

**Instituto de
Computação**

UNIVERSIDADE ESTADUAL DE CAMPINAS



Organização Básica de computadores e linguagem de montagem

Controlando o fluxo de execução

Prof. Edson Borin

<https://www.ic.unicamp.br/~edson>

Institute of Computing - UNICAMP

Agenda

- **Sentenças condicionais**
- Laços de repetição
- Chamada e retorno de funções
- Exemplos

Controle do fluxo de execução

Sentença condicional “Se-Então” (`if-then`)

Exemplo: C/C++

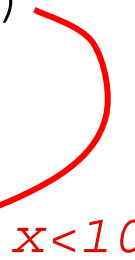
```
...  
if (x >= 10)  
{  
    y = x;  
}  
...
```

Controle do fluxo de execução

Sentença condicional “Se-Então” (if-then)

Exemplo: C/C++

```
...  
if (x >= 10)  
{  
  y = x;  
}  
...
```



Exemplo: Ling. de Montagem

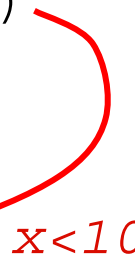
```
# x está em a3  
# y está em a4
```

Controle do fluxo de execução

Sentença condicional “Se-Então” (if-then)

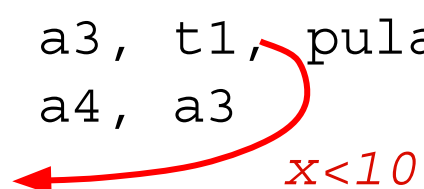
Exemplo: C/C++

```
...  
if (x >= 10)  
{  
  y = x;  
}  
...
```



Exemplo: Ling. de Montagem

```
# x está em a3  
# y está em a4  
...  
li  t1, 10  
blt a3, t1, pula # salta se a3 < 10  
mv  a4, a3  
pula:  
...
```



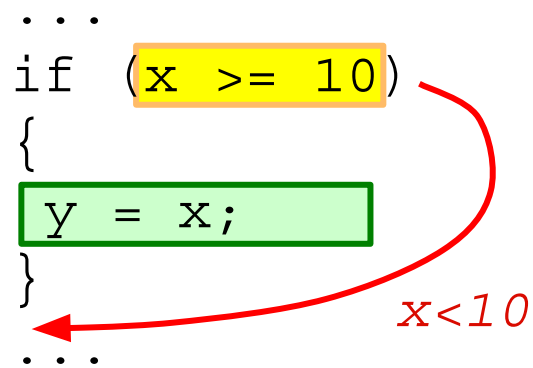
- `blt rs1, rs2, alvo`: salta para o alvo se o valor em `rs1` for menor que o valor em `rs2` (supondo números com sinal). Para números sem sinal use “`bltu`”.

Controle do fluxo de execução

Sentença condicional “Se-Então” (if-then)

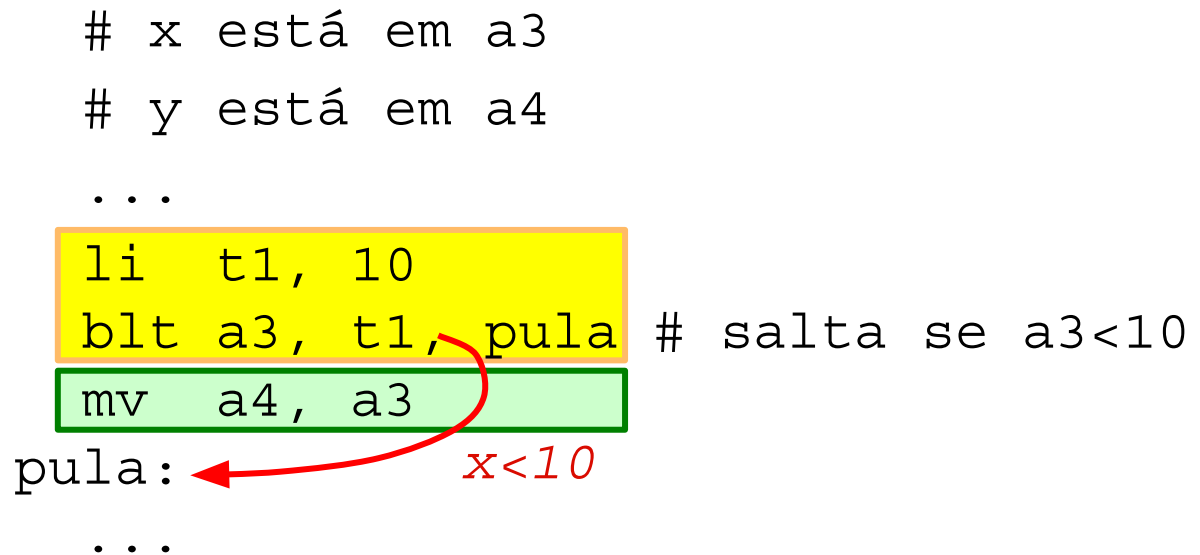
Exemplo: C/C++

```
...  
if (x >= 10)  
{  
    y = x;  
}  
...
```



