[C208] Atividade - Revisão

**Questão 1 -** Analise as afirmativas a seguir e as classifique com verdadeiras ou falsas. Além disto, explique o porquê daquelas que classificadas como falsas. Não é necessário explicar as verdadeiras.

(F) É possível afirmar que os compiladores são usados para gerar o programa executável a partir do código objeto.

R.: Compiladores transformam o codigo fonte em codigo objeto.

(F) A compilação cruzada é realizada somente quando se tem sistemas operacionais iguais, mas em versões diferentes.

R.: Podem ser realizadas em SO diferentes também.

(F) O código objeto se difere do código executável apenas pela etapa de montagem.

R.: Se difere pelo linker

(F) Os programas executáveis gerados em dois computadores idênticos com sistemas operacionais diferentes, a partir do mesmo código fonte em baixo nível, serão sempre iguais.

R.: Serão diferentes devido ao SO.

**Questão 2 -** Considere a seguinte parte de programa em linguagem Assembly do MIPS:

Texto

Descrição gerada automaticamente

Complete o quadro abaixo considerando as variáveis declaradas no código acima. (Não há necessidade de se utilizar hexadecimal para representar a palavra “CAFE”). Ométodode armazenamento é Big Endian.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MEMÓRIA DE DADOS** | | | | |
| **Endereço** | **Dado** |  | **Endereço** | **Dado** |
| 0x10010000 |  | 0x10010008 | 0x00 |
| 0x10010001 |  | 0x10010009 | 0x00 |
| 0x10010002 |  | 0x1001000A | 0x00 |
| 0x10010003 |  | 0x1001000B | 0x15 |
| 0x10010004 | 0x00 | 0x1001000C | C |
| 0x10010005 | 0x0D | 0x1001000D | A |
| 0x10010006 | 0x00 | 0x1001000E | F |
| 0x10010007 | 0x00 | 0x1001000F | E |

**Questão 3:** Converta a seguinte instrução em assembly MIPS. Considere a = $s0, b = $s1, c =

$s2, d = $s4. Tente reutilizar os registradores temporários.

**Correspondente em assembly MIPS:**

**.text**

**Lw $t1, 16($s0)**

**Add $t1, $t1, $s4**

**Sub $t1, $t1, $s2**

**Sw $t1, 20($s2)**

**Instrução:**

**c[5] = a[4] + d – c**

**Questão 4 :** Dado o estado atual dos registradores e memória de dados:

|  |  |
| --- | --- |
| **REGISTRADOR** | |
| **ENDEREÇO** | **DADO** |
| $t0 | 0x10010001 |
| $t1 | 0x10010004 |
| $t2 | 0x00000030 |
| $t3 | 0x00000040 |
| $t4 | 0x00000000 |
| $t5 | 0x00000000 |
| $t6 | 0xABCDEF00 |
| $t7 | 0x00000000 |
| $s0 | 0x00000000 |
| $s1 | 0x00000050 |
| $s2 | 0x00000060 |
| $s3 | 0x00000000 |

|  |  |
| --- | --- |
| **MEMÓRIA** | |
| **ENDEREÇO** | **DADO** |
| 0x10010000 | 0xAA |
| 0x10010001 | 0x1C |
| 0x10010002 | 0x53 |
| 0x10010003 | 0x28 |
| 0x10010004 | 0x84 |
| 0x10010005 | 0xF1 |
| 0x10010006 | 0x12 |
| 0x10010007 | 0x64 |
| 0x10010008 | 0xE5 |
| 0x10010009 | 0x87 |
| 0x1001000A | 0x99 |
| 0x1001000B | 0x3D |

Mostre os efeitos na memória e nos registradores das seguintes instruções, considerando que cada uma é executada independentemente das outras.

