

## **Exercício Prático 06 - AC-II**

### **Parte I**

**1. O que é um arquivo fonte?**

- A. um arquivo de texto que contém instruções de linguagem de programação.**
- B. um subdiretório que contém os programas.**
- C. um arquivo que contém dados para um programa.**
- D. um documento que contém os requisitos para um projeto.**

**R.: Letra A**

**2. O que é um registrador?**

- A. parte do sistema de computador que mantém o controle dos parâmetros do sistema.**
- B. uma parte do processador que possui um padrão de bits.**
- C. parte do processador que contém o seu número de série único.**
- D. parte do bus de sistema que contém dados.**

**R.: Letra B**

**3. Qual o caracter que, na linguagem assembly do SPIM, inicia um comentário?**

- A. #**
- B. \$**
- C. //**
- D. \***

**R.: Letra A**

**4. Quantos bits há em cada instrução de máquina MIPS?**

- A. 8**
- B. 16**
- C. 32**
- D. instruções diferentes possuem diferentes comprimentos.**

**R.: Letra C**

**5. O que é o contador de programa?**

- A. um registrador que mantém a conta do número de erros durante a execução de um programa.**
- B. uma parte do processador que contém o endereço da primeira palavra de dados.**
- C. uma variável na montadora que os números das linhas do arquivo de origem.**
- D. parte do processador que contém o endereço da próxima instrução de máquina para ser obtida.**

**R.: Letra D**

**6. Ao executarmos uma instrução, quanto será adicionado ao contador de programa?**

- A. 1**
- B. 2**
- C. 4**
- D. 8**

**R.: Letra C**

**7. O que é uma diretiva, tal como a diretiva .text?**

- A. uma instrução em linguagem assembly que resulta em uma instrução em linguagem de máquina.**
- B. uma das opções de menu do sistema SPIM.**
- C. uma instrução em linguagem de máquina que faz com que uma operação sobre os dados ocorra.**
- D. uma declaração que diz o montador algo sobre o que o programador quer, mas não corresponde diretamente a uma instrução de máquina.**

**R.: Letra D**

**8. O que é um endereço simbólico?**

- A. um local de memória que contém dados simbólicos.**
- B. um byte na memória que contém o endereço de dados.**
- C. símbolo dado como argumento para uma directiva.**
- D. um nome usado no código-fonte em linguagem assembly para um local na memória.**

**R.: Letra D**

**9. Em qual endereço o simulador SPIM coloca a primeira instrução de máquina quando ele está sendo executado?**

- A. 0x00000000**
- B. 0x00400000**
- C. 0x10000000**
- D. 0xFFFFFFFF**

**R.: Letra B**

**10. Algumas instruções de máquina possuem uma constante como um dos operandos. Como é chamado tal operando?**

- A. operando imediato**
- B. operando embutido**
- C. operando binário**
- D. operando de máquina**

**R.: Letra A**

**11. Como é chamada uma operação lógica executada entre bits de cada coluna dos operandos para produzir um bit de resultado para cada coluna?**

- A. operação lógica**
- B. operação bitwise**
- C. operação binária**
- D. operação coluna**

**R.: Letra B**

**12. Quando uma operação é de fato executada, como estão os operandos na ALU?**

- A. Pelo menos um operando deve ser de 32 bits.**
- B. Cada operando pode ser de qualquer tamanho.**
- C. Ambos os operandos devem vir de registros.**
- D. Cada um dos registradores deve possuir 32 bit.**

**R.: Letra D**

**13. Dezesesseis bits de dados de uma instrução de ori são usados como um operando imediato. Durante a execução, o que deve ser feito primeiro?**

- A. Os dados são estendidos em zero à direita por 16 bits.**
- B. Os dados são estendidos em zero à esquerda por 16 bits.**
- C. Nada precisa ser feito.**
- D. Apenas 16 bits são usados pelo outro operando.**

**R.: Letra B**

**14. Qual das instruções seguintes armazenam no registrador \$5 um padrão de bits que representa positivo 48?**

- A. ori \$5,\$0,0x48**
- B. ori \$5,\$5,0x48**
- C. ori \$5,\$0,48**
- D. ori \$0,\$5,0x48**

**R.: Letra C**

**15. A instrução de ori pode armazenar o complemento de dois de um número em um registrador?**

- A. Não.**
- B. Sim.**

**R.: Letra A**

**16. Qual das instruções seguintes limpa todos os bits no registrador \$8 com exceção do byte de baixa ordem que fica inalterado?**

- A. ori \$8,\$8,0xFF
- B. ori \$8,\$0,0x00FF
- C. xori \$8,\$8,0xFF
- D. andi \$8,\$8,0xFF

R.: Letra D

17. Qual é o resultado de um ou exclusivo de padrão sobre ele mesmo?

- A. Todos os bits em zero.
- B. Todos os bits em um.
- C. O padrão original utilizado.
- D. O resultado é o contrário do original.

R.: Letra A

18. Todas as instruções de máquina têm os mesmos campos?

- A. Não. Diferentes de instruções de máquina possuem campos diferentes.
- B. Não. Cada instrução de máquina é completamente diferente de qualquer outra.
- C. Sim. Todas as instruções de máquina têm os mesmos campos na mesma ordem.
- D. Sim. Todas as instruções de máquina têm os mesmos campos, mas eles podem estar em ordens diferentes.

R.: Letra A

## Parte II

//programa 1 (add, addi, sub, lógicas)

```
{  
    a = 2;  
    b = 3;  
    c = 4;  
    d = 5;  
    x = (a+b) - (c+d);  
    y = a - b + x;  
    b = x - y;  
}
```

```

1 # Exercício 1
2
3 #inicio
4
5 .text
6 .globl main
7
8 main:
9
10 # Atribuições
11 addi $s0, $zero, 2 # a = 2
12 addi $s1, $zero, 3 # b = 3
13 addi $s2, $zero, 4 # c = 4
14 addi $s3, $zero, 5 # d = 5
15
16 # x = (a + b) - (c + d)
17 add $t0, $s0, $s1 # $t0 = a + b
18 add $t1, $s2, $s3 # $t1 = c + d
19 sub $s4, $t0, $t1 # $s4 = x = (a + b) - (c + d)
20
21 # y = a - b + x
22 sub $t2, $s0, $s1 # $t2 = a - b
23 add $s5, $t2, $s4 # $s5 = y = (a - b) + x
24
25 # b = x - y
26 sub $s1, $s4, $s5 # $s1 = b = x - y
27
28 #fim
29

```

0x10010000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

0x00000000

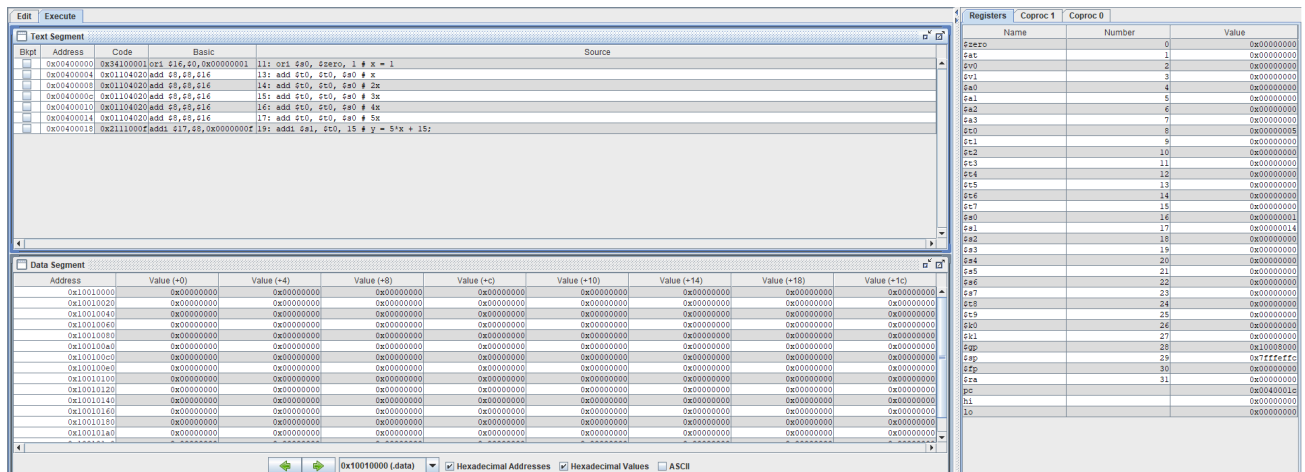
0x00000000

0x00000000

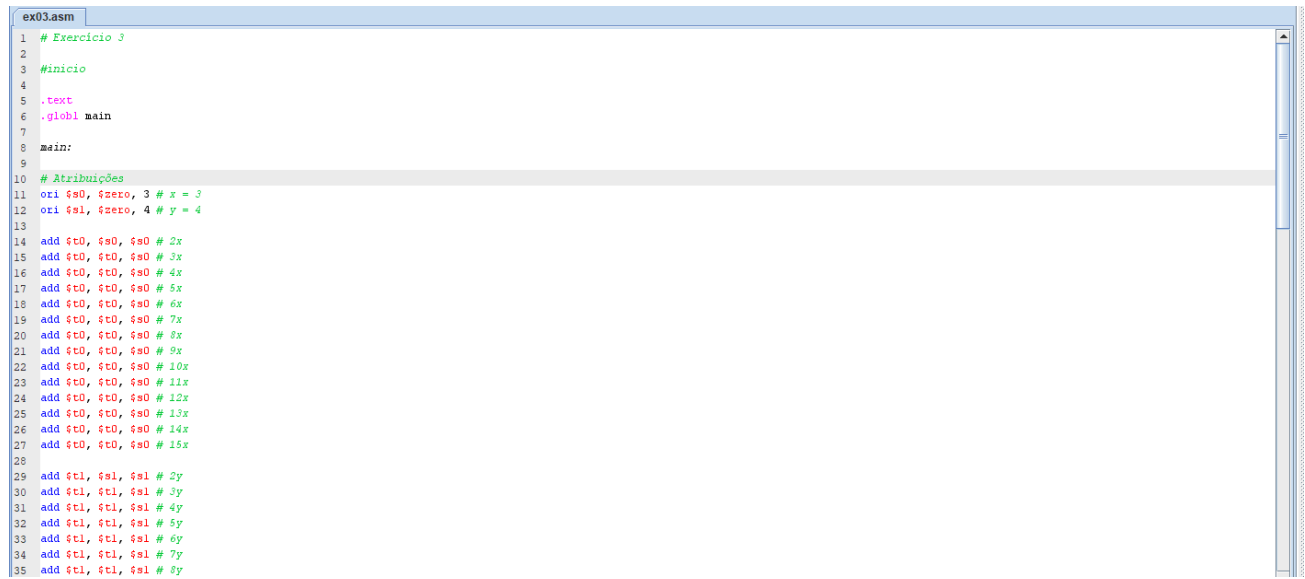
0x00000000

0x00000000

0x00000000



```
// programa 3 (add, addi, sub, lógicas)
{
    x = 3;
    y = 4 ;
    z = ( 15*x + 67*y)*4
}
```



```

69 add $t1, $t1, $s1 # 42y
70 add $t1, $t1, $s1 # 43y
71 add $t1, $t1, $s1 # 44y
72 add $t1, $t1, $s1 # 45y
73 add $t1, $t1, $s1 # 46y
74 add $t1, $t1, $s1 # 47y
75 add $t1, $t1, $s1 # 48y
76 add $t1, $t1, $s1 # 49y
77 add $t1, $t1, $s1 # 50y
78 add $t1, $t1, $s1 # 51y
79 add $t1, $t1, $s1 # 52y
80 add $t1, $t1, $s1 # 53y
81 add $t1, $t1, $s1 # 54y
82 add $t1, $t1, $s1 # 55y
83 add $t1, $t1, $s1 # 56y
84 add $t1, $t1, $s1 # 57y
85 add $t1, $t1, $s1 # 58y
86 add $t1, $t1, $s1 # 59y
87 add $t1, $t1, $s1 # 60y
88 add $t1, $t1, $s1 # 61y
89 add $t1, $t1, $s1 # 62y
90 add $t1, $t1, $s1 # 63y
91 add $t1, $t1, $s1 # 64y
92 add $t1, $t1, $s1 # 65y
93 add $t1, $t1, $s1 # 66y
94 add $t1, $t1, $s1 # 67y
95
96 add $t3, $t0, $t1 # j = 15*x + 67*y
97 addi $t3, $t3, 4 # 4*j
98
99 #fin