Exemplo: Ling. de Montagem

```
# x está em a3  
# y está em a4  
...  
li t1, 10  
blt a3, t1, pula # salta se a3 < 10  
mv a4, a3  
pula:  
...
```



- `blt rs1, rs2, alvo`: salta para o alvo se o valor em `rs1` for menor que o valor em `rs2` (supondo números com sinal). Para números sem sinal use “`bltu`”.

Controle do fluxo de execução

Sentença condicional “Se-Então-Senão” (if-then-else)

Exemplo: C/C++

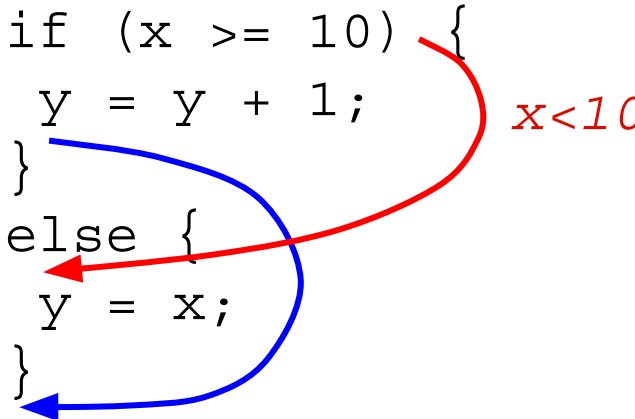
```
...  
if (x >= 10) {  
    y = y + 1;  
}  
else {  
    y = x;  
}  
...
```

Controle do fluxo de execução

Sentença condicional “Se-Então-Senão” (if-then-else)

Exemplo: C/C++

```
...  
if (x >= 10) {  
    y = y + 1;  
}  
else {  
    y = x;  
}  
...
```



Exemplo: Ling. de Montagem

```
# x está em a1  
# y está em a2
```


Controle do fluxo de execução

Sentença condicional “Se-Então-Senão” (if-then-else)

Exemplo: C/C++

```
...  
if (x >= 10) {  
    y = y + 1;  
}  
else {  
    y = x;  
}  
...
```

x < 10

Exemplo: Ling. de Montagem

```
# x está em a1  
# y está em a2  
...  
li    t3, 10  
blt  a1, t3, else  
addi a2, a2, 1  
j    cont  
else:  
    mv  a2, a1  
cont:  
...
```

x < 10

Controle do fluxo de execução

Sentença condicional “Se-Então-Senão” (if-then-else)

