ATIVIDADE AVALIATIVA

Instruções - 4

Estudante:

Estudante:

Obs.: a) Cole no texto de solução de cada exercício *print* de tela do simulador MARS (ou DrMIPS) com o código em assembly, tabela de registradores (abra Registers) e tabela de memória RAM (Data Segment). Use-o para buscar erros de sintaxe.

b) Introduza valores nos registradores e memória (manualmente, diretamente nas tabelas) para verificar o funcionamento do código assembly. Por padrão, há somente zero em todas as posições no início da execução.

c) Para carregar (“atribuir”) valor a um registrador via código assembly, é possível usar a instrução addi. Ex.: addi $t0, $zero, 2 (o efeito é $t0=2). Isso ajuda a verificar o funcionamento do código.

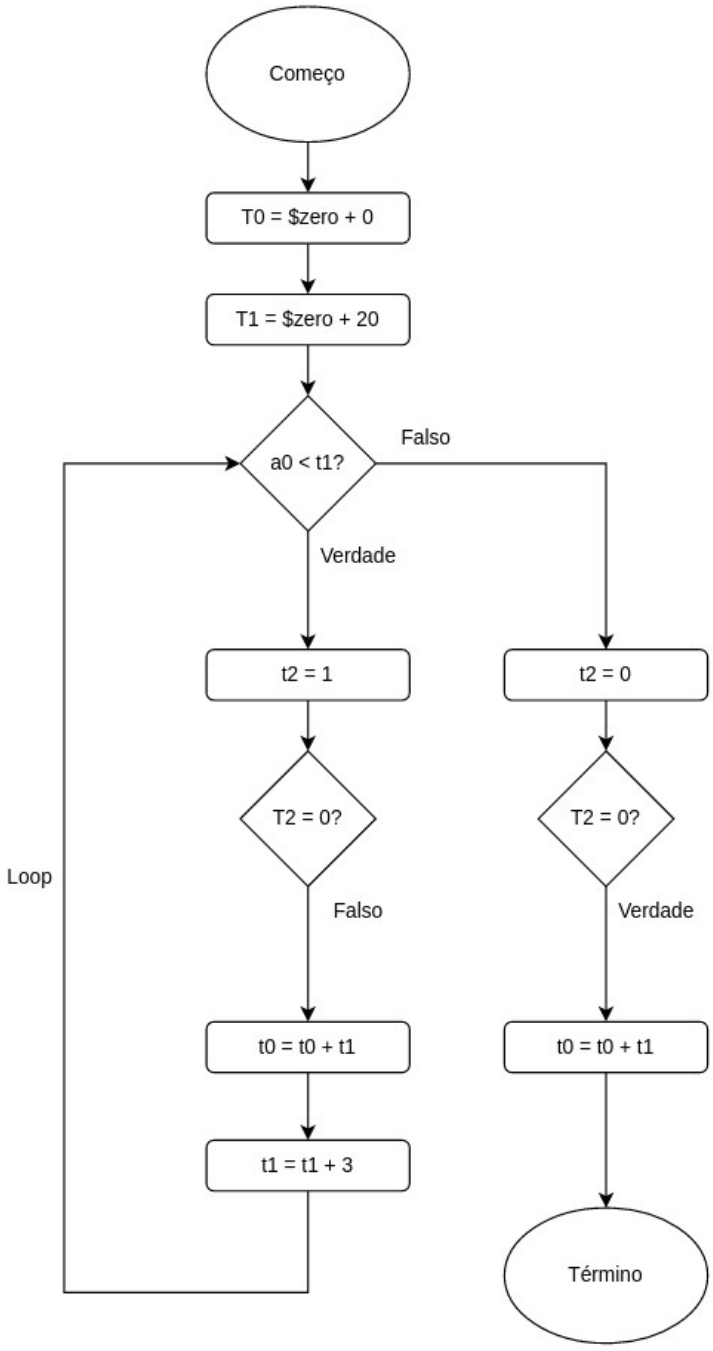
d) Caso não for possível visualizar o valor na RAM (Data Segment), altere o offset (load, store) para um número negativo (-10, -100, -1000, -10000 etc) até ele constar na tela. Escreva um comentário (#) no código assembly para avisar que foi necessário ajustar o offset. Essa problemática de *range* de endereços de memória depende da versão do MARS; basta avisar esse problema em comentário no código assembly que o/a estudante não perderá pontos na correção.

Considere o seguinte código em assembly do MIPS. Assuma que $a0 é usado para a entrada, e inicialmente contém **n**, um inteiro positivo. Suponha que $v0 é usado para a saída.