```

**Nos exercícios a seguir procure usar as inst. sll, srl e sra:**

```
// programa 4
{
    x = 3;
    y = 4 ;
    z = ( 15*x + 67*y)*4
```

```

1  # Exercício 4
2
3  #inicio
4
5  .text
6  .globl main
7
8  main:
9
10 # Atribuições
11 ori $s0, $zero, 3 # x = 3
12 ori $s1, $zero, 4 # y = 4
13
14 sll $t0, $s0, 4 # t0 = 16
15 sub $t0, $t0, $s0 # t0 = 15x
16
17 sll $t1, $s1, 6 #t1 = 64y
18 sll $t2, $s1, 1 #t1 = 2y
19 add $t2, $s1, 1 #t2 = 3y
20 add $t1, $t1, $t2 #t1 = 67y
21
22 add $t0, $t0, $t1 #t0 = 15x+67y
23 sll $s2, $t0, 2 # z = 4 * t0
24
25
26 #fim
27
```

```
// programa 5
```

```

1  # Exercício 5
2
3  #inicio
4
5  .text
6  .globl main
7
8  main:
9
10 # Atribuições
11
12 # x = 100000:
13 # y = 200000:
14 # z = x + y:
15
16 ori %t0, %zero, 0x186A
17 sll %s0, %t0, 4 # 6250 * 16(2^4) = 100000
18
19 ori %t1, %zero, 0x30D4
20 sll %s1, %t1, 4 # 12500 * 16(2^4) = 200000
21
22 add %s2, %s0, %s1 # x + y
23
24 #fim
25

```



Assembly code editor showing assembly instructions and registers.

**Text Segment**

Blkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x3408186a	ori \$8,\$0,0x0000186a	16: ori \$t0, \$zero, 0x186A
<input type="checkbox"/>	0x00400004	0x00081100	sl \$1,\$t0,\$t0,4 # 4250 * 16(2^4) = 100000	17: sl \$t0, \$t0, 4 # 4250 * 16(2^4) = 100000
<input type="checkbox"/>	0x00400008	0x34095004	ori \$9,\$0,0x00005004	19: ori \$t1, \$zero, 0x5004
<input type="checkbox"/>	0x0040000c	0x00095900	sl \$1,\$t1,\$t1,4 # 12500 * 16(2^4) = 200000	20: sl \$t1, \$t1, 4 # 12500 * 16(2^4) = 200000
<input type="checkbox"/>	0x00400010	0x02119020	add \$16,\$t0,\$t1	22: add \$s2, \$s0, \$t1 # x + y

**Data Segment**

Address	Value (+0)	Value (-4)	Value (+8)	Value (-C)	Value (+10)	Value (+14)	Value (-18)	Value (+1C)
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100A0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100C0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100E0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010140	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010160	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010180	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100101A0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

**Registers**

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x0000186A
\$t1	9	0x00003044
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x0000186A
\$s1	17	0x00003044
\$s2	18	0x000493e0
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$s8	24	0x00000000
\$s9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x00000000
\$fp	29	0x7ffffcfc
\$tp	30	0x00000000
\$ra	31	0x00000000
\$pc		0x00400014
\$hi		0x00000000
\$lo		0x00000000

```
// programa 6
{
    x = o maior inteiro possível;
    y = 300000;
    z = x - 4y
}
```

Assembly code editor showing assembly instructions and registers.

**Text Segment**

Blkpt	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x340872ff	ori \$8,\$0,0x000072ff	12: ori \$t0, \$zero, 0x7FFF
<input type="checkbox"/>	0x00400004	0x00084400	sl \$1,\$t0,\$t0,16	13: sl \$t0, \$t0, 16
<input type="checkbox"/>	0x00400008	0x35102fff	ori \$16,\$0,0x00002fff	14: ori \$t0, \$t0, 0xFFFF # \$t0 = maior inteiro possível
<input type="checkbox"/>	0x0040000c	0x3409493e	ori \$9,\$0,0x0000493e	16: ori \$t1, \$zero, 0x493E
<input type="checkbox"/>	0x00400010	0x00095900	sl \$1,\$t1,\$t1,4 # \$t1 = 300 000	17: sl \$t1, \$t1, 4 # \$t1 = 300 000
<input type="checkbox"/>	0x00400014	0x00115080	sl \$10,\$t1,2 # \$t2 = 1 200 000	19: sl \$t2, \$t1, 2 # \$t2 = 1 200 000
<input type="checkbox"/>	0x00400018	0x020a5022	sub \$16,\$t0,\$t2	20: sub \$s2, \$s0, \$t2

**Data Segment**

Address	Value (+0)	Value (-4)	Value (+8)	Value (-C)	Value (+10)	Value (+14)	Value (-18)	Value (+1C)
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100A0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100C0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100E0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010140	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010160	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010180	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100101A0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

**Registers**

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x7ffffcfc
\$t1	9	0x000493e0
\$t2	10	0x00124930
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x7ffffcfc
\$s1	17	0x000493e0
\$s2	18	0x7fede07e
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$s8	24	0x00000000
\$s9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x00000000
\$fp	29	0x7ffffcfc
\$tp	30	0x00000000
\$ra	31	0x00000000
\$pc		0x0040001c
\$hi		0x00000000
\$lo		0x00000000

```
// programa 7
Considere a seguinte instrução iniciando um programa:
ori $8, $0, 0x01
```

Usando apenas instruções reg-reg lógicas e/ou instruções de deslocamento (sll, srl e sra), continuar o programa de forma que ao final, tenhamos o seguinte conteúdo no registrador \$8:

**\$8 = 0xFFFFFFFF**

```

1  # Exercício 7
2
3  #inicio
4
5  .text
6  .globl main
7
8  main:
9
10 # Atribuições
11
12 ori $8, $0, 0x01
13
14 sll $t0, $t0, 31
15 sra $t0, $t0, 31
16
17
18 #fim

```

The screenshot displays a MIPS simulator interface with three main panels:

- Text Segment:** Shows assembly instructions with their addresses and binary/hex values.
 

Expr	Address	Code	Basic	Source
<input type="checkbox"/>	0x00400000	0x34800001	ori \$8,\$0,0x00000001	12: ori \$8, \$0, 0x01
<input type="checkbox"/>	0x00400004	0x00847c0d	sll \$t0,\$t0,0x0000001f	14: sll \$t0, \$t0, 31
<input type="checkbox"/>	0x00400008	0x00847c0d	sra \$t0,\$t0,0x0000001f	15: sra \$t0, \$t0, 31
- Registers:** A table showing the state of MIPS registers.
 

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0xFFFFFFFF
\$t1	9	0x00000000
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0x00000000
\$s2	18	0x00000000
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$s8	24	0x00000000
\$s9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x00000000
\$t9	29	0xFFFFFFFF
\$fp	30	0x00000000
\$ra	31	0x00000000
\$l0		0x00000000
- Data Segment:** A memory dump showing addresses and values for the first 16 bytes.
 

Address	Value (+0)	Value (+4)	Value (+8)	Value (+C)	Value (+10)	Value (+14)	Value (+18)	Value (+1C)
0x10010000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100A0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100C0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100100E0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010140	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010160	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10010180	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x100101A0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

// programa 8

Inicialmente escreva um programa que faça:

**\$8 = 0x12345678.**

A partir do registrador \$8 acima, usando apenas instruções lógicas (or, ori, and, andi, xor, xori) e instruções de deslocamento (sll, srl e sra), você deverá obter os seguintes valores nos respectivos registradores:

**\$9 = 0x12**

**\$10 = 0x34**

**\$11 = 0x56**

**\$12 = 0x78**

Para os programas a seguir use instruções de Memória (lw e sw)

```

1  # Exercício 9
2
3  #inicio
4
5  .text
6  .globl main
7
8  main:
9
10 # Atribuições
11 ori $0, $zero, 0x1234
12 sll $0, $0, 16
13 ori $0, $0, 0x5678
14 # t0 = 0x12345678
15
16 srl $9, $0, 24
17 # t1 = 0x00000012
18
19 sll $10, $0, 8
20 srl $10, $10, 24
21 # t2 = 0x00000034
22
23 sll $11, $0, 16
24 srl $11, $11, 24
25 # t3 = 0x00000034
26
27 sll $12, $0, 24
28 srl $12, $12, 24
29 # t4 = 0x00000078
30
31 #fim

```

The screenshot displays the GNU Binutils linker interface. The main window shows assembly code with columns for Address, Code, Basic, and Source. The code includes instructions like `ori $0, $zero, 0x1234` and `sll $0, $0, 16`. A 'Labels' window on the right shows the 'main' label at address 0x00400000. On the far right, a 'Registers' window lists registers from \$zero to \$31, with their current values in hexadecimal.

// programa 9  
 Considere a memória inicial da seguinte forma:  
 .text

.data  
 x1: .word 15  
 x2: .word 25  
 x3: .word 13  
 x4: .word 17  
 soma: .word -1

Escrever um programa que leia todos os números, calcule e substitua o valor da variável soma por este valor.

```

ex09.asm
1  # Exercício 9
2
3  #inicio
4
5  .text
6  .globl main
7
8  main:
9
10 # Atribuições
11 .data
12 x1: .word 15
13 x2: .word 25
14 x3: .word 13
15 x4: .word 17
16 soma: .word -1
17
18 .text
19 lui $t0, 0x1001
20
21 lw $s0, 0($t0)
22 lw $s1, 4($t0)
23 lw $s2, 8($t0)
24 lw $s3, 12($t0)
25
26 add $t1, $s0, $s1
27 add $t1, $t1, $s2
28 add $t1, $t1, $s3
29
30 sw $t1, 16($t0)
31
32 #fim

```

The screenshot shows the MARS MIPS simulator interface. The main window displays the assembly code from the previous block. The 'Labels' window on the right shows the following labels and their addresses:

Label	Address
(global)	
main	0x00400000
ex09.asm	
x1	0x10010000
x2	0x10010004
x3	0x10010008
x4	0x1001000c
soma	0x10010010

The 'Registers' window on the right shows the state of the MIPS registers. The 'Registers' tab is selected, showing the register number, name, and value. The 'Coproc 1' and 'Coproc 0' tabs are also visible.