Exemplo: C/C++

```
...  
if (x >= 10) {  
    y = y + 1;  
}  
else {  
    y = x;  
}  
...
```

x < 10

Exemplo: Ling. de Montagem

```
# x está em a1  
# y está em a2  
...  
li    t3, 10  
blt   a1, t3, else  
addi  a2, a2, 1  
j     cont  
else:  
mv    a2, a1  
cont:  
...
```

x < 10

Controle do fluxo de execução

Sentença condicional com múltiplas condições

Exemplo: C/C++

```
...  
if ((x>=10) && (y<20))  
{  
    x = y;  
}  
...
```

Exemplo: Ling. de Montagem

```
# x está em a1  
# y está em a2
```

Controle do fluxo de execução

Sentença condicional com múltiplas condições

Exemplo: C/C++

```
...  
if ((x>=10) && (y<20))  
{  
    x = y;  
}  
...
```

Exemplo: Ling. de Montagem

```
# x está em a1  
# y está em a2  
...  
li    t1, 10          #Salta para  
blt  a1, t1, pula   #pula se x<10  
li    t1, 20          #Salta para  
bge  a2, t1, pula   #pula se y>=20  
mv   a1, a2          #x = y  
pula:  
...
```

Controle do fluxo de execução

Sentença condicional com múltiplas condições

Exemplo: C/C++

```
...  
if ( (x >= 10) && (y < 20) )  
{  
    x = y;  
}  
...
```

Exemplo: Ling. de Montagem

```
# x está em a1  
# y está em a2  
...  
li t1, 10 #Salta para  
blt a1, t1, pula #pula se x < 10  
li t1, 20 #Salta para  
bge a2, t1, pula #pula se y >= 20  
mv a1, a2 #x = y  
pula:  
...
```

Agenda

- Sentenças condicionais
- **Laços de repetição**
- Chamada e retorno de funções
- Exemplos

Controle do fluxo de execução

Laço de repetição “enquanto” (`while`)

Exemplo: C/C++

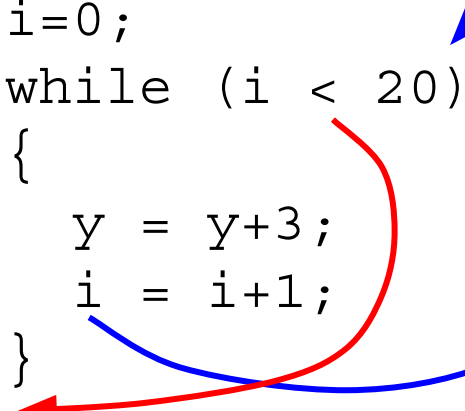
```
...  
i=0;  
while (i < 20)  
{  
    y = y+3;  
    i = i+1;  
}  
...
```

Controle do fluxo de execução

Laço de repetição “enquanto” (while)

Exemplo: C/C++

```
...  
i=0;  
while (i < 20)  
{  
    y = y+3;  
    i = i+1;  
}  
... i >= 20
```



Exemplo: Ling. de Montagem

```
# i está em a1  
# y está em a2
```


Controle do fluxo de execução

Laço de repetição “enquanto” (while)

Exemplo: C/C++

```
...  
i=0;  
while (i < 20)  
{  
    y = y+3;  
    i = i+1;  
}  
... i >= 20
```

Exemplo: Ling. de Montagem

```
# i está em a1  
# y está em a2  
...  
li    a1, 0           # i=0  
enquanto:  
li    t1, 20         # se i >= 20  
bge   a1, t1, cont   # sai do laço  
addi  a2, a2, 3      # y=y+3  
addi  a1, a1, 1      # i=i+1  
j     enquanto  
cont:  
... i >= 20
```

Controle do fluxo de execução

Laço de repetição “enquanto” (while)

Exemplo: C/C++

```
...  
i=0;  
while (i < 20)  
{  
    y = y+3;  
    i = i+1;  
}  
... i >= 20
```

The diagram illustrates the flow of a C/C++ while loop. A yellow box highlights the condition `i < 20`. A blue arrow loops from the end of the loop body back to the condition. A red arrow points from the condition to the text `i >= 20`, indicating the exit condition.

Exemplo: Ling. de Montagem

```
# i está em a1  
# y está em a2  
...  
li    a1, 0           # i=0  
enquanto:  
li    t1, 20          # se i >= 20  
bge   a1, t1, cont   # sai do laço  
addi  a2, a2, 3      # y=y+3  
addi  a1, a1, 1      # i=i+1  
j     enquanto  
cont:  
...  
... i >= 20
```

The diagram illustrates the flow of an assembly while loop. A yellow box highlights the branch instruction `bge a1, t1, cont`. A blue arrow loops from the `enquanto` label back to the branch instruction. A red arrow points from the branch instruction to the text `i >= 20`, indicating the exit condition.