# a) lh $t2, 4($t1)

|  |  |
| --- | --- |
| **REGISTRADOR** | |
| **ENDEREÇO** | **DADO** |
| $t0 |  |
| $t1 |  |
| $t2 | 0x000087E5 |
| $t3 |  |
| $t4 |  |
| $t5 |  |
| $t6 |  |
| $t7 |  |
| $s0 |  |
| $s1 |  |
| $s2 |  |
| $s3 |  |

|  |  |
| --- | --- |
| **MEMÓRIA** | |
| **ENDEREÇO** | **DADO** |
| 0x10010000 |  |
| 0x10010001 |  |
| 0x10010002 |  |
| 0x10010003 |  |
| 0x10010004 |  |
| 0x10010005 |  |
| 0x10010006 |  |
| 0x10010007 |  |
| 0x10010008 |  |
| 0x10010009 |  |
| 0x1001000A |  |
| 0x1001000B |  |

**b) sb $t3, 8($t0)**

|  |  |
| --- | --- |
| **REGISTRADOR** | |
| **ENDEREÇO** | **DADO** |
| $t0 |  |
| $t1 |  |
| $t2 |  |
| $t3 |  |
| $t4 |  |
| $t5 |  |
| $t6 |  |
| $t7 |  |
| $s0 |  |
| $s1 |  |
| $s2 |  |
| $s3 |  |

|  |  |
| --- | --- |
| **MEMÓRIA** | |
| **ENDEREÇO** | **DADO** |
| 0x10010000 |  |
| 0x10010001 |  |
| 0x10010002 |  |
| 0x10010003 |  |
| 0x10010004 |  |
| 0x10010005 |  |
| 0x10010006 |  |
| 0x10010007 |  |
| 0x10010008 |  |
| 0x10010009 | 0x40 |
| 0x1001000A |  |
| 0x1001000B |  |

**Questão 5**: Escreva um programa em Assembly MIPS que faça a leitura de três valores numéricos inteiros fornecidos pelo usuário e exiba uma mensagem informando se a soma destes é maior, menor ou igual a 100.

.data

f1: .asciiz "Entre com o primeiro valor: "

f2: .asciiz "Entre com o segundo valor: "

f3: .asciiz "Entre com o terceiro valor: "

f\_maior: .asciiz "\n É maior que 100"

f\_menor: .asciiz "\n É menor que 100"

f\_igual: .asciiz "\n É igual a 100"

.text

#Primeiro valor

la $a0, f1 # carrega a frase

li $v0, 4 #mostra a string

syscall

li $v0, 5 #recebe o valor

syscall

add $t1, $v0, 0 #add o valor a t1

#Segundo valor

la $a0, f1 # carrega a frase

li $v0, 4 #mostra a string

syscall

li $v0, 5 #recebe o valor

syscall

add $t2, $v0, 0 #add o valor a t2

#Terceiro valor

la $a0, f1 # carrega a frase

li $v0, 4 #mostra a string

syscall

li $v0, 5 #recebe o valor

syscall

add $t3, $v0, 0 #add o valor a t3

#processamento

add $t1, $t1, $t2

add $t1, $t1, $t3

blt $t1,100, menor

bgt $t1,100, maior

la $a0, f\_igual

li $v0, 4

syscall

j fim

maior:

la $a0, f\_maior

li $v0, 4

syscall

j fim

menor:

la $a0, f\_menor

li $v0, 4

syscall

j fim

fim:

li $v0, 10

|  |  |  |  |
| --- | --- | --- | --- |
| **Senice** | Code |  | Result |
| print integer | I | $a0 = integer to print |  |
| print flat | 2 | gft2 = flat to print |  |
| print double | 3 | gft2 = double to print |  |
|  | 4 | $a0 = address ofnull-teuninated |  |
| ixdmeg‹x |  |  | $v0 contains integer read |
| eadQoW | 6 |  | $f0 contains Ooat read |
| ixddouble | 7 |  | $f0 contains double read |
|  | 8 | $a0 = address of input buffer  $a1 = maximum number of characters to read | M role h‹z/ou- /ofJe |
| * \*' h p   « | 9 | - number of bytes to allocate | $v0 contains address of allocated memory |
| exit  **execubon)** | 10 |  |  |

syscall