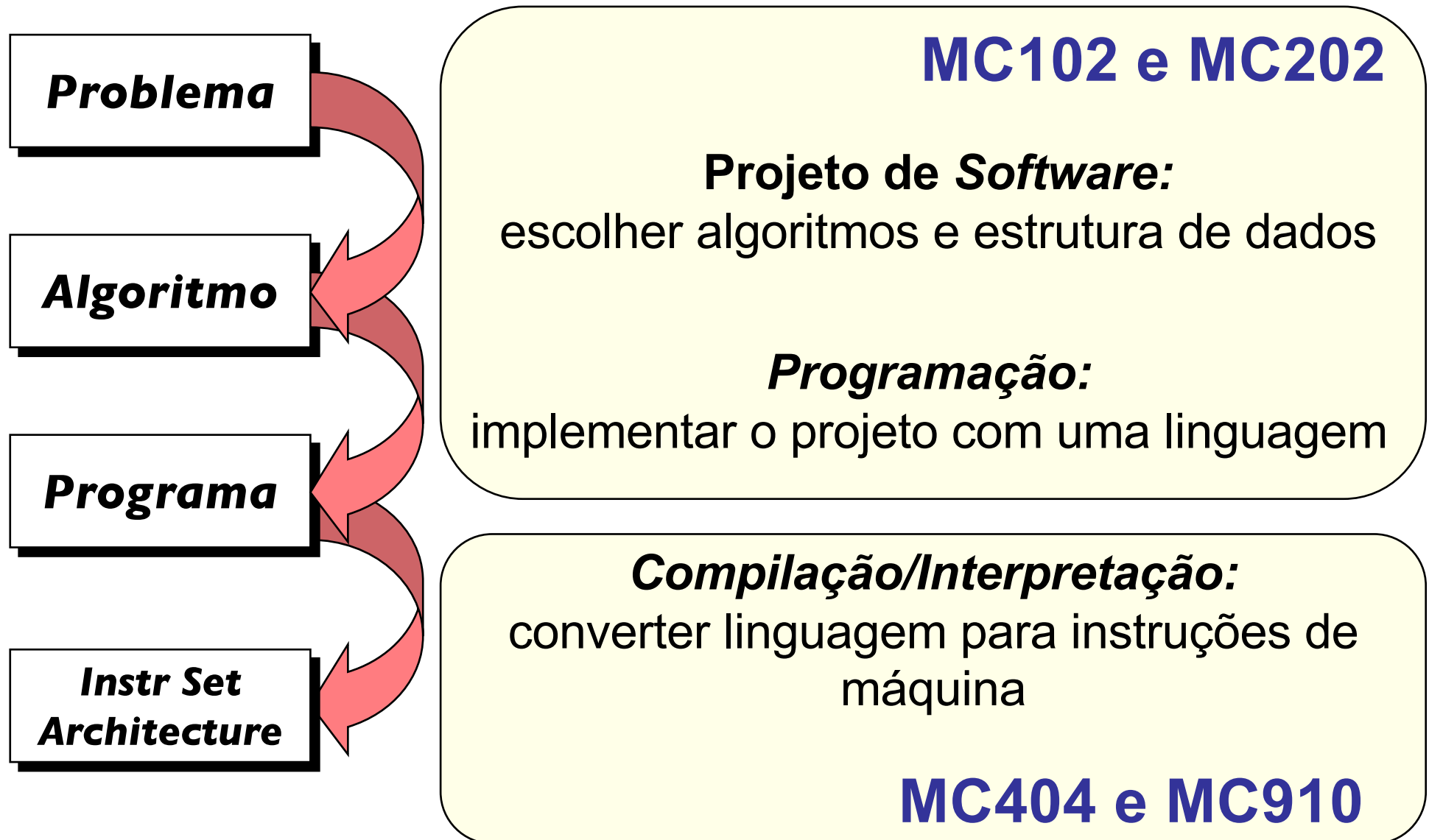
# Conceitos Básicos

# Resolução de problemas com Computadores



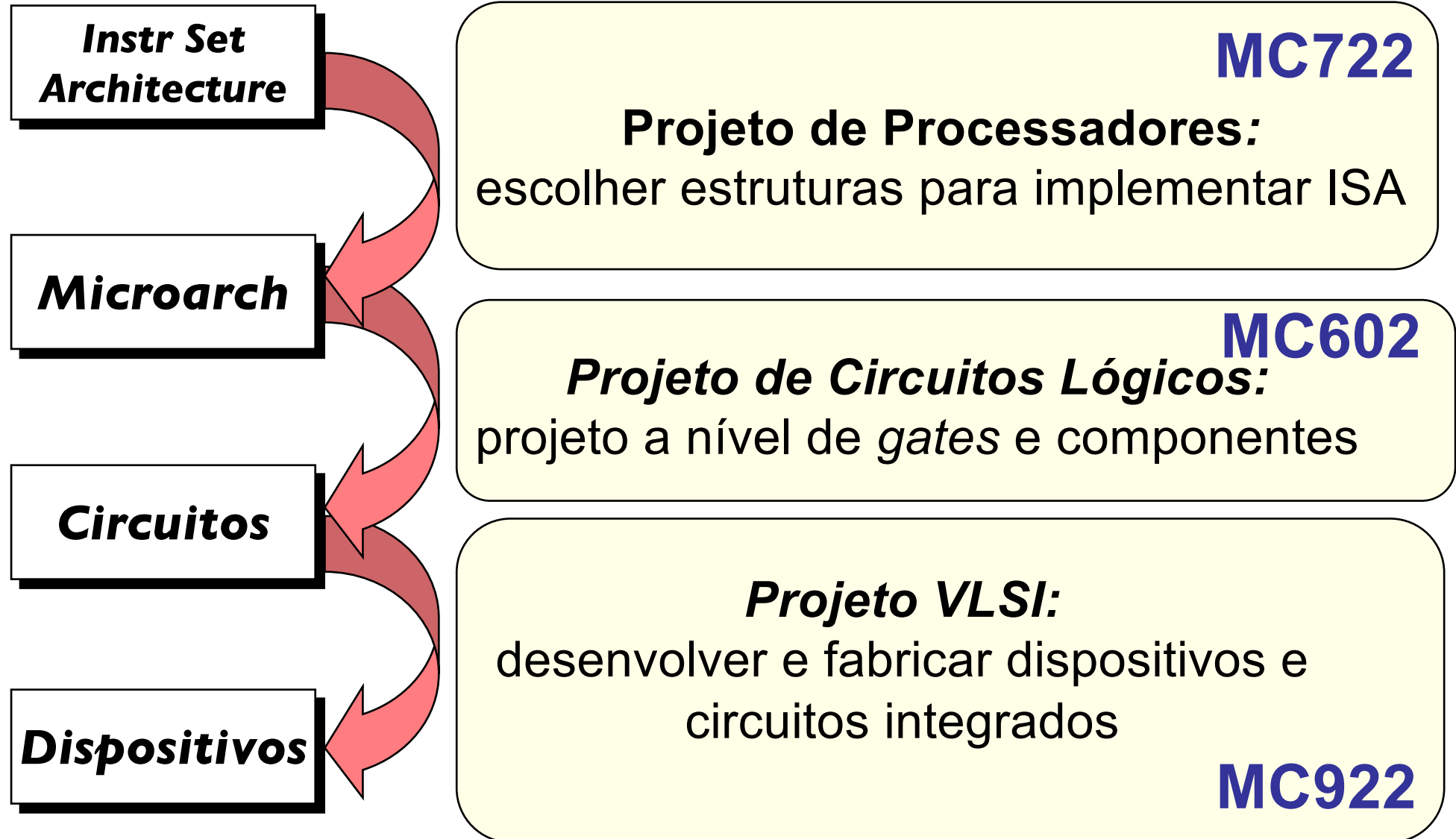