Controle do fluxo de execução

Laço de repetição “para” (for)

Exemplo: C/C++

```
...  
for (i=0; i<10; i++)  
{  
    y = y+2;  
}  
...
```

Controle do fluxo de execução

Laço de repetição “para” (for)

Exemplo: C/C++

```
...  
for (i=0; i<10; i++)  
{  
    y = y+2;  
}  
...
```

Exemplo: Ling. de Montagem

```
# i está em a1  
# y está em a2
```

Controle do fluxo de execução

Laço de repetição “para” (for)

Exemplo: C/C++

```
...  
for (i=0; i<10; i++)  
{  
    y = y+2;  
}  
...
```

Exemplo: Ling. de Montagem

```
# i está em a1  
# y está em a2  
...  
li    a1, 0          # i=0  
for:  
li    t1, 10        # se i >= 10  
bge   a1, t1, cont  # sai do laço  
addi  a2, a2, 2     # y=y+2  
addi  a1, a1, 1     # i++  
j     for  
cont:  
...
```

Controle do fluxo de execução

Laço de repetição “para” (for)

Exemplo: C/C++

```
...  
for (i=0; i<10; i++)  
{  
    y = y+2;  
}  
...
```

Exemplo: Ling. de Montagem

```
# i está em a1  
# y está em a2  
...  
li    a1, 0 # i=0  
for:  
li    t1, 10 # se i >= 10  
bge   a1, t1, cont # sai do laço  
addi  a2, a2, 2 # y=y+2  
addi  a1, a1, 1 # i++  
j     for  
cont:  
...
```

Controle do fluxo de execução

Laço de repetição “faça-enquanto” (do-while)

Exemplo: C/C++

```
...  
i=0;  
do  
{  
    y = y+2;  
    i = i+1;  
} while (i < 10);  
...
```

Exemplo: Ling. de Montagem

```
# i está em a1  
# y está em a2  
...
```

=> Exercício

Controle do fluxo de execução

Laço de repetição “faça-enquanto” (do-while)

Exemplo: C/C++

```
...  
i=0;  
do  
{  
    y = y+2;  
    i = i+1;  
} while (i < 10);  
...
```

Exemplo: Ling. de Montagem

```
# i está em a1  
# y está em a2  
...  
li    a1, 0  
li    t1, 10  
do:  
    addi a2, a2, 2 # y=y+2  
    addi a1, a1, 1 # i=i+1  
    blt  a1, t1, do  
...  
...
```


Agenda

- Sentenças condicionais
- Laços de repetição
- **Chamada e retorno de funções**
- Exemplos

Controle do fluxo de execução

Chamando funções

Exemplo: C/C++

```
...  
i = soma(2, 3)  
...
```

Exemplo: Ling. de Montagem

```
# i está em a3  
# Soma:  
# parâmetros em a0 e a1  
# retorno em a0  
...  
li a0, 2  
li a1, 3  
jal soma  
mv a3, a0  
...
```

Controle do fluxo de execução

Retornando de funções

Exemplo: C/C++

```
int soma(int a, int b)
{
    return a+b;
}
```

Exemplo: Ling. de Montagem

```
# Soma:
#  parâmetros em a0 e a1
#  retorno em a0
...
soma:
    add a0, a0, a1
    ret
```

Controle do fluxo de execução

Retornando de funções

Exemplo: C/C++

```
int soma(int a, int b)
{
    return a+b;
}
```

Exemplo: Ling. de Montagem

```
# Soma:
# parâmetros em a0 e a1
# retorno em a0
...
soma:
    add a0, a0, a1
    ret
```

Pseudo-instrução `ret` é convertida pelo montador para:

```
jalr x0, ra, 0
```

Salta para o endereço `ra+0` e grava `PC+4` no registrador `x0` (ou seja, descarta `PC+4`)