Register	Name	Number	Value
\$zero		0	0x00000000
\$at		1	0x00000000
\$v0		2	0x00000000
\$v1		3	0x00000000
\$a0		4	0x00000000
\$a1		5	0x00000000
\$a2		6	0x00000000
\$a3		7	0x00000000
\$a4		8	0x00000000
\$t1		9	0x00000004
\$t2		10	0x00000000
\$t3		11	0x00000000
\$t4		12	0x00000000
\$t5		13	0x00000000
\$t6		14	0x00000000
\$t7		15	0x00000000
\$s0		16	0x00000004
\$s1		17	0x00000018
\$s2		18	0x00000004
\$s3		19	0x00000011
\$s4		20	0x00000000
\$s5		21	0x00000000
\$s6		22	0x00000000
\$s7		23	0x00000000
\$s8		24	0x00000000
\$s9		25	0x00000000
\$t8		26	0x00000000
\$t9		27	0x00000000
\$d0		28	0x00000000
\$d1		29	0x00000000
\$d2		30	0x00000000
\$ra		31	0x00000000
\$pc			0x00400004
\$hi			0x00000000
\$lo			0x00000000

// programa 10

Considere o seguinte programa:  $y = 127x - 65z + 1$

Faça um programa que calcule o valor de y conhecendo os valores de x e z. Os valores de x e z estão armazenados na memória e, na posição imediatamente a seguir, o valor de y deverá ser escrito, ou seja:

```

.data
x: .word 5
z: .word 7
y: .word 0 # esse valor deverá ser sobrescrito após a execução do programa.

```

```

1  # Exercício 10
2
3  #início
4
5  .text
6  .globl main
7
8  main:
9
10 # Atribuições
11 .data
12 x: .word 5
13 z: .word 7
14 y: .word 0
15
16 .text
17
18 lui $t0, 0x1001
19
20 lw $s0, 0($t0) # x
21 lw $s1, 4($t0) # z
22
23 sll $t1, $s0, 7 # 128x = 2^7
24 sub $t1, $t1, $s0 #127x
25
26 sll $t2, $s1, 6 # 64z = 2^6
27 add $t2, $t2, $s1 # 65z
28
29 sub $t3, $t1, $t2 # 127x-65z
30 addi $t3, $t3, 1 # 127x-65z + 1
31
32 sw $t3, 0($t0)
33
34 #fim

```

Dispt	Address	Code	Basic	Source
	0x00400000	0x3c081001	lui \$t0, 0x1001	18: lui \$t0, 0x1001
	0x00400004	0x8d100000	lw \$s0, 0(\$t0) # x	20: lw \$s0, 0(\$t0) # x
	0x00400008	0x8d100004	lw \$s1, 4(\$t0) # z	21: lw \$s1, 4(\$t0) # z
	0x0040000c	0x00148001	sll \$t1, \$s0, 7 # 128x = 2^7	23: sll \$t1, \$s0, 7 # 128x = 2^7
	0x00400010	0x01304822	sub \$t1, \$t1, \$s0 #127x	24: sub \$t1, \$t1, \$s0 #127x
	0x00400014	0x00151801	sll \$t2, \$s1, 6 # 64z = 2^6	26: sll \$t2, \$s1, 6 # 64z = 2^6
	0x00400018	0x01858000	add \$t2, \$t2, \$s1 # 65z	27: add \$t2, \$t2, \$s1 # 65z
	0x0040001c	0x012a5822	sub \$t3, \$t1, \$t2 # 127x-65z	29: sub \$t3, \$t1, \$t2 # 127x-65z
	0x00400020	0x014b0001	addi \$t3, \$t3, 1 # 127x-65z + 1	30: addi \$t3, \$t3, 1 # 127x-65z + 1
	0x00400024	0x0a000000	sw \$t3, 0(\$t0)	32: sw \$t3, 0(\$t0)

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x0000007b
\$t2	10	0x000000c7
\$t3	11	0x000000b5
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000005
\$s1	17	0x00000007
\$s2	18	0x00000000
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$s8	24	0x00000000
\$s9	25	0x00000000
\$a0	26	0x00000000
\$a1	27	0x00000000
\$a2	28	0x00000000
\$a3	29	0x00000000
\$a4	30	0x00000000
\$a5	31	0x00000000
\$a6	32	0x00000000
\$a7	33	0x00000000
\$a8	34	0x00000000
\$a9	35	0x00000000
\$a10	36	0x00000000
\$a11	37	0x00000000
\$a12	38	0x00000000
\$a13	39	0x00000000
\$a14	40	0x00000000
\$a15	41	0x00000000
\$a16	42	0x00000000
\$a17	43	0x00000000
\$a18	44	0x00000000
\$a19	45	0x00000000
\$a20	46	0x00000000
\$a21	47	0x00000000
\$a22	48	0x00000000
\$a23	49	0x00000000
\$a24	50	0x00000000
\$a25	51	0x00000000
\$a26	52	0x00000000
\$a27	53	0x00000000
\$a28	54	0x00000000
\$a29	55	0x00000000
\$a30	56	0x00000000
\$a31	57	0x00000000
\$a32	58	0x00000000
\$a33	59	0x00000000
\$a34	60	0x00000000
\$a35	61	0x00000000
\$a36	62	0x00000000
\$a37	63	0x00000000

// programa 11

Considere o seguinte programa:  $y = x - z + 300000$

Faça um programa que calcule o valor de y conhecendo os valores de x e z. Os valores de x e z estão armazenados na memória e, na posição imediatamente a seguir, o valor de y deverá ser escrito, ou seja:

```

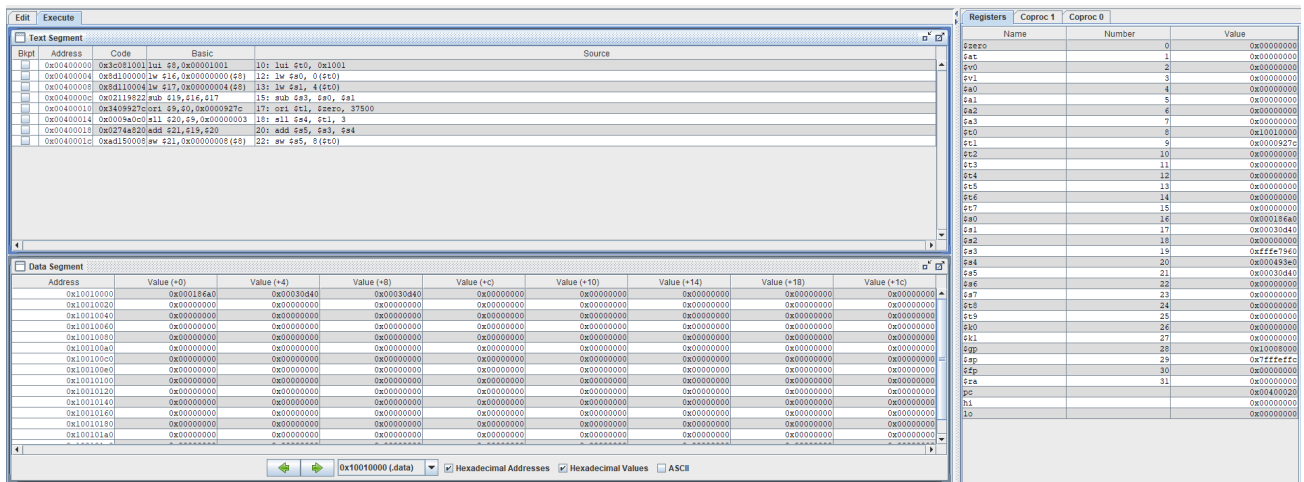
.data
x: .word 100000
z: .word 200000
y: .word 0 # esse valor deverá ser sobrescrito após a execução do programa.

```

```

1  # Exercício 11
2
3  .data
4  x: .word 100000
5  z: .word 200000
6  y: .word 0
7
8  .text
9
10 lui $t0, 0x1001
11
12 lw $s0, 0($t0)
13 lw $s1, 4($t0)
14
15 sub $s3, $s0, $s1
16
17 ori $t1, $zero, 37500
18 sll $s4, $t1, 3
19
20 add $s5, $s3, $s4
21
22 sw $s5, 8($t0)

```



// programa 12

Considere a seguinte situação:

```
int ***x;
```

onde x contem um ponteiro para um ponteiro para um ponteiro para um inteiro.

Nessa situação, considere que a posição inicial de memória contenha o inteiro em questão. Coloque todos os outros valores em registradores, use os endereços de memória que quiser dentro do espaço de endereçamento do Mips.

Resumo do problema:

$k = \text{MEM} [ \text{MEM} [ \text{MEM} [ x ] ] ]$ .

Crie um programa que implemente a estrutura de dados acima, leia o valor de K, o multiplique por 2 e o reescreva no local correto conhecendo-se apenas o valor de x.

Para os programas a seguir use instruções de desvio (beq, bne, j)

```
1  # Exercício 12
2
3  .data
4  x: .word 10
5  p: .word 0
6  pp: .word 0
7  ppp: .word 0
8
9  .text
10 lui $t0, 0x1001 # endereço do inteiro
11 addi $t1, $t0, 4
12 addi $t2, $t0, 8
13 addi $t3, $t0, 12
14
15 sw $t0, 4($t0)
16 sw $t1, 8($t0)
17 sw $t2, 12($t0)
18
19
20 lw $t1, 0($t3)
21 lw $t1, 0($t1)
22 lw $t1, 0($t1)
23 lw $t1, 0($t1)
24
25 sll $t1, $t1, 1
26
27 lw $t2, 0($t3)
28 lw $t2, 0($t2)
29 lw $t2, 0($t2)
30 sw $t1, 0($t2)
```

[illegible]

```
// programa 13:
```

Escreva um programa que leia um valor A da memória, identifique se o número é negativo ou não e encontre o seu módulo. O valor deverá ser reescrito sobre A.

```

1  # Exercício 13
2
3  .data
4  A: word -100
5
6  .text
7  ori $t0, $zero, 1
8
9  lui $t1, 0x1001
10
11 lw $t3, 0($t1)
12
13 srl $t2, $t3, 31
14
15 beq $t0, $t2, negativo
16   escreva:
17
18 negativo:
19   sub $t3, $zero, $t3
20
21 escreva:
22   sw $t3 0($t1)
23

```

[illegible]

```
// programa 14:
```

Escreva um programa que leia um valor A da memória, identifique se o número é par ou não. Um valor deverá ser escrito na segunda posição livre da memória (0 para par e 1 para ímpar).

```

1 # Exercício 14
2
3 .data
4 A: .word 22
5
6 .text
7 lui $t0, 0x1001
8
9 lw $t1, 0($t0)
10
11 ori $t2, $zero, 1
12
13 and $t3, $t1, $t2
14
15 sw $t3, 4($t0)
16