\* Cortesia do Prof. Mário Cortes

# Resolução de problemas com Computadores



\* Cortesia do Prof. Mário Cortes

# Resolução de problemas com Computadores



\* Cortesia do Prof. Mário Cortes

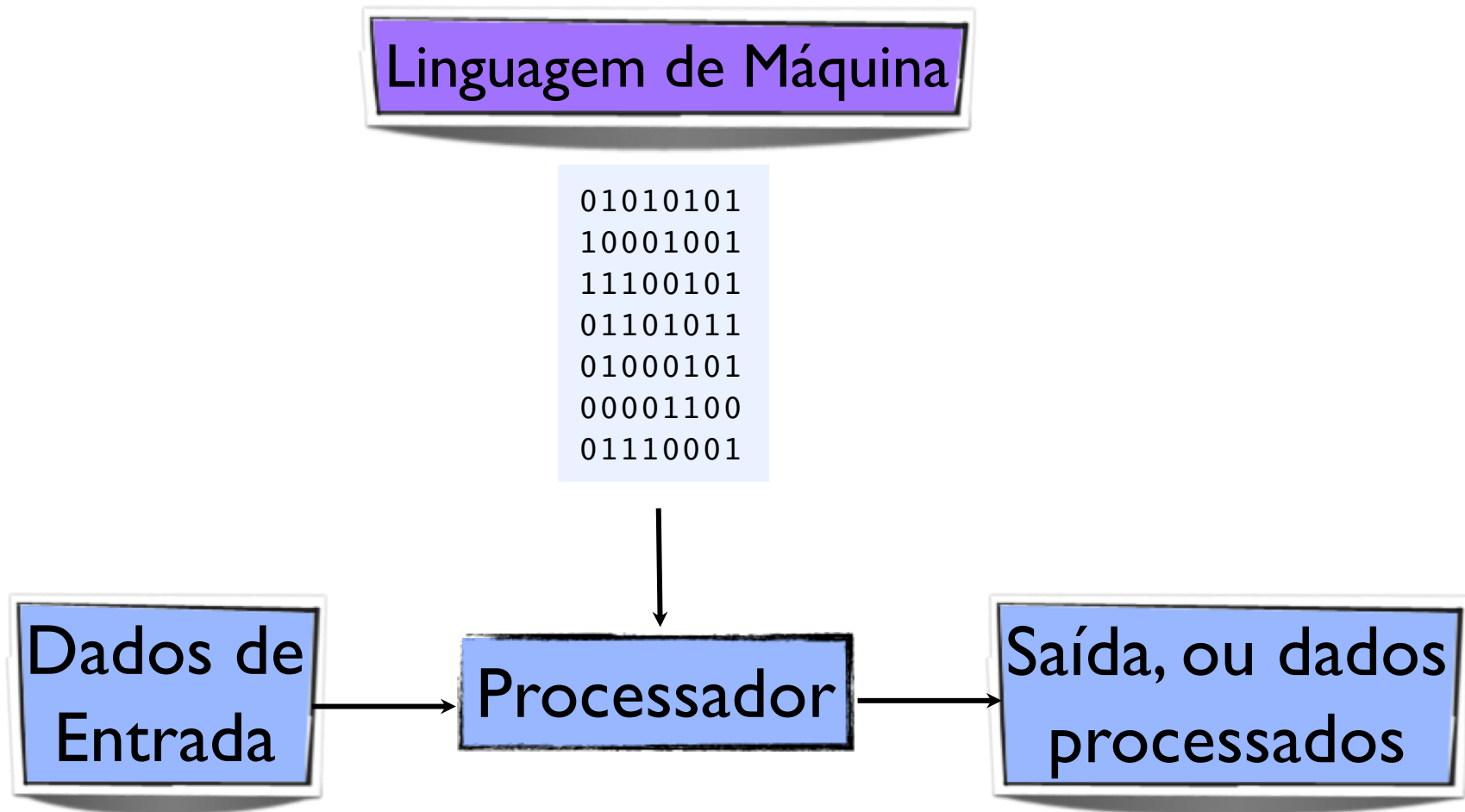


# Conceitos Básicos: Computadores

Máquinas para manipular informações (ou dados)