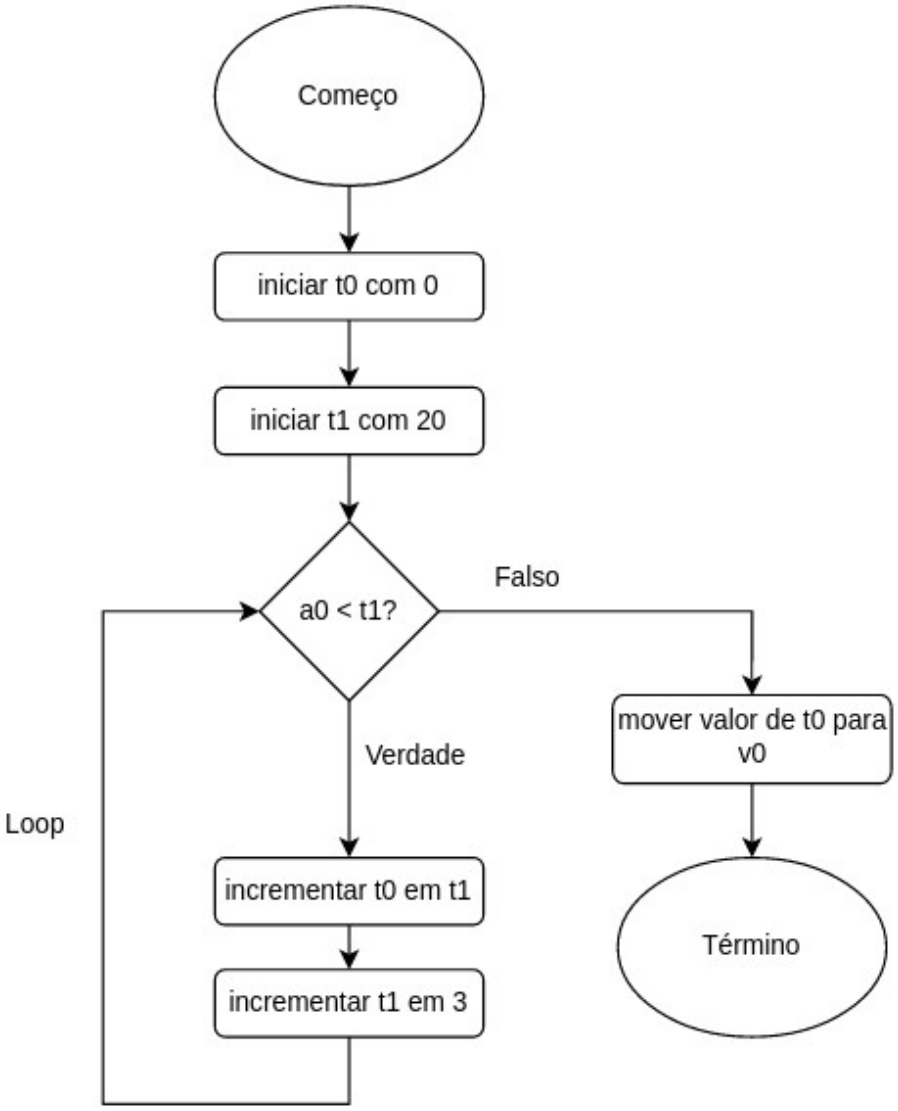
```
begin:  addi $t0, $zero, 0
        addi $t1, $zero, 20
loop:   slt $t2, $a0, $t1
```

```
            beq $t2, $zero, finish
            add $t0, $t0, $t1
            addi $t1, $t1, 3
            j loop
finish:     add $v0, $t0, $zero
```

a) Desenhe um fluxograma que represente o algoritmo implementado *em baixo nível* por esse código em MIPS (proceda da maneira tradicional, empregando losangos, retângulos, setas etc).



b) Desenhe um fluxograma que represente um algoritmo *em alto nível* que poderia ter sido a origem dessa implementação em código MIPS (proceda da maneira tradicional, empregando losangos, retângulos, setas etc).



c) Escreva esse algoritmo em linguagem C (em consonância com o algoritmo em *alto nível*).

Use a seguinte correspondência entre registradores MIPS e variáveis em C (itens *b* e *c*):

$t0, $t1, $t2, $a0, $v0 : V, W, X, Y, Z.

```
1 v = 0;
2 for( w = 20; x < w; w += 3 ) {
3    v += w;
4 }
5 z = v;
```