## Exercício Pratico 06

Aluno: Luís Augusto Starling Toledo Matrícula: 761670

### Parte 1

## 1. O que é um arquivo fonte?

A. um arquivo de texto que contém instruções de linguagem de programação.

#### 2. O que é um registrador?

A. parte do sistema de computador que mantém o controle dos parâmetros do sistema.

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

A. #

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

C. 32

## 5. O que é o contador de programa?

D. parte do processador que contém o endereço da próxima instrução de máquina para ser obtida.

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

C. 4

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

D. uma declaração que diz o montador algo sobre o que o programador quer, mas não corresponde diretamente a uma instrução de máquina.

## 8. O que é um endereço simbólico?

D. um nome usado no código-fonte em linguagem assembly para um local na memória.

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

B. 0x00400000

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

A. operando imediato

- 11. Como é chamada uma operação lógica executada entre bits de cada coluna dos operandos para produzir um bit de resultado para cada coluna? B. operação bitwise
- 12. Quando uma operação é de fato executada, como estão os operandos na ALU?
- D. Cada um dos registradores deve possuir 32 bit.
- 13. Dezesseis bits de dados de uma instrução de ori são usados como um operando imediato. Durante execução, o que deve ser feito primeiro?

  B. Os dados são estendidos em zero à esquerda por 16 bits.
- 14. Qual das instruções seguintes armazenam no registrador \$5 um padrão de bits que representa positivo 48? C. ori \$5,\$0,48
- 15. A instrução de ori pode armazenar o complemento de dois de um número em um registrador?
  B. Sim.
- 16. Qual das instruções seguintes limpa todos os bits no registrador \$8 com exceção do byte de baixa ordem que fica inalterado?

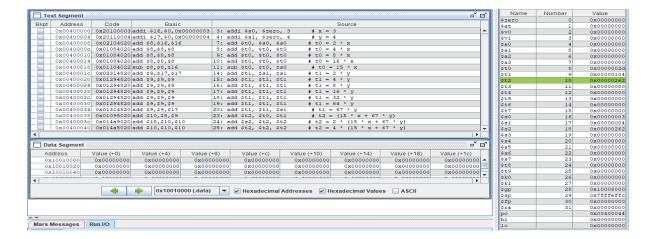
  D. andi \$8,\$8.0xFF
- 17. Qual é o resultado de um ou exclusivo de padrão sobre ele mesmo? A. Todos os bits em zero.
- **18.** Todas as instruções de máquina têm os mesmos campos? A. Não. Diferentes de instruções de máquina possuem campos diferentes.