```

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

0x10010000 (data)

Hexadecimal Addresses

Hexadecimal Values

ASCII

</



Edit Execute
Registers Coproc 1 Coproc 0

**Text Segment**

Dispt	Address	Code	Basic	Source
	0x00400000	0x34080064	ori \$5, \$0, 0x00000064	4: ori \$t0, \$zero, 100
	0x00400004	0x34090000	ori \$9, \$0, 0x00000000	5: ori \$t1, \$zero, 0
	0x00400008	0x34080001	ori \$1, \$0, 0x00000001	6: ori \$t6, \$zero, 1
	0x0040000c	0x00100101	li \$1, 0x00000101	7: li \$t5, 0x101
	0x00400010	0x00007820	add \$10, \$0, \$13	8: add \$t7, \$zero, \$t5
	0x00400014	0x11000007	beq \$5, \$0, 0x00000007	11: beq \$t0, \$zero, part2
	0x00400018	0x00098481	slw \$10, \$0, 0x00000000	12: slw \$t2, \$t1
	0x0040001c	0x014e5020	add \$10, \$10, \$14	13: add \$t2, \$t2, \$t6
	0x00400020	0x05ea0000	sw \$10, 0x00000000(\$t4)	14: sw \$t2, 0(\$t7)
	0x00400024	0x12ef0004	addi \$15, \$15, 0x0000...	16: addi \$t7, \$t7, 4
	0x00400028	0x00440022	sub \$5, \$5, \$14	17: sub \$t0, \$t0, \$t6
	0x0040002c	0x012e4820	addi \$9, \$9, \$14	18: addi \$t1, \$t1, \$t6
	0x00400030	0x00100005	0x00000004	19: j loop
	0x00400034	0x34080064	ori \$10, \$0, 0x00000064	22: ori \$t0, \$zero, 100
	0x00400038	0x340a0000	ori \$10, \$0, 0x00000000	23: ori \$t2, \$zero, 0
	0x0040003c	0x11000005	beq \$5, \$0, 0x00000005	25: beq \$t0, \$zero, fin
	0x00400040	0x05ea0000	sw \$5, 0x00000000(\$t3)	26: sw \$t5, 0(\$t3)

**Data Segment**

Address	Value (+0)	Value (+4)	Value (+8)	Value (+C)	Value (+10)	Value (+14)	Value (+18)	Value (+1C)
0x10101000	0x00000001	0x00000003	0x00000005	0x00000007	0x00000009	0x0000000b	0x0000000d	0x0000000f
0x10101004	0x00000011	0x00000015	0x00000019	0x0000001d	0x00000021	0x00000025	0x00000029	0x0000002d
0x10101008	0x00000021	0x00000025	0x00000029	0x0000002d	0x00000031	0x00000035	0x00000039	0x0000003d
0x1010100c	0x00000031	0x00000035	0x00000039	0x0000003d	0x00000041	0x00000045	0x00000049	0x0000004d
0x10101010	0x00000041	0x00000045	0x00000049	0x0000004d	0x00000051	0x00000055	0x00000059	0x0000005d
0x10101014	0x00000051	0x00000055	0x00000059	0x0000005d	0x00000061	0x00000065	0x00000069	0x0000006d
0x10101018	0x00000061	0x00000065	0x00000069	0x0000006d	0x00000071	0x00000075	0x00000079	0x0000007d
0x1010101c	0x00000071	0x00000075	0x00000079	0x0000007d	0x00000081	0x00000085	0x00000089	0x0000008d
0x10101020	0x00000081	0x00000085	0x00000089	0x0000008d	0x00000091	0x00000095	0x00000099	0x0000009d
0x10101024	0x00000091	0x00000095	0x00000099	0x0000009d	0x000000a1	0x000000a5	0x000000a9	0x000000ad
0x10101028	0x0000009d	0x000000a1	0x000000a5	0x000000a9	0x000000ad	0x000000b1	0x000000b5	0x000000b9
0x1010102c	0x000000a1	0x000000a5	0x000000a9	0x000000ad	0x000000b1	0x000000b5	0x000000b9	0x000000bd
0x10101030	0x000000a5	0x000000a9	0x000000ad	0x000000b1	0x000000b5	0x000000b9	0x000000bd	0x000000c1
0x10101034	0x000000a9	0x000000ad	0x000000b1	0x000000b5	0x000000b9	0x000000bd	0x000000c1	0x000000c5
0x10101038	0x000000ad	0x000000b1	0x000000b5	0x000000b9	0x000000bd	0x000000c1	0x000000c5	0x000000c9
0x1010103c	0x000000b1	0x000000b5	0x000000b9	0x000000bd	0x000000c1	0x000000c5	0x000000c9	0x000000cd
0x10101040	0x000000b5	0x000000b9	0x000000bd	0x000000c1	0x000000c5	0x000000c9	0x000000cd	0x000000d1

**// programa 16**

Escreva um programa que avalie a expressão:  $(x*y)/z$ .

Use  $x = 1600000$  ( $=0x186A00$ ),  $y = 80000$  ( $=0x13880$ ), e  $z = 400000$  ( $=0x61A80$ ).

**Inicializar os registradores com os valores acima.**

```

1  # Exercício 16
2
3  .data
4  x: word 0x186A00
5  y: word 0x13980
6  z: word 0x61A80
7
8  .text
9  lui $t0, 0x1001
10
11 lw $s0, 0($t0)      #s0 = x
12 lw $s1, 4($t0)      #s1 = y
13 lw $s2, 8($t0)      #s2 = z
14
15 mult $s0, $s1 # x * y
16 mflo $t1 # resultado da multiplicacao
17
18 div $t1, $s2
19 mflo $t2 # resultado da divisao

```

[illegible]

**// programa 17**

Para a expressão a seguir, escreva um programa que calcule o valor de k:

**k = x \* y (Você deverá realizar a multiplicação através de somas!)**

O valor de  $x$  deve ser lido da primeira posição livre da memória e o valor de  $y$  deverá lido da segunda posição livre. O valor de  $k$ , após calculado, deverá ainda ser escrito na terceira posição livre da memória.

```

1 # Exercício 17
2
3 .data
4 x: .word 10
5 y: .word 200
6
7 .text
8
9 lui $t0, 0x1001
10 lw $t1, 0($t0)
11 lw $t2, 4($t0)
12 ori $t3, $zero, 1
13
14 loop:
15 beq $t2, $zero, fim
16 sub $t2, $t2, $t3
17 add $t4, $t4, $t1
18 j loop
19 fim:
20 sw $t4, 8($t0)
21
22
23

