

# Linguagem Assembly

Laços de Repetição (While)



## LAÇOS DE REPETIÇÃO - WHILE

Os laços de repetição em Assembly são combinações de lfs e Jumps. Para implementar um loop, teremos pelo menos dois rótulos: um para manter no loop, e outro para sair dele.

Ex.:

*while:*

#aqui vão os comandos que serão executados no loop

*saida:*

#aqui vão os comandos depois que o loop terminar

# LAÇOS DE REPETIÇÃO - WHILE

Ex.:

```
int i = 0;
while (i < 10){
    i++
}
print("%d", &i)
```



```
.text
move $t0, $zero

while:
    beq $t0, 10, saida
    addi $t0, $t0, 1
    j while
saida:
    #imprime o valor de i
    li $v0, 1
    move $a0, $t0
    syscall
```

Tabela - Tipos de dados básicos e representação

Linguagem C	Formato	Tipo de dados
<u>char</u>	%c	<u>caracter</u>
<u>int</u>	%d	<u>inteiro</u>
<u>float</u>	%f	<u>real</u>
<u>char[ ]</u>	%s	<u>cadeia de caracteres (string)</u>

## EXERCÍCIO

Escreva um programa que lê um número inteiro positivo e imprime todos os números inteiros de zero até o número lido.

Ex.:

Entrada: 7

Saída: 0 1 2 3 4 5 6 7

```

.data
    msg:.asciiz "Digite um número inteiro positivo: "
    espaco:.byte ' '

.text

    li $v0, 4
    la $a0,msg
    syscall

    li $v0,5 #leia inteiro
    syscall

    move $t0, $v0

    move $t1, $zero

laco:
    bgt $t1, $t0,sai_do_laco

    #imprime $t1
    li $v0, 1
    move $a0, $t1
    syscall

    li $v0, 4
    la $a0,espaco
    syscall

    addi $t1, $t1, 1

    j laco

sai_do_laco:

    li $v0, 10
    syscall

```

## Exercício 2

Faça um programa em Assembly, que solicite dois números inteiros ao utilizador e em seguida realize uma soma, caso os dois números sejam diferentes ou uma subtração, caso os dois números sejam iguais.

Entretanto, nenhum dos números digitados podem ser igual a zero.

No final o programa deve apresentar no ecrã uma frase, a dizer se somou ou subtraiu, acompanhada do resultado do cálculo.