# Conceitos Básicos: Computadores



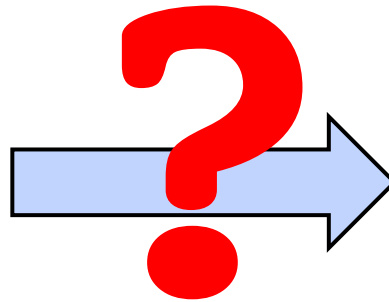
# Conceitos Básicos: Linguagens de Programação

Programa fonte

```
int func(int a)
{
    return a*113;
}
```

Linguagem de Máquina

```
01010101
10001001
11100101
01101011
01000101
00001100
01110001
```

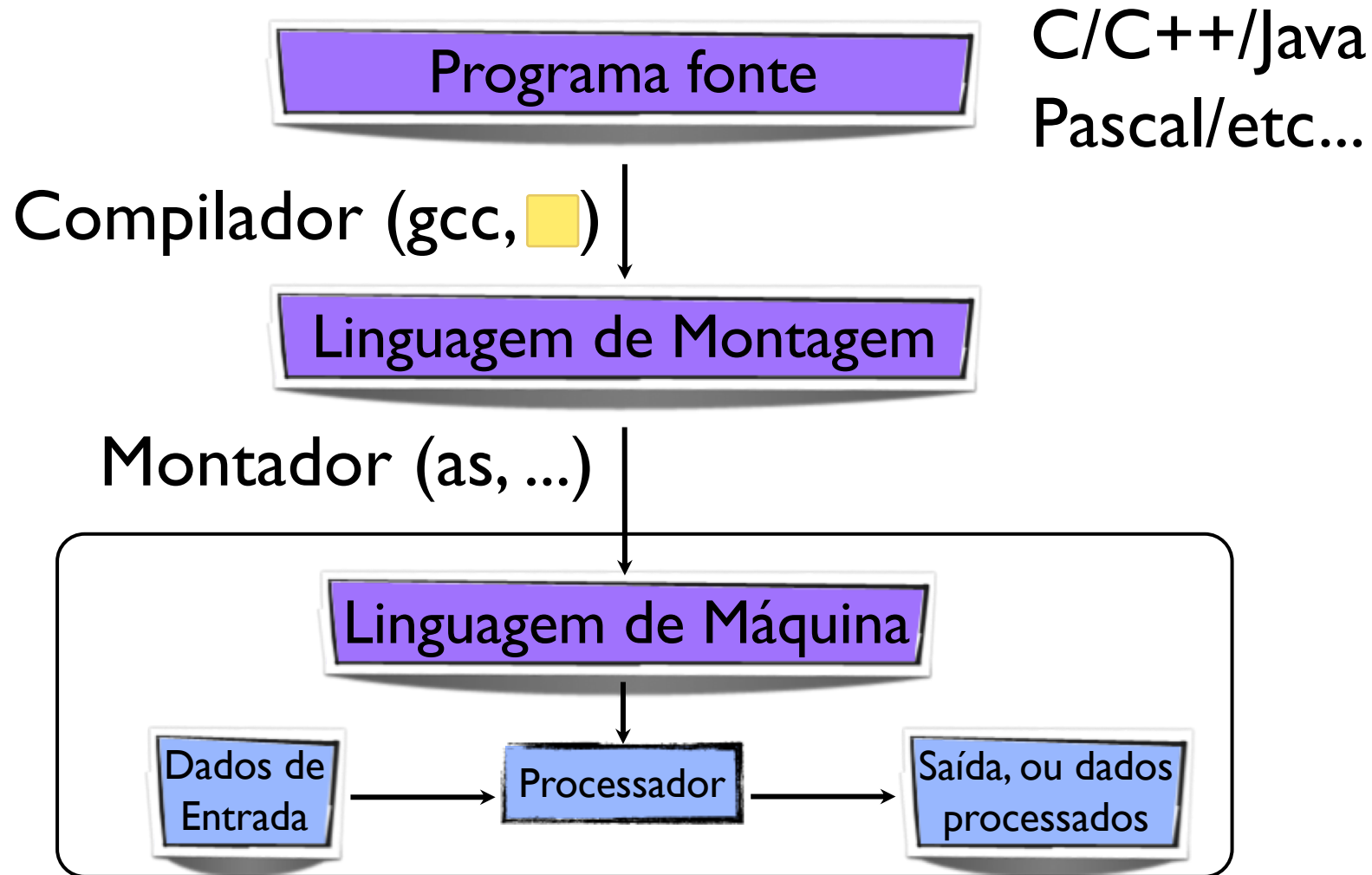


Dados de  
Entrada

Processador

Saída, ou dados  
processados

# Conceitos Básicos: Linguagens de Programação



# Conceitos Básicos: Linguagens de Programação

Programa fonte



Linguagem de Montagem



Linguagem de Máquina

Ling. de alto nível (Java, ...)  
Laços, variáveis, objetos, ...  
Independente de máquina

Ling. de baixo nível  
Sequência de instruções, registradores,  
posições de memória, ...  
Dependente de máquina

Ling. de máquina  
Codificada de forma binária (0s e 1s)  
Dependente de máquina

# Conceitos Básicos: Linguagens de Programação

Programa fonte



Linguagem de Montagem

Programa fonte na linguagem C

```
int func_1(int a, int b, int c)
{
    return (a + (113 * b)) * c;
}
```

Linguagem de Montagem do x86

```
_func_1:
    push    %ebp
    mov     %esp, %ebp
    imul    $113, 12(%ebp), %eax
    add     8(%ebp), %eax
    imul    16(%ebp), %eax
    pop     %ebp
    ret
```

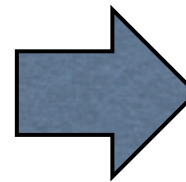
# Conceitos Básicos: Linguagens de Programação

Linguagem de  
Montagem

Montador

Linguagem de  
Máquina

```
_func_1:  
  push  %ebp  
  mov   %esp, %ebp  
  imul  $113, 12(%ebp), %eax  
  add   8(%ebp), %eax  
  imul  16(%ebp), %eax  
  pop   %ebp  
  ret
```



```
01010101  
10001001  
11100101  
01101011  
01000101  
00001100  