```

Edit Execute				Registers Coproc 1 Coproc 0			
Text Segment				Name Number Value			
Byte	Address	Code	Basic Source				
<input type="checkbox"/>	0x00400000	0x3c081001	lui \$t0, 0x1001	\$zero	0	0x00000000	
<input type="checkbox"/>	0x00400004	0x1d090000	lw \$t1, 0(\$t0)	\$at	1	0x00000000	
<input type="checkbox"/>	0x00400008	0x3d0a0004	lw \$t2, 4(\$t0)	\$v0	2	0x00000000	
<input type="checkbox"/>	0x0040000c	0x3d0b0001	ori \$t3, \$zero, 1	\$v1	3	0x00000000	
<input type="checkbox"/>	0x00400010	0x11400003	beq \$t2, \$zero, fim	\$a0	4	0x00000000	
<input type="checkbox"/>	0x00400014	0x01405022	sub \$t2, \$t2, \$t3	\$a1	5	0x00000000	
<input type="checkbox"/>	0x00400018	0x01890020	add \$t4, \$t4, \$t1	\$a2	6	0x00000000	
<input type="checkbox"/>	0x0040001c	0x01000047	j loop	\$a3	7	0x00000000	
<input type="checkbox"/>	0x00400020	0x3d0c0008	sw \$t4, 8(\$t0)	\$t0	8	0x00101000	
				\$t1	9	0x0000000a	
				\$t2	10	0x00000000	
				\$t3	11	0x00000001	
				\$t4	12	0x000007d0	
				\$t5	13	0x00000000	
				\$t6	14	0x00000000	
				\$t7	15	0x00000000	
				\$a0	16	0x00000000	
				\$a1	17	0x00000000	
				\$a2	18	0x00000000	
				\$a3	19	0x00000000	
				\$a4	20	0x00000000	
				\$a5	21	0x00000000	
				\$a6	22	0x00000000	
				\$a7	23	0x00000000	
				\$t8	24	0x00000000	
				\$t9	25	0x00000000	
				\$t0	26	0x00000000	
				\$t1	27	0x00000000	
				\$fp	28	0x00000000	
				\$sp	29	0x00000000	
				\$ra	30	0x00000000	
				\$s0	31	0x00000000	
				\$s1		0x00000000	
				\$s2		0x00000000	
				\$s3		0x00000000	
				\$s4		0x00000000	
				\$s5		0x00000000	
				\$s6		0x00000000	
				\$s7		0x00000000	
				\$s8		0x00000000	
				\$s9		0x00000000	
				\$s10		0x00000000	
				\$s11		0x00000000	
				\$s12		0x00000000	
				\$s13		0x00000000	
				\$s14		0x00000000	
				\$s15		0x00000000	
				\$s16		0x00000000	
				\$s17		0x00000000	
				\$s18		0x00000000	
				\$s19		0x00000000	
				\$s20		0x00000000	
				\$s21		0x00000000	
				\$s22		0x00000000	
				\$s23		0x00000000	
				\$s24		0x00000000	
				\$s25		0x00000000	
				\$s26		0x00000000	
				\$s27		0x00000000	
				\$s28		0x00000000	
				\$s29		0x00000000	
				\$s30		0x00000000	
				\$s31		0x00000000	
				\$s32		0x00000000	
				\$s33		0x00000000	
				\$s34		0x00000000	
				\$s35		0x00000000	
				\$s36		0x00000000	
				\$s37		0x00000000	
				\$s38		0x00000000	
				\$s39		0x00000000	
				\$s40		0x00000000	
				\$s41		0x00000000	
				\$s42		0x00000000	
				\$s43		0x00000000	
				\$s44		0x00000000	
				\$s45		0x00000000	
				\$s46		0x00000000	
				\$s47		0x00000000	
				\$s48		0x00000000	
				\$s49		0x00000000	
				\$s50		0x00000000	
				\$s51		0x00000000	
				\$s52		0x00000000	
				\$s53		0x00000000	
				\$s54		0x00000000	
				\$s55		0x00000000	
				\$s56		0x00000000	
				\$s57		0x00000000	
				\$s58		0x00000000	
				\$s59		0x00000000	
				\$s60		0x00000000	
				\$s61		0x00000000	
				\$s62		0x00000000	
				\$s63		0x00000000	
				\$s64		0x00000000	
				\$s65		0x00000000	
				\$s66		0x00000000	
				\$s67		0x00000000	
				\$s68		0x00000000	
				\$s69		0x00000000	
				\$s70		0x00000000	
				\$s71		0x00000000	
				\$s72		0x00000000	
				\$s73		0x00000000	
				\$s74		0x00000000	
				\$s75		0x00000000	
				\$s76		0x00000000	
				\$s77		0x00000000	
				\$s78		0x00000000	
				\$s79		0x00000000	
				\$s80		0x00000000	
				\$s81		0x00000000	
				\$s82		0x00000000	
				\$s83		0x00000000	
				\$s84		0x00000000	
				\$s85		0x00000000	
				\$s86		0x00000000	
				\$s87		0x00000000	
				\$s88		0x00000000	
				\$s89		0x00000000	
				\$s90		0x00000000	
				\$s91		0x00000000	
				\$s92		0x00000000	
				\$s93		0x00000000	
				\$s94		0x00000000	
				\$s95		0x00000000	
				\$s96		0x00000000	
				\$s97		0x00000000	
				\$s98		0x00000000	
				\$s99		0x00000000	
				\$s100		0x00000000	
				\$s101		0x00000000	
				\$s102		0x00000000	
				\$s103		0x00000000	
				\$s104		0x00000000	
				\$s105		0x00000000	
				\$s106		0x00000000	
				\$s107		0x00000000	
				\$s108		0x00000000	
				\$s109		0x00000000	
				\$s110		0x00000000	
				\$s111		0x00000000	
				\$s112		0x00000000	
				\$s113		0x00000000	
				\$s114		0x00000000	
				\$s115		0x00000000	
				\$s116		0x00000000	
				\$s117		0x00000000	
				\$s118		0x00000000	
				\$s119		0x00000000	
				\$s120		0x00000000	
				\$s121		0x00000000	
				\$s122		0x00000000	
				\$s123		0x00000000	
				\$s124		0x00000000	
				\$s125		0x00000000	
				\$s126		0x00000000	
				\$s127		0x00000000	
				\$s128		0x00000000	
				\$s129		0x00000000	
				\$s130		0x00000000	
				\$s131		0x00000000	
				\$s132		0x00000000	
				\$s133		0x00000000	
				\$s134		0x00000000	
				\$s135		0x00000000	
				\$s136		0x00000000	
				\$s137		0x00000000	
				\$s138		0x00000000	
				\$s139		0x00000000	
				\$s140		0x00000000	
				\$s141		0x00000000	
				\$s142		0x00000000	
				\$s143		0x00000000	
				\$s144		0x00000000	
				\$s145		0x00000000	
				\$s146		0x00000000	
				\$s147		0x00000000	
				\$s148		0x00000000	
				\$s149		0x00000000	
				\$s150		0x00000000	
				\$s151		0x00000000	
				\$s152		0x00000000	
				\$s153		0x00000000	
				\$s154		0x00000000	
				\$s155		0x00000000	
				\$s156		0x00000000	
				\$s157		0x00000000	
				\$s158		0x00000000	
				\$s159		0x00000000	
				\$s160		0x00000000	
				\$s161		0x00000000	
				\$s162		0x00000000	
				\$s163		0x00000000	
				\$s164		0x00000000	
				\$s165		0x00000000	
				\$s166		0x00000000	
				\$s167		0x00000000	
				\$s168		0x00000000	
				\$s169		0x00000000	
				\$s170		0x00000000	
				\$s171		0x00000000	
				\$s172		0x00000000	
				\$s173		0x00000000	
				\$s174		0x00000000	
				\$s175		0x00000000	
				\$s176		0x00000000	
				\$s177		0x00000000	
				\$s178		0x00000000	
				\$s179		0x00000000	
				\$s180		0x00000000	
				\$s181		0x00000000	
				\$s182		0x00000000	
				\$s183		0x00000000	
				\$s184		0x00000000	
				\$s185		0x00000000	
				\$s186		0x00000000	
				\$s187		0x00000000	
				\$s188		0x00000000	
				\$s189		0x00000000	
				\$s190		0x00000000	
				\$s191		0x00000000	
				\$s192		0x00000000	
				\$s193		0x00000000	
				\$s194		0x00000000	
				\$s195		0x00000000	
				\$s196		0x00000000	
				\$s197		0x00000000	
				\$s198		0x00000000	
				\$s199		0x00000000	
				\$s200		0x00000000	
				\$s201		0x00000000	
				\$s202		0x00000000	
				\$s203		0x00000000	
				\$s204		0x00000000	
				\$s205		0x00000000	
				\$s206		0x00000000	
				\$s207		0x00000000	
				\$s208		0x00000000	
				\$s209		0x00000000	
				\$s210		0x00000000	
				\$s211		0x00000000	
				\$s212		0x00000000	
				\$s213		0x00000000	
				\$s214		0x00000000	
				\$s215		0x00000000	
				\$s216		0x00000000	
				\$s217		0x00000000	
				\$s218		0x00000000	
				\$s219		0x00000000	
				\$s220		0x00000000	
				\$s221		0x00000000	
				\$s222		0x00000000	
				\$s223		0x00000000	
				\$s224		0x00000000	
				\$s225		0x00000000	
				\$s226		0x00000000	
				\$s227		0x00000000	
				\$s228		0x00000000	
				\$s229		0x00000000	
				\$s230		0x00000000	
				\$s231		0x00000000	
				\$s232		0x00000000	
				\$s233		0x00000000	
				\$s234		0x00000000	
				\$s235		0x00000000	
				\$s236		0	

```

3 .data
4 x: .word 1
5 y: .word 0
6
7 .text
8
9 # 5^2 = 5 * 5 = (5+5+5+5+5)
10 # 5^3 = 5 * 5 * 5 = (5+5+5+5+5) + (5+5+5+5+5) + (5+5+5+5+5) + (5+5+5+5+5) + (5+5+5+5+5)
11 # 5^4 = 5*5*5*5 = 5^3 + 5^3 + 5^3 + 5^3 + 5^3
12
13 lui $t0, 0x1001
14 lw $t1, 0($t0) # t1 = X
15 lw $t2, 4($t0) # t2 = Y
16 ori $t3, $zero, 1 # t3 = 1
17 or $t5, $zero, $t1 # t5 = X
18 or $t7, $zero, $t1 # t7 = X
19 addi $t6, $t2, -1 # t6 = Y - 1
20
21
22 beq $t2, $zero, exp_0 # Caso o expoente for 0, go to exp_0
23 beq $t6, $zero, loop_2 # Caso o expoente for 1, go to loop_2. Porque no entraria no loop_1, porque t6 = 0
24
25 loop_1:
26 beq $t6, $zero, fim # Repetir (y-1)vezes
27 or $t4, $zero, $zero # t4 = 0 - importante a partir do segundo loop_1
28
29 loop_2:
30 beq $t5, $zero, controle # repetir x vezes e then go to controle
31 sub $t5, $t5, $t3 # t5 = t5 - 1
32 add $t4, $t4, $t1 # t4 = t4 + t1 (resultado)
33 beq $t2, $t3, fim # Se t2 = 1 sair
34 j loop_2
35
36 controle:
37 or $t5, $zero, $t7 # t5 = X
38 or $t1, $zero, $t4 # t1 = t4
39 sub $t6, $t6, $t3 # t6 = t6 - 1
40 j loop_1
41
42 exp_0:
43 ori $t4, $zero, 1 # resultado = 1
44
45 fim:
46 sw $t4, 0($t0) # Escrevendo