Agenda

- Sentenças condicionais
- Laços de repetição
- Chamada e retorno de funções
- **Exemplos**

Exemplo 1

Traduza o seguinte programa para ling. de montagem RV32

```
/* Global array */
int numbers[10];

/* Returns the largest value from array numbers. */
int get_largest_number()
{
    int largest = numbers[0];
    for (int i=1; i<10; i++) {
        if (numbers[i] > largest)
            largest = numbers[i];
    }
    return largest;
}
```

Exemplo 1

Traduza o seguinte programa para ling. de montagem RV32

```
.data
numbers: .skip 40      # int numbers[10];
.text
get_largest_number:
    la  t1, numbers    # t1 <= &numbers[0]
    lw  a0, (t1)       # largest <= numbers[0]
    li  t2, 1          # i <= 1
    li  t3, 10
    li  t5, 4
for:
    bge t2, t3, cont   # Sai do laço se i >= 10
    mul t4, t2, t5     # t4 <= i*4
    add t4, t4, t1     # t4 <= &numbers[0] + i*4
    lw  t4, (t4)       # t4 <= numbers[i]
    ble t4, a0, end_if # Se t4 <= largest salta
    mv  a0, t4         # senão: largest <= t4
end_if:
    addi t2, t2, 1     # i <= i+1
    j  for
cont:
    ret
```

```
/* Global array */
int numbers[10];

/* Returns the largest value
from array numbers. */
int get_largest_number()
{
    int largest = numbers[0];
    for (int i=1; i<10; i++) {
        if (numbers[i] > largest)
            largest = numbers[i];
    }
    return largest;
}
```

Exemplo 2

Escreva uma função chamada `busca_caractere` que verifica se uma cadeia de caracteres terminada em zero possui um determinado caractere.

- Entrada:
 - `a0`: endereço inicial da cadeia
 - `a1`: caractere a ser procurado
- Retorna (em `a0`):
 - endereço da primeira posição da cadeia onde a letra ocorre; ou
 - o valor zero, caso não seja encontrado.

Exemplo 2

```
# Exemplo: Implementar uma função que verifica se uma cadeia
# de caracteres terminada em zero contém um determinado caractere.
# Entrada:
# a0: endereço inicial da cadeia
# a1: caractere a ser procurado
# Saída:
# a0: endereço da primeira posição que contém o caractere ou
# zero, caso não contenha.
busca_caractere:
    laco:
        lbu t1, 0(a0)          # Carrega o caractere atual
        beq t1, zero, nao_enc # Caso seja zero, retorna nao encontrou
        beq t1, a1, encontrou # Caso seja igual ao caractere de interesse
                                # retorne o endereço em a0
        addi a0, a0, 1        # Caso contrário, avança com o apontador a0
        j laco

    nao_enc:
        li a0, 0
    encontrou:
        ret
```

Exemplo 3

Escreva um trecho de programa que determina qual o maior valor de um vetor de números inteiros de 32 *bits* sem sinal cujo endereço inicial é dado em a_2 .

Inicialmente, a_3 contém o número de valores presentes na cadeia; suponha que o valor em a_3 é > 0 .

Ao final do trecho, a_0 deve conter o valor máximo e a_1 deve conter o endereço de memória onde se encontra o valor máximo.

Exemplo 3

```
# Exemplo: Maior valor de uma cadeia de números de 32 bits, sem sinal
# Entrada:
#   a2: endereço inicial da cadeia: base do vetor
#   a3: número de elementos na cadeia: N (a3>0)
# Saída:
#   a0: valor máximo
#   a1: endereço do elemento com valor máximo
```

```
#...
```

```
lw   a0, (a2)
mv   a1, a2
addi a3, a3, -1
```

```
laco:
```

```
ble  a3, zero, terminei
addi a2, a2, 4
lw   t1, (a2)
bleu t1, a0, pula_valor
mv   a0, t1
mv   a1, a2
```

```
pula_valor:
```

```
addi a3, a3, -1
j    laco
```

```
terminei:
```

```
#...
```