01110001  
00000011  
01000101  
00001000  
00001111  
10101111  
01000101  
00010000  
01011101  
11000011
```

# Conceitos Básicos: Linguagens de Programação

Linguagem de  
Montagem

Montador

Linguagem de  
Máquina

```
_func_1:  
  push  %ebp  
  mov   %esp, %ebp  
  imul  $113, 12(%ebp), %eax  
  add   8(%ebp), %eax  
  imul  16(%ebp), %eax  
  pop   %ebp  
  ret
```

```
01010101  
10001001  
11100101  
01101011  
01000101  
00001100  
01110001  
00000011  
01000101  
00001000  
00001111  
10101111  
01000101  
00010000  
01011101  
11000011
```



# Conceitos Básicos: Linguagens de Programação



objdump -D arquivo.o

```
00000000 <_func_1>:
 0: 55                push    %ebp
 1: 89 e5             mov     %esp,%ebp
 3: 6b 45 0c 71       imul    $0x71,0xc(%ebp),%eax
 7: 03 45 08           add     0x8(%ebp),%eax
 a: 0f af 45 10       imul    0x10(%ebp),%eax
 e: 5d                pop     %ebp
 f: c3                ret
```

# Conceitos Básicos: Linguagens de Programação

## Programa fonte na linguagem C

```
int func_1(int a, int b, int c)
{
    return (a + (113 * b)) * c;
}
```

## Linguagem de Montagem do **ARM**

```
_func_1:
    rsb    r3, r1, r1, asl #3
    add    r1, r1, r3, asl #4
    add    r1, r1, r0
    mul    r0, r2, r1
    bx     lr
```

## Linguagem de Montagem do **x86**

```
_func_1:
    push   %ebp
    mov    %esp, %ebp
    imul   $113, 12(%ebp), %eax
    add    8(%ebp), %eax
    imul   16(%ebp), %eax
    pop    %ebp
    ret
```

# Conceitos Básicos: Linguagens de Programação

## **Leitura**

- Capítulo 1.2 do livro do Patterson e Hennessy (Computer Organization and Design)