Execução condicional

O comando CMP

- Utilizado para comparar dois operandos (equal e not equal)
- CMP não altera o valor dos registradores fonte, destino

  Ex: ***cmp*** ah, 10
- Realiza uma subtração
- É utilizado em geral em conjunto com o comando j<cond>

Exercício

- Verifique no nasm o valor dos flags na comparação ***CMP al, dl.*** Justifique

AL	DL	CF	ZF	SF	OF	PF	AF
56	57	1	0	1	0	1	1
200	101	0	0	0	1	1	0
101	200	1	0	1	1	0	1
200	200	0	1	0	0	1	0
−105	−105	0	1	0	0	1	0
−125	−124	1	0	1	0	1	1
−124	−125	0	0	0	0	0	0

O comando j<cond>

- Com o jump condicional, a execução do programa é transferida para uma instrução alvo quando uma condição é satisfeita
- A sintaxe é: ***j<cond> label*** onde ***<cond>*** é a condição necessária para executar a instrução referenciada pelo ***label***

```
CMP al, 0dh      ; compara al com 0dh
Je teste         ; se igual, pula para teste

teste:
Mov al, bl
```

O comando j<cond>

- Como o processador lembra o resultado da instrução ***cmp*** quando vai executar a instrução ***je*** ?
- Utiliza o flag ***ZF***
- ***ZF=1*** se os operandos são iguais, senão ***ZF=0***
- Para realizar o ***jump*** o processador carrega no registrador ***IP*** o endereço da instrução alvo.

Comandos j<cond>

`je`	jump if equal
`jg`	jump if greater
`jl`	jump if less
`jge`	jump if greater or equal
`jle`	jump if less than or equal
`jne`	jump if not equal

`jz`	jump if zero (i.e., if ZF = 1)
`jnz`	jump if not zero (i.e., if ZF = 0)
`jc`	jump if carry (i.e., if CF = 1)
`jnc`	jump if not carry (i.e., if CF = 0)

Comandos j<cond>

Mnemonic	Meaning	condition tested
je jz	jump if equal jump if zero	ZF = 1
jne jnz	jump if not equal jump if not zero	ZF = 0
jg jnle	jump if greater jump if not less or equal	ZF = 0 and SF = OF
jge jnl	jump if greater or equal jump if not less	SF = OF
jl jnge	jump if less jump if not greater or equal	SF ≠ OF
jle jng	jump if less or equal jump if not greater	ZF = 1 or SF ≠ OF

Comandos j<cond>

- JO - Jump on Overflow OF=1
- JNO - Jump on No Overflow OF=0
- JC Jump on Carry CF=1
- JNC Jump on No Carry CF=0
- JS Jump on Sign (Negative) SF=1
- JNS Jump on No Sign (Positive) SF=0
- JZ Jump if Zero (same as JE) ZF=1
- JNZ Jump if Not Zero ZF=0
- JP Jump on parity
- Jnp Jump no parity

Ações a serem tomadas pelo jump

```
go_back:
        inc     AL
          .  .  .
          .  .  .
        cmp     AL,BL
        statement_1
        mov     BL,77H
```

statement_1	AL	BL	Action taken
`je go_back`	56H	56H	Program control is transferred to `inc AL`
`jg go_back`	56H	55H	Program control is transferred to `inc AL`
`jg go_back` `jl go_back`	56H	56H	No jump; executes `mov BL,77H`
`jle go_back` `jge go_back`	56H	56H	Program control is transferred to `inc AL`

`jne go_back`
`jg go_back`
`jge go_back`
	27H	26H	Program control is transferred to `inc AL`

Exemplo

- Lê caractere do teclado até que CR é digitado:

```
read_char:
        mov     DL,0
          . . .
        (code for reading a character into AL)
          . . .
        cmp     AL,0DH          ;compare the character to CR
        je      CR_received     ;if equal, jump to CR_received
        inc     CL              ;otherwise, increment CL and
        jmp     read_char       ;go back to read another
                                ;character from keyboard
CR_received:
        mov     DL,AL
          . . .
```

Comandos j<cond>

- Como os jumps condicionais sabem que um número é maior, menor, igual do que outro ?

```
cmp     num1,num2
```

$num1 = num2$
$num1 \neq num2$
$num1 > num2$
$num1 \geq num2$
$num1 < num2$
$num1 \leq num2$

- Para números sem sinal é suficiente utilizar os flags CF e ZF.

Comandos j<cond>

- Jumps para comparação sem sinal

Mnemonic	Meaning	condition tested
je jz	jump if equal jump if zero	ZF = 1
jne jnz	jump if not equal jump if not zero	ZF = 0
ja jnbe	jump if above jump if not below or equal	CF = 0 and ZF = 0
jae jnb	jump if above or equal jump if not below	CF = 0
jb jnae	jump if below jump if not above or equal	CF = 1
jbe jna	jump if below or equal jump if not above	CF = 1 or ZF = 1

Comandos j<cond>

- Para números com sinal os flags ZF, OF e SF definem as comparações relacionais.

```
cmp     Snum1,Snum2
```

Snum1	Snum2	ZF	OF	SF
56	55	0	0	0
56	−55	0	0	0
−55	−56	0	0	0
55	−75	0	1	1

Comando j<cond>

- Snum1 > Snum2 se a tabela abaixo registrar os seguintes valores para os flags:

Snum1	Snum2	ZF	OF	SF
56	55	0	0	0
56	−55	0	0	0
−55	−56	0	0	0
55	−75	0	1	1

ZF	OF	SF
0	0	0
or		
0	1	1

Comando j<cond>

- Snum1 < Snum2 se a tabela abaixo registrar os seguintes valores para os flags:

Snum1	Snum2	ZF	OF	SF
55	56	0	0	1
−55	56	0	0	1
−56	−55	0	0	1
−75	55	0	1	0

	ZF	OF	SF
	0	0	1
or			
	0	1	0

Comandos j<cond>

- Jumps para comparação com sinal:

Mnemonic	Meaning	condition tested
je jz	jump if equal jump if zero	ZF = 1
jne jnz	jump if not equal jump if not zero	ZF = 0
jg jnle	jump if greater jump if not less or equal	ZF = 0 and SF = OF
jge jnl	jump if greater or equal jump if not less	SF = OF
jl jnge	jump if less jump if not greater or equal	SF ≠ OF
jle jng	jump if less or equal jump if not greater	ZF = 1 or SF ≠ OF

O comando JMP

- Jmp é uma instrução incondicional que diz ao processador que a próxima instrução a ser executada está localizada em um rótulo.
- A sintaxe é : ***jmp label***

```
mov eax, 1

incremente:

inc eax
Jmp  incremente
mov ebx, eax
```

Exercicios

- Faça um programa em assembly que receba dois números e imprima qual é o maior e o menor
- Faça um programa em assembly que indique se um número é primo
- Faça um programa em assembly que indique no intervalo de 100 a 200 quais são os números pares

Exercícios

- Escreva um programa que mostre na tela os 256 caracteres do código ASCII.
- Escreva um programa que receba dois números

  entre 0 e 9 do teclado e apresente o maior deles.
- Escreva um programa que receba um número

  inteiro e retorne se o número é par ou ímpar

Exercício

- Faça um programa em C para exibir a tabuada de 0 a 9
- Faça um programa em C para gerar os n primeiros termos da seqüência:

  1 1 2 3 5 8 13 21
- Faça um programa que receba um número e calcule o seu fatorial (ex: 5 ! = 5 x 4 x 3 x 2 x 1)