```

Edit		Execute				Registers		Coproc 1	Coproc 0		
Text Segment						Registers		Number	Value		
Offset	Address	Code	Basic	Source			Name				
0x00400000	0x00400000	lui \$t0, 0x1001	13: lui \$t0, 0x1001			\$zero	0		0x00000000		
0x00400004	0x00400004	lw \$t1, 0(\$t0)	14: lw \$t1, 0(\$t0) # t1 = X			\$t0	1		0x00000000		
0x00400008	0x00400008	lw \$t2, 4(\$t0)	15: lw \$t2, 4(\$t0) # t2 = Y			\$t0	2		0x00000000		
0x0040000c	0x0040000c	ori \$t3, \$zero, 1	16: ori \$t3, \$zero, 1 # t3 = 1			\$t0	3		0x00000000		
0x00400010	0x00400010	or \$t5, \$zero, \$t1	17: or \$t5, \$zero, \$t1 # t5 = X			\$t0	4		0x00000000		
0x00400014	0x00400014	or \$t7, \$zero, \$t1	18: or \$t7, \$zero, \$t1 # t7 = X			\$t0	5		0x00000000		
0x00400018	0x00400018	addi \$t6, \$t2, -1	19: addi \$t6, \$t2, -1 # t6 = Y - 1			\$t0	6		0x00000000		
0x0040001c	0x0040001c	beq \$t2, \$zero, exp_0	22: beq \$t2, \$zero, exp_0 # Caso o expoente for 0, go to exp_0			\$t0	7		0x00000000		
0x00400020	0x00400020	beq \$t6, \$zero, loop_2	23: beq \$t6, \$zero, loop_2 # Caso o expoente for 1, go to loop_2. Porque no entraria no loop_1, porque t6 = 0			\$t0	8		0x00000000		
0x00400024	0x00400024	beq \$t5, \$zero, fim	26: beq \$t5, \$zero, fim # Repetir (y-1)vezes			\$t0	9		0x00000000		
0x00400028	0x00400028	or \$t4, \$zero, \$zero	27: or \$t4, \$zero, \$zero # t4 = 0 - importante a partir do segundo loop_1			\$t0	10		0x00000000		
0x0040002c	0x0040002c	beq \$t5, \$zero, controle	30: beq \$t5, \$zero, controle # repetir x vezes e then go to controle			\$t0	11		0x00000000		
0x00400030	0x00400030	sub \$t5, \$t5, \$t3	31: sub \$t5, \$t5, \$t3 # t5 = t5 - 1			\$t0	12		0x00000000		
0x00400034	0x00400034	add \$t4, \$t4, \$t1	32: add \$t4, \$t4, \$t1 # t4 = t4 + t1 (resultado)			\$t0	13		0x00000000		
0x00400038	0x00400038	beq \$t2, \$t3, fim	33: beq \$t2, \$t3, fim # Se t2 = 1 sair			\$t0	14		0x00000000		
0x0040003c	0x0040003c	j loop_2	34: j loop_2			\$t0	15		0x00000000		
0x00400040	0x00400040	sw \$t4, 0(\$t0)	43: sw \$t4, 0(\$t0) # Escrevendo			\$t0	16		0x00000000		
0x00400044	0x00400044	sw \$t1, 4(\$t0)	44: sw \$t1, 4(\$t0) # Escrevendo			\$t0	17		0x00000000		
0x00400048	0x00400048	sw \$t2, 8(\$t0)	45: sw \$t2, 8(\$t0) # Escrevendo			\$t0	18		0x00000000		
0x0040004c	0x0040004c	sw \$t3, 12(\$t0)	46: sw \$t3, 12(\$t0) # Escrevendo			\$t0	19		0x00000000		
0x00400050	0x00400050	sw \$t6, 16(\$t0)	47: sw \$t6, 16(\$t0) # Escrevendo			\$t0	20		0x00000000		
0x00400054	0x00400054	sw \$t7, 20(\$t0)	48: sw \$t7, 20(\$t0) # Escrevendo			\$t0	21		0x00000000		
0x00400058	0x00400058	sw \$t8, 24(\$t0)	49: sw \$t8, 24(\$t0) # Escrevendo			\$t0	22		0x00000000		
0x0040005c	0x0040005c	sw \$t9, 28(\$t0)	50: sw \$t9, 28(\$t0) # Escrevendo			\$t0	23		0x00000000		
0x00400060	0x00400060	sw \$t0, 32(\$t0)	51: sw \$t0, 32(\$t0) # Escrevendo			\$t0	24		0x00000000		
0x00400064	0x00400064	sw \$t1, 36(\$t0)	52: sw \$t1, 36(\$t0) # Escrevendo			\$t0	25		0x00000000		
0x00400068	0x00400068	sw \$t2, 40(\$t0)	53: sw \$t2, 40(\$t0) # Escrevendo			\$t0	26		0x00000000		
0x0040006c	0x0040006c	sw \$t3, 44(\$t0)	54: sw \$t3, 44(\$t0) # Escrevendo			\$t0	27		0x00000000		
0x00400070	0x00400070	sw \$t4, 48(\$t0)	55: sw \$t4, 48(\$t0) # Escrevendo			\$t0	28		0x00000000		
0x00400074	0x00400074	sw \$t5, 52(\$t0)	56: sw \$t5, 52(\$t0) # Escrevendo			\$t0	29		0x00000000		
0x00400078	0x00400078	sw \$t6, 56(\$t0)	57: sw \$t6, 56(\$t0) # Escrevendo			\$t0	30		0x00000000		
0x0040007c	0x0040007c	sw \$t7, 60(\$t0)	58: sw \$t7, 60(\$t0) # Escrevendo			\$t0	31		0x00000000		
0x00400080	0x00400080	sw \$t8, 64(\$t0)	59: sw \$t8, 64(\$t0) # Escrevendo			\$t0	32		0x00000000		
0x00400084	0x00400084	sw \$t9, 68(\$t0)	60: sw \$t9, 68(\$t0) # Escrevendo			\$t0	33		0x00000000		
0x00400088	0x00400088	sw \$t0, 72(\$t0)	61: sw \$t0, 72(\$t0) # Escrevendo			\$t0	34		0x00000000		
0x0040008c	0x0040008c	sw \$t1, 76(\$t0)	62: sw \$t1, 76(\$t0) # Escrevendo			\$t0	35		0x00000000		
0x00400090	0x00400090	sw \$t2, 80(\$t0)	63: sw \$t2, 80(\$t0) # Escrevendo			\$t0	36		0x00000000		
0x00400094	0x00400094	sw \$t3, 84(\$t0)	64: sw \$t3, 84(\$t0) # Escrevendo			\$t0	37		0x00000000		
0x00400098	0x00400098	sw \$t4, 88(\$t0)	65: sw \$t4, 88(\$t0) # Escrevendo			\$t0	38		0x00000000		
0x0040009c	0x0040009c	sw \$t5, 92(\$t0)	66: sw \$t5, 92(\$t0) # Escrevendo			\$t0	39		0x00000000		
0x004000a0	0x004000a0	sw \$t6, 96(\$t0)	67: sw \$t6, 96(\$t0) # Escrevendo			\$t0	40		0x00000000		
0x004000a4	0x004000a4	sw \$t7, 100(\$t0)	68: sw \$t7, 100(\$t0) # Escrevendo			\$t0	41		0x00000000		
0x004000a8	0x004000a8	sw \$t8, 104(\$t0)	69: sw \$t8, 104(\$t0) # Escrevendo			\$t0	42		0x00000000		
0x004000ac	0x004000ac	sw \$t9, 108(\$t0)	70: sw \$t9, 108(\$t0) # Escrevendo			\$t0	43		0x00000000		
0x004000b0	0x004000b0	sw \$t0, 112(\$t0)	71: sw \$t0, 112(\$t0) # Escrevendo			\$t0	44		0x00000000		
0x004000b4	0x004000b4	sw \$t1, 116(\$t0)	72: sw \$t1, 116(\$t0) # Escrevendo			\$t0	45		0x00000000		
0x004000b8	0x004000b8	sw \$t2, 120(\$t0)	73: sw \$t2, 120(\$t0) # Escrevendo			\$t0	46		0x00000000		
0x004000bc	0x004000bc	sw \$t3, 124(\$t0)	74: sw \$t3, 124(\$t0) # Escrevendo			\$t0	47		0x00000000		
0x004000c0	0x004000c0	sw \$t4, 128(\$t0)	75: sw \$t4, 128(\$t0) # Escrevendo			\$t0	48		0x00000000		
0x004000c4	0x004000c4	sw \$t5, 132(\$t0)	76: sw \$t5, 132(\$t0) # Escrevendo			\$t0	49		0x00000000		
0x004000c8	0x004000c8	sw \$t6, 136(\$t0)	77: sw \$t6, 136(\$t0) # Escrevendo			\$t0	50		0x00000000		
0x004000cc	0x004000cc	sw \$t7, 140(\$t0)	78: sw \$t7, 140(\$t0) # Escrevendo			\$t0	51		0x00000000		
0x004000d0	0x004000d0	sw \$t8, 144(\$t0)	79: sw \$t8, 144(\$t0) # Escrevendo			\$t0	52		0x00000000		
0x004000d4	0x004000d4	sw \$t9, 148(\$t0)	80: sw \$t9, 148(\$t0) # Escrevendo			\$t0	53		0x00000000		
0x004000d8	0x004000d8	sw \$t0, 152(\$t0)	81: sw \$t0, 152(\$t0) # Escrevendo			\$t0	54		0x00000000		
0x004000dc	0x004000dc	sw \$t1, 156(\$t0)	82: sw \$t1, 156(\$t0) # Escrevendo			\$t0	55		0x00000000		
0x004000e0	0x004000e0	sw \$t2, 160(\$t0)	83: sw \$t2, 160(\$t0) # Escrevendo			\$t0	56		0x00000000		
0x004000e4	0x004000e4	sw \$t3, 164(\$t0)	84: sw \$t3, 164(\$t0) # Escrevendo			\$t0	57		0x00000000		
0x004000e8	0x004000e8	sw \$t4, 168(\$t0)	85: sw \$t4, 168(\$t0) # Escrevendo			\$t0	58		0x00000000		
0x004000ec	0x004000ec	sw \$t5, 172(\$t0)	86: sw \$t5, 172(\$t0) # Escrevendo			\$t0	59		0x00000000		
0x004000f0	0x004000f0	sw \$t6, 176(\$t0)	87: sw \$t6, 176(\$t0) # Escrevendo			\$t0	60		0x00000000		
0x004000f4	0x004000f4	sw \$t7, 180(\$t0)	88: sw \$t7, 180(\$t0) # Escrevendo			\$t0	61		0x00000000		
0x004000f8	0x004000f8	sw \$t8, 184(\$t0)	89: sw \$t8, 184(\$t0) # Escrevendo			\$t0	62		0x00000000		
0x004000fc	0x004000fc	sw \$t9, 188(\$t0)	90: sw \$t9, 188(\$t0) # Escrevendo			\$t0	63		0x00000000		
0x00400100	0x00400100	sw \$t0, 192(\$t0)	91: sw \$t0, 192(\$t0) # Escrevendo			\$t0	64		0x00000000		
0x00400104	0x00400104	sw \$t1, 196(\$t0)	92: sw \$t1, 196(\$t0) # Escrevendo			\$t0	65		0x00000000		
0x00400108	0x00400108	sw \$t2, 200(\$t0)	93: sw \$t2, 200(\$t0) # Escrevendo			\$t0	66		0x00000000		
0x0040010c	0x0040010c	sw \$t3, 204(\$t0)	94: sw \$t3, 204(\$t0) # Escrevendo			\$t0	67		0x00000000		
0x00400110	0x00400110	sw \$t4, 208(\$t0)	95: sw \$t4, 208(\$t0) # Escrevendo			\$t0	68		0x00000000		
0x00400114	0x00400114	sw \$t5, 212(\$t0)	96: sw \$t5, 212(\$t0) # Escrevendo			\$t0	69		0x00000000		
0x00400118	0x00400118	sw \$t6, 216(\$t0)	97: sw \$t6, 216(\$t0) # Escrevendo			\$t0	70		0x00000000		
0x0040011c	0x0040011c	sw \$t7, 220(\$t0)	98: sw \$t7, 220(\$t0) # Escrevendo			\$t0	71		0x00000000		
0x00400120	0x00400120	sw \$t8, 224(\$t0)	99: sw \$t8, 224(\$t0) # Escrevendo			\$t0	72		0x00000000		
0x00400124	0x00400124	sw \$t9, 228(\$t0)	100: sw \$t9, 228(\$t0) # Escrevendo			\$t0	73		0x00000000		
0x00400128	0x00400128	sw \$t0, 232(\$t0)	101: sw \$t0, 232(\$t0) # Escrevendo			\$t0	74		0x00000000		
0x0040012c	0x0040012c	sw \$t1, 236(\$t0)	102: sw \$t1, 236(\$t0) # Escrevendo			\$t0	75		0x00000000		
0x00400130	0x00400130	sw \$t2, 240(\$t0)	103: sw \$t2, 240(\$t0) # Escrevendo			\$t0	76		0x00000000		
0x00400134	0x00400134	sw \$t3, 244(\$t0)	104: sw \$t3, 244(\$t0) # Escrevendo			\$t0	77		0x00000000		
0x00400138	0x00400138	sw \$t4, 248(\$t0)	105: sw \$t4, 248(\$t0) # Escrevendo			\$t0	78		0x00000000		
0x0040013c	0x0040013c	sw \$t5, 252(\$t0)	106: sw \$t5, 252(\$t0) # Escrevendo			\$t0	79		0x00000000		
0x00400140	0x00400140	sw \$t6, 256(\$t0)	107: sw \$t6, 256(\$t0) # Escrevendo			\$t0	80		0x00000000		
0x00400144	0x00400144	sw \$t7, 260(\$t0)	108: sw \$t7, 260(\$t0) # Escrevendo			\$t0	81		0x00000000		
0x00400148	0x00400148	sw \$t8, 264(\$t0)	109: sw \$t8, 264(\$t0) # Escrevendo			\$t0	82		0x00000000		
0x0040014c	0x0040014c	sw \$t9, 268(\$t0)	110: sw \$t9, 268(\$t0) # Escrevendo			\$t0	83		0x00000000		
0x00400150	0x00400150	sw \$t0, 272(\$t0)	111: sw \$t0, 272(\$t0) # Escrevendo			\$t0	84		0x00000000		
0x00400154	0x00400154	sw \$t1, 276(\$t0)	112: sw \$t1, 276(\$t0) # Escrevendo			\$t0	85		0x00000000		
0x00400158	0x00400158	sw \$t2, 280(\$t0)	113: sw \$t2, 280(\$t0) # Escrevendo			\$t0	86		0x00000000		
0x0040015c	0x0040015c	sw \$t3, 284(\$t0)	114: sw \$t3, 284(\$t0) # Escrevendo			\$t0	87		0x00000000		
0x00400160	0x00400160	sw \$t4, 288(\$t0)	115: sw \$t4, 288(\$t0) # Escrevendo			\$t0	88		0x00000000		
0x00400164	0x00400164	sw \$t5, 292(\$t0)	116: sw \$t5, 292(\$t0) # Escrevendo			\$t0	89		0x00000000		
0x00400168	0x00400168	sw \$t6, 296(\$t0)	117: sw \$t6, 296(\$t0) # Escrevendo			\$t0	90		0x00000000		
0x0040016c	0x0040016c	sw \$t7, 300(\$t0)	118: sw \$t7, 300(\$t0) # Escrevendo			\$t0	91		0x00000000		
0x00400170	0x00400170	sw \$t8, 304(\$t0)	119: sw \$t8, 304(\$t0) # Escrevendo			\$t0	92		0x00000000		
0x00400174	0x00400174	sw \$t9, 308(\$t0)	120: sw \$t9, 308(\$t0) # Escrevendo			\$t0	93		0x00000000		
0x00400178	0x00400178	sw \$t0, 312(\$t0)	121: sw \$t0, 312(\$t0) # Escrevendo			\$t0	94		0x00000000		
0x0040017c	0x0040017c	sw \$t1, 316(\$t0)	122: sw \$t1, 316(\$t0) # Escrevendo			\$t0	95		0x00000000		
0x00400180	0x00400180	sw \$t2, 320(\$t0)	123: sw \$t2, 320(\$t0) # Escrevendo			\$t0	96		0x00000000		
0x00400184	0x00400184	sw \$t3, 324(\$t0)	124: sw \$t3, 324(\$t0) # Escrevendo			\$t0	97		0x00000000		
0x00400188	0x00400188	sw \$t4, 328(\$t0)	125: sw \$t4, 328(\$t0) # Escrevendo			\$t0	98		0x00000000		
0x0040018c	0x0040018c	sw \$t5, 332(\$t0)	126: sw \$t5, 332(\$t0) # Escrevendo			\$t0	99		0x00000000		
0x00400190	0x00400190	sw \$t6, 336(\$t0)	127: sw \$t6, 336(\$t0) # Escrevendo			\$t0	100		0x00000000		
0x00400194	0x00400194	sw \$t7, 340(\$t0)	1								



```

1  # Exercício 20
2
3  .data
4  x: .word 3
5
6  .text
7  lui $t0, 0x1001
8  ori $t1, $zero, 2
9
10 lv $s0, 0($t0)
11
12 div $s0, $t1
13 mfh1 $s1
14
15 mult $s0, $s0
16 mflo $s2 # resultado da multi x^2
17
18 mult $s2, $s0
19 mflo $s3 # resultado da multi x^3
20
21 mult $s3, $s0
22 mflo $s4 # resultado da multi x^4
23
24 mult $s4, $s0
25 mflo $s5 # resultado da multi x^5
26
27 beq $s1, $zero, primeira_parte
28 j segunda_parte
29
30 primeira_parte:
31 add $t2, $s4, $s3
32
33 mult $t1, $s2
34 mflo $s6 # 2*x^2
35
36 sub $t3, $t2, $s6
37 sw $t3, 4($t0)
38 j fim
39
40 segunda_parte:
41 sub $t4, $s5, $s3
42 addi $s7, $t4, 1
43 sw $s7, 4($t0)
44
45 fim:

