```
#
# Codigo Assembly MIPS para um Hello World.
# Le um inteiro do teclado e verifica se nr < 0
# Se nr < 0 imprime a string Hello World --! seguida do nr digitado
# Se nr >=0 imprime a string Hello World ++! seguida do nr digitado
#
#
#include <stdio.h>
#
#main ()
#{
#   int vlr_int;
#
#   printf("Digite um valor:");
#   scanf("%d", vlr_int);
#
#   if( oper2 < 0)
#       printf("Hello World --! %d \n", vlr_int);
#   else
#       printf("Hello World ++! %d \n", vlr_int);
#
#   return;
#}
#
    .data       # inicia segmento de dados
    .align 0    # alinhamento a byte (2^0)
str_dig: .asciiz "Digite um numero: " # definindo a string str_dig
str_pos: .asciiz "\nHello World ++! " # definindo a string str_pos
str_neg: .asciiz "\nHello World --! " # definindo a string str_neg

                    # estes dois dados abaixo não são necessarios no programa
#   .align 2        # exemplifica alinhamento para inteiros com 32 bits
#vlr_int: .word 5   # exemplifica armazenamento de um inteiro na memoria de maneira estática
#string: .asciiz "ABCDEFGHI" # armazena estaticamente a cadeia rotulada como string na memoria

    .text               # inicia segmento de texto (instrucoes)
    .align 2            # alinhamento a word 4 bytes (2^2)
    .globl main         # rotulo main e global

main:                   # inicio do programa

    # vai imprimir uma string
    # load immediate
    li $v0, 4           # atribui 4 para $v0. Codigo para print_str
    # load address
    la $a0, str_dig     # carrega endereco de str_dig em $a0
                        # eh o end da string a ser impressa
    syscall             # chamada de sistema para E/S

    # vai ler um inteiro vindo do teclado
    li $v0, 5           # atribui 5 para $vo. Codigo para read_int
    syscall             # chamada de sistema para E/S. Retorno estara em $v0
    move $t2, $v0       # copia conteudo digitado para $t2 para preservar dado

    li $t1, 1           # $t1=1 Eh usado como aux.
```

```
56
57      # set-on-less-than
58      slt $t0, $t2, $zero # verifica se nr digitado eh negativo
59
60      # branch-if-equal
61      beq $t0, $t1, print_neg # se é negativo então goto print_neg
62
63      # vai imprimir uma string
64      li $v0, 4           # atribui 4 para $v0. Codigo para print_str
65      la $a0, str_pos     # carrega endereco de str_pos em $a0
66                          # eh o end da string a ser impressa
67      syscall             # chamada de sistema para E/S
68
69      # vai imprimir um inteiro se ele for >= 0
70      li $v0, 1           # atribui 1 para $v0. Codigo para print_int
71      move $a0, $t2       # copia nr a ser impresso para $a0.
72                          # argumento de entrada para a impressao do int
73      syscall             # chamada de sistema para E/S
74
75      # jump
76      j thatsallfolks     # goto thatsallfolks. Salta a impressao do nr negativo
77
78  print_neg:              # aqui fara impressao no nr negativo
79
80      # vai imprimir uma string
81      li $v0, 4           # atribui 4 para $v0. Codigo para print_str
82      la $a0, str_neg     # carrega endereco de str_neg em $a0
83                          # eh o end da string a ser impressa
84      syscall             # chamada de sistema para E/S
85
86      # vai imprimir um inteiro
87      li $v0, 1           # atribui 1 para $v0. Codigo para print_int
88      move $a0, $t2       # copia nr a ser impresso para $a0.
89                          # argumento de entrada para a impressao do int
90      syscall             # chamada de sistema para E/S
91
92  thatsallfolks:          # o que que ha velhinho?
93      li $v0, 10          # atribui 10 para $v0. Codigo para exit (termina programa)
94      syscall             # chamada de sistema para E/S
95
```