```

0x004000000x3c081001lui \$9, 0x000010017: lui \$t0, 0x1001

0x004000040x34090002ori \$9, \$0, 0x000000028: ori \$t1, \$zero, 2

0x004000080x41000000lw \$s0, 0(\$t0)10: lw \$s0, 0(\$t0)

0x0040000c0x0209001adiv \$f9, \$f912: div \$a0, \$t1

0x004000100x00008810mfhl \$f113: mfhl \$a1

0x004000140x02100018mult \$f0, \$f015: mult \$a0, \$a0

0x004000180x00008812mflo \$f816: mflo \$a2 # resultado da multi x^2

0x0040001c0x02500018mult \$f0, \$f018: mult \$a2, \$a0

0x004000200x00008812mflo \$f819: mflo \$a3 # resultado da multi x^3

0x004000240x02700018mult \$f0, \$f020: mult \$a3, \$a0

0x004000280x00008812mflo \$f822: mflo \$a4 # resultado da multi x^4

0x0040002c0x02800018mult \$f0, \$f024: mult \$a4, \$a0

0x004000300x00008812mflo \$f825: mflo \$a5 # resultado da multi x^5

0x004000340x12200001beq \$t1, \$0, 0x0000000127: beq \$a1, \$zero, primeira parte

0x004000380x02935020add \$t0, \$t0, \$t131: add \$t2, \$a4, \$a3

0x0040003c0x01320018mult \$f0, \$f033: mult \$t2, \$a2

0x004000400x00008812mflo \$f834: mflo \$a6 # 2\*x^2

0x004000440x01565522sub \$t1, \$t0, \$t236: sub \$t3, \$t2, \$a6

0x004000480xad0b0004sw \$t1, 0x00000004(\$t0)37: sw \$t3, 4(\$t0)

0x0040004c0x01300018mult \$f0, \$f038: j fim

0x004000500x01300018mult \$f0, \$f038: j fim

0x004000540x02b34022sub \$t2, \$t2, \$t141: sub \$t4, \$a5, \$a3

0x004000580x21970001addi \$t2, \$t2, 0x0000000142: addi \$a7, \$t4, 1

0x004000000x00000000\$zero00x00000000

0x004000010x00000000\$a101x00000000

0x004000020x00000000\$a002x00000000

0x004000030x00000000\$v103x00000000

0x004000040x00000000\$a004x00000000

0x004000050x00000000\$a105x00000000

0x004000060x00000000\$a206x00000000

0x004000070x00000000\$a307x00000000

0x004000080x00100000\$t008x00100000

0x004000090x00000002\$t109x00000002

0x0040000a0x00000000\$t210x00000000

0x0040000b0x00000000\$t311x00000000

0x0040000c0x00000000\$t412x00000000

0x0040000d0x00000000\$t513x00000000

0x0040000e0x00000000\$t614x00000000

0x0040000f0x00000000\$t715x00000000

0x004000100x00000000\$a616x00000000

0x004000110x00000000\$a717x00000000

0x004000120x00000000\$a018x00000000

0x004000130x00000009\$a219x00000009

0x004000140x00000001\$a420x00000001

0x004000150x0000000b\$a521x0000000b

0x004000160x00000000\$a622x00000000

0x004000170x00000000\$a723x00000000

0x004000180x00000000\$t024x00000000

0x004000190x00000000\$t125x00000000

0x0040001a0x00000000\$t226x00000000

0x0040001b0x00000000\$t327x00000000

0x0040001c0x00000000\$t428x00000000

0x0040001d0x00000000\$t529x00000000

0x0040001e0x00000000\$t630x00000000

0x0040001f0x00000000\$t731x00000000

0x004000200x00000000\$a032x00000000

0x004000210x00000000\$a133x00000000

0x004000220x00000000\$a234x00000000

0x004000230x00000000\$a335x00000000

0x004000240x00000000\$a436x00000000

0x004000250x00000000\$a537x00000000

0x004000260x00000000\$a638x00000000

0x004000270x00000000\$a739x00000000

0x004000280x00000000\$t040x00000000

0x004000290x00000000\$t141x00000000

0x0040002a0x00000000\$t242x00000000

0x0040002b0x00000000\$t343x00000000

0x0040002c0x00000000\$t444x00000000

0x0040002d0x00000000\$t545x00000000

0x0040002e0x00000000\$t646x00000000

0x0040002f0x00000000\$t747x00000000

0x004000300x00000000\$a048x00000000

0x004000310x00000000\$a149x00000000

0x004000320x00000000\$a250x00000000

0x004000330x00000000\$a351x00000000

0x004000340x00000000\$a452x00000000

0x004000350x00000000\$a553x00000000

0x004000360x00000000\$a654x00000000

0x004000370x00000000\$a755x00000000

0x004000380x00000000\$t056x00000000

0x004000390x00000000\$t157x00000000

0x0040003a0x00000000\$t258x00000000

0x0040003b0x00000000\$t359x00000000

0x0040003c0x00000000\$t460x00000000

0x0040003d0x00000000\$t561x00000000

0x0040003e0x00000000\$t662x00000000

0x0040003f0x00000000\$t763x00000000

0x004000400x00000000\$a064x00000000

0x004000410x00000000\$a165x00000000

0x004000420x00000000\$a266x00000000

0x004000430x00000000\$a367x00000000

0x004000440x00000000\$a468x00000000

0x004000450x00000000\$a569x00000000

0x004000460x00000000\$a670x00000000

0x004000470x00000000\$a771x00000000

0x004000480x00000000\$t072x00000000

0x004000490x00000000\$t173x00000000

0x0040004a0x00000000\$t274x00000000

0x0040004b0x00000000\$t375x00000000

0x0040004c0x00000000\$t476x00000000

0x0040004d0x00000000\$t577x00000000

0x0040004e0x00000000\$t678x00000000

0x0040004f0x00000000\$t779x00000000

0x004000500x00000000\$a080x00000000

0x004000510x00000000\$a181x00000000

0x004000520x00000000\$a282x00000000

0x004000530x00000000\$a383x00000000

0x004000540x00000000\$a484x00000000

0x004000550x00000000\$a585x00000000

0x004000560x00000000\$a686x00000000

0x004000570x00000000\$a787x00000000

0x004000580x00000000\$t088x00000000

0x004000590x00000000\$t189x00000000

0x0040005a0x00000000\$t290x00000000

0x0040005b0x00000000\$t391x00000000

0x0040005c0x00000000\$t492x00000000

0x0040005d0x00000000\$t593x00000000

0x0040005e0x00000000\$t694x00000000

0x0040005f0x00000000\$t795x00000000

0x004000000x00000000\$zero00x00000000

0x004000010x00000000\$a101x00000000

0x004000020x00000000\$a002x00000000

0x004000030x00000000\$v103x00000000

0x004000040x00000000\$a004x00000000

0x004000050x00000000\$a105x00000000

0x004000060x00000000\$a206x00000000

0x004000070x00000000\$a307x00000000

0x004000080x00100000\$t008x00100000

0x004000090x00000002\$t109x00000002

0x0040000a0x00000000\$t210x00000000

0x0040000b0x00000000\$t311x00000000

0x0040000c0x00000000\$t412x00000000

0x0040000d0x00000000\$t513x00000000

0x0040000e0x00000000\$t614x00000000

0x0040000f0x00000000\$t715x00000000

0x004000100x00000000\$a616x00000000

0x004000110x00000000\$a717x00000000

0x004000120x00000000\$a018x00000000

0x004000130x00000009\$a219x00000009

0x004000140x00000001\$a420x00000001

0x004000150x0000000b\$a521x0000000b

0x004000160x00000000\$a622x00000000

0x004000170x00000000\$a723x00000000

0x004000180x00000000\$t024x00000000

0x004000190x00000000\$t125x00000000

0x0040001a0x00000000\$t226x00000000

0x0040001b0x00000000\$t327x00000000

0x0040001c0x00000000\$t428x00000000

0x0040001d0x00000000\$t529x00000000

0x0040001e0x00000000\$t630x00000000

0x0040001f0x00000000\$t731x00000000

0x004000200x00000000\$a032x00000000

0x004000210x00000000\$a133x00000000

0x004000220x00000000\$a234x00000000

0x004000230x00000000\$a335x00000000

0x004000240x00000000\$a436x00000000

0x004000250x00000000\$a537x00000000

0x004000260x00000000\$a638x00000000

0x004000270x00000000\$a739x00000000

0x004000280x00000000\$t040x00000000

0x004000290x00000000\$t141x00000000

0x0040002a0x00000000\$t242x00000000

0x0040002b0x00000000\$t343x00000000

0x0040002c0x00000000\$t444x00000000

0x0040002d0x00000000\$t545x00000000

0x0040002e0x00000000\$t646x00000000

0x0040002f0x00000000\$t747x00000000

0x004000300x00000000\$a048x00000000

0x004000310x00000000\$a149x00000000

0x004000320x00000000\$a250x00000000

0x004000330x00000000\$a351x00000000

0x004000340x00000000\$a452x00000000

0x004000350x00000000\$a553x00000000

0x004000360x00000000\$a654x00000000

0x004000370x00000000\$a755x00000000

0x004000380x00000000\$t056x00000000

0x004000390x00000000\$t157x00000000

0x0040003a0x00000000\$t258x00000000

0x0040003b0x00000000\$t359x00000000

0x0040003c0x00000000\$t460x00000000

0x0040003d0x00000000\$t561x00000000

0x0040003e0x00000000\$t662x00000000

0x0040003f0x00000000\$t763x00000000

0x004000400x00000000\$a064x00000000

0x004000410x00000000\$a165x00000000

0x004000420x00000000\$a266x00000000

0x004000430x00000000\$a367x00000000

0x004000440x00000000\$a468x00000000

0x004000450x00000000\$a569x00000000

0x004000460x00000000\$a670x00000000

0x004000470x00000000\$a771x00000000

0x004000480x00000000\$t072x00000000

0x004000490x00000000\$t173x00000000

0x0040004a0x00000000\$t274x00000000

0x0040004b0x00000000\$t375x00000000

0x0040004c0x00000000\$t476x00000000

0x0040004d0x00000000\$t577x00000000

0x0040004e0x00000000\$t678x00000000

0x0040004f0x00000000\$t779x00000000

0x004000500x00000000\$a080x00000000

0x004000510x00000000\$a181x00000000

0x004000520x00000000\$a282x00000000

0x004000530x00000000\$a383x00000000

0x004000540x00000000\$a484x00000000

0x004000550x00000000\$a585x00000000

0x004000560x00000000\$a686x00000000

0x004000570x00000000\$a787x00000000

0x004000580x00000000\$t088x00000000

0x004000590x00000000\$t189x00000000

0x0040005a0x00000000\$t290x00000000

0x0040005b0x00000000\$t391x00000000

0x0040005c0x00000000\$t492x00000000

0x0040005d0x00000000\$t593x00000000

0x0040005e0x00000000\$t694x00000000

0x0040005f0x00000000\$t795x00000000

0x004000000x00000000\$zero00x00000000

0x004000010x00000000\$a101x00000000

0x004000020x00000000\$a002x00000000

0x004000030x00000000\$v103x00000000

0x004000040x00000000\$a004x00000000

0x004000050x00000000\$a105x00000000

0x004000060x00000000\$a206x00000000

0x004000070x00000000\$a307x00000000

0x004000080x00100000\$t008x00100000

0x004000090x00000002\$t109x00000002

0x0040000a0x00000000\$t210x00000000

0x0040000b0x00000000\$t311x00000000

0x0040000c0x00000000\$t412x00000000

0x0040000d0x00000000\$t513x00000000

0x0040000e0x00000000\$t614x00000000

0x0040000f0x00000000\$t715x00000000

0x004000100x00000000\$a616x00000000

0x004000110x00000000\$a717x00000000

0x004000120x00000000\$a018x00000000

0x004000130x00000009\$a219x00000009

0x004000140x00000001\$a420x00000001

0x004000150x0000000b\$a521x0000000b

0x004000160x00000000\$a622x00000000

0x004000170x00000000\$a723x00000000

0x004000180x00000000\$t024x00000000

0x004000190x00000000\$t125x00000000

0x0040001a0x00000000\$t226x00000000

0x0040001b0x00000000\$t327x00000000

0x0040001c0x00000000\$t428x00000000

0x0040001d0x00000000\$t529x00000000

0x0040001e0x00000000\$t630x00000000

0x0040001f0x00000000\$t731x00000000

0x004000200x00000000\$a032x00000000

0x004000210x00000000\$a133x00000000

0x004000220x00000000\$a234x00000000

0x004000230x00000000\$a335x00000000

0x004000240x00000000\$a436x00000000

0x004000250x00000000\$a537x00000000

0x004000260x00000000\$a638x00000000

0x004000270x00000000\$a739x00000000

0x004000280x00000000\$t040x00000000

0x004000290x00000000\$t141x00000000

0x0040002a0x00000000\$t242x00000000

0x0040002b0x00000000\$t343x00000000

0x0040002c0x00000000\$t444x00000000

0x0040002d0x00000000\$t545x00000000

0x0040002e0x00000000\$t646x00000000

0x0040002f0x00000000\$t747x00000000

0x004000300x00000000\$a048x00000000

0x004000310x00000000\$a149x00000000

0x004000320x00000000\$a250x00000000

0x004000330x00000000\$a351x00000000

0x004000340x00000000\$a452x00000000

0x004000350x00000000\$a553x00000000

0x004000360x00000000\$a654x00000000

0x004000370x00000000\$a755x00000000

0x004000380x00000000\$t056x00000000

0x004000390x00000000\$t157x00000000

0x0040003a0x00000000\$t258x00000000

0x0040003b0x00000000\$t359x00000000

0x0040003c0x00000000\$t460x00000000

0x0040003d0x00000000\$t561x00000000

0x0040003e0x00000000\$t662x00000000

0x0040003f0x00000000\$t763x00000000

0x004000400x00000000\$a064x00000000

0x004000410x00000000\$a165x00000000

0x004000420x00000000\$a266x00000000

0x004000430x00000000\$a367x00000000

0x004000440x00000000\$a468x00000000

0x004000450x00000000\$a569x00000000

0x004000460x00000000\$a670x00000000

0x004000470x00000000\$a771x00000000

0x004000480x00000000\$t072x00000000

0x004000490x00000000\$t173x00000000

0x0040004a0x00000000\$t274x00000000

0x0040004b0x00000000\$t375x00000000

0x0040004c0x00000000\$t476x00000000

0x0040004d0x00000000\$t577x00000000

0x0040004e0x00000000\$t678x00000000

0x0040004f0x00000000\$t779x00000000

0x004000500x00000000\$a080x00000000

0x004000510x00000000\$a181x00000000

0x004000520x00000000\$a282x00000000

0x004000530x00000000\$a383x00000000

0x004000540x00000000\$a484x00000000

0x004000550x00000000\$a585x00000000

0x004000560x00000000\$a686x00000000

0x004000570x00000000\$a787x00000000

0x004000580x00000000\$t088x00000000

0x004000590x00000000\$t189x00000000

0x0040005a0x00000000\$t290x00000000

0x0040005b0x00000000\$t391x00000000

0x0040005c0x00000000\$t492x00000000

0x0040005d0x00000000\$t593x00000000

0x0040005e0x00000000\$t694x00000000

0x0040005f0x00000000\$t795x00000000

0x004000000x00000000\$zero00x00000000

0x004000010x00000000\$a101x00000000

0x004000020x00000000\$a002x00000000

0x004000030x00000000

posição a memória. Procure usar a versão 3 do algoritmo de multiplicação, pode ser mais simples !!

**Atenção** que, ao multiplicarmos dois números de 32 bits a resposta poderá ser um número de 64 bits, assim a resposta deverá estar contida em dois registradores temporários, um armazenará a parte superior do número e outro a parte inferior, portanto duas posições de memória serão escritas (a terceira e a quarta).

Para os programas a seguir use instruções mult, div, mflo e mfhi.

1. Se tivermos 2 inteiros, cada um com 32 bits, quantos bits podemos esperar para o produto?

- A. 16
- B. 32
- C. 64
- D. 128

R.: Letra C

2. Quais os registradores que armazenam os resultados na multiplicação?

- A. high e low
- B. hi e lo
- C. R0 e R1
- D. \$0 e \$1

R.: Letra B

3. Qual a operação usada para multiplicar inteiros em comp. de dois?

- A. mult
- B. multu
- C. multi
- D. mutt

R.: Letra A

4. Qual instrução move os bits menos significativos da multiplicação para o reg. 8?

- A. move \$8,lo
- B. mvlo \$8,lo
- C. mflo \$8
- D. addu \$8,\$0,lo

R.: Letra C

5. Se tivermos dois inteiros, cada um com 32 bits, quantos bits deveremos estar

**preparados para receber no quociente?**

- A. 16**
- B. 32**
- C. 64**
- D. 128**

**R.: Letra B**

**6. Após a instrução div, qual registrador possui o quociente?**

- A. lo**
- B. hi**
- C. high**
- D. \$2**

**R.: Letra A**

**7. Qual a inst. Usada para dividir dois inteiros em comp. de dois?**

- A. dv**
- B. divide**
- C. divu**
- D. div**

**R.: Letra**

**8. Faça um arithmetic shift right de dois no seguinte padrão de bits: 1001 1011**

- A. 1110 0110**
- B. 0010 0110**
- C. 1100 1101**
- D. 0011 0111**

**R.: Letra A**

**9. Qual o efeito de um arithmetic shift right de uma posição?**

- A. Se o inteiro for unsigned, o shift o divide por 2. Se o inteiro for signed, o shift o divide por 2.**
- B. Se o inteiro for unsigned, o shift o divide por 2. Se o inteiro for signed, o shift pode resultar em um valor errado.**
- C. Se o inteiro for unsigned, o shift pode ocasionar um valor errado. Se o inteiro for signed, o shift o divide por 2.**
- D. O shift multiplica o número por dois.**

**R.: Letra A**

**10. Qual sequência de instruções avalia  $3x+7$ , onde  $x$  é iniciado no reg. \$8 e o resultado armazenado em \$9?**

**A.**

**ori \$3,\$0,3  
mult \$8,\$3  
mflo \$9  
addi \$9,\$9,7**

**B.**

**ori \$3,\$0,3  
mult \$8,\$3  
addi \$9,\$8,7**

**C.**

**ori \$3,\$0,3  
mult \$8,\$3  
mfhi \$9  
addi \$9,\$9,7**

**D.**

**mult \$8,3  
mflo \$9  
addi \$9,\$9,7**

**R.: Letra A**