## Parte 2

## Implementar em MIPS/MARS os seguintes programas

```
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;
}
  # a -> $s0
2
  # b -> $s1
 3
 4
  # c -> $s2
 5
  # d -> $33
  # x -> $s4
   # y -> $35
10 # inicio
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 add $t0, $s0, $s1 # a + b
16 add $t1, $s2, $s3 # c + d
17 sub $s4, $t0, $t1 # (a + b) - (c + d)
18 sub $t0, $s0, $s1 # a - b
19 add $s5, $t0, $s5 # (a - b) + 5
20 sub $s1, $s4, $s5 # b = x - y
21 # fim
                                                                           o" ⊠"
Text Segment
                                                             □ Labels
 0x000000000
0x000000000
                                                                                                       0x000000000
0x000000002
                                                                   ✓ Data
✓ Text
   0x00000000
0x00400028
 Clear -- program is finished running (dropped off bottom) --
```

```
//programa 2 (add, addi, sub, lógicas) {
                                                    x = 1;
                                                    y = 5*x + 15;
  }
         Edit Execute
          mips1.asm
         2 # Inicio
                     addi $s4, $zero, 1
                     add $t0, $s4, $s4
                                                                                                                                    \# \text{ temp} = x + x (2 * x)
                                                                                                                                   # temp = temp + x (3 * x)
# temp = temp + x (4 * x)
# temp = temp + x (5 * x)
                      add $t0, $t0, $s4
                     add $t0, $t0, $s4
                     add $t0, $t0, $s4
                                                                                                                                    # y = 5 * x + 15
                    addi $s5, $t0, 15
    Text Segment
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  o" Ø"
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    0x00000000
0x00000000
0x00000000
      | Bkpt | Address | Code | Basic | Standard |
                                                                                                                                                                                                                                                                                                              # x = 1
                                                                                                                                                                                                                                                                                                             # x = 1
# temp = x + x (2 * x)
# temp = temp + x (3 * x)
# temp = temp + x (4 * x)
# temp = temp + x (5 * x)
# temp = temp + x (5 * x)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    0x00000000
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    0x00000000
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    0x00000000
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     0x00000000
      Data Segment
                 Address
                                                                      Value (+0)
                                                                                                                              Value (+4)
                                                                                                                                                                                   Value (+8)
                                                                                                                                                                                                                                         Value (+c)
                                                                                                                                                                                                                                                                                             Value (+10)
                                                                                                                                                                                                                                                                                                                                                    Value (+14)
                                                                                                                                                                                                                                                                                                                                                                                                           Value (+18)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                Value (+1c)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    0x00000000
                   0x10010
                                                                                                                                                                                     0x00000000
                                                                                                                                                                                                                                           0x00000000
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   $82
                                                                                                                                                                                                                                            0x00000000
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  0x0000000
                                                                                                                               0x00000000
0x000000000
                                                                                                                                                                                                                                                                                                                                                          0x00000000
0x000000000
                                                                                                                                                                                                                                                                                                                                                                                                                                                                       0x0000000
                                                                                                                                                                                       0x00000000
0x000000000
                                                                                                                                                                                                                                           0x000000000
                                                                                                                                                                                                                                                                                                    0x000000000
                                                                                                                                                                                                                                                                                                                                                                                                                0x00000000
0x000000000
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   $87
$t8
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        23
24
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    0x00000000
                                                                                                                                0x00000000
                                                                                                                                                                                                                                                                                                                                                                                                                                                                       0x00000000
                                                                                                                                                                                       0x00000000
                                                                                                                                                                                                                                            0x00000000
                                                                                                                                                                                                                                                                                                     0x00000000
                                                                                                                                                                                                                                                                                                                                                                                                                  0x00000000
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   $k0
$k1
$gp
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    0x000000000
0x000000000
0x10008000
                                                                                 (a) | (a) |
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    $sp
$fp
      Mars Messages Run I/O
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    0x00000000
0x00400018
      Clear
 // programa 3 (add, addi, sub, lógicas) {
                                                   x = 3;
                                                    y = 4;
                                                    z = (15*x + 67*y)*4
   }
      4 addi $sl. $zero, 4
     7 addi $t0, $s0, $s0
8 addi $t0, $t0, $t0
                                                                                                                   # t0 = 2 * x
# t0 = 4 * x
# t0 = 8 * x
               addi $t0, $t0, $t0
addi $t0, $t0, $t0
                                                                                                                    # t0 = 16 * x
# t0 = 15 * x
  11
                sub $t0, $t0, $s0
  13 # 67 * y
                                                                                                                   # t1 = 2 * y
# t1 = 4 * y
# t1 = 8 * y
# t1 = 16 * y
# t1 = 32 * y
# t1 = 64 * y
                addi $t1, $s1, $s1
addi $t1, $t1, $t1
  15
               addi $t1, $t1, $t1
addi $t1, $t1, $t1
 18 addi $t1, $t1, $t1
19 addi $t1, $t1, $t1
  20
                 add $tl, $tl, $sl
 22 # (15 * x + 67 * y)
                                                                                                                   # t2 = (15 * x + 67 * y)
# t2 = 2 * (15 * x + 67 * y)
# t2 = 4 * (15 * x + 67 * y)
23
24
               add $t2, $t0, $t1
addi $s2, $t2, $t2
25 addi $t2, $t2, $t2
26 # Fim
27
```



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

```
// programa 4 {
                              x = 3;
                              y = 4;
                              z = (15*x + 67*y)*4
 }
     mips1.asm
                      ori $sl, $zero, 4
                     # Calcule 15 * x usando sll
                     sl1 $t2, $s0, 3
sl1 $t3, $s0, 2
                     sl1 $t4, $s0, 1
add $t5, $t2, $t3
                                                                       # t4 = x * 2
                                                                   # t4 = x ^ 2

# t5 = x * 8 + x * 4

# t5 = (x * 8 + x * 4) + x * 2

# t5 = (x * 8 + x * 4 + x * 2) + x = 15 * x
 10
11
12
13
14
15
16
17
18
19
20
21
22
                     add $t5, $t5, $t4
                     # Calcule 67 * v usando sll e soma
                     sll $t6, $sl, 6
sll $t7, $sl, 1
add $t8, $t6, $t7
                                                                      mao sil e soma

# t6 = y * 64

# t7 = y * 2

# t8 = y * 64 + y * 2

# t8 = (y * 64 + y * 2) + y = 67 * y
                     add $t8, $t8, $sl
                     # Calcule 15 * x + 67 * y
add $t9, $t5, $t8  # t9 = 15 * x + 67 * y
                     # Multiplique o resultado por 4 usando sll
                                                                                                                                                                                                                                                                                            ם ֿוֹם
   ori $s0, $zero, 3 # x = 3
                                                                                                                                        o* Ø"
   Data Segment

        Address
        Value (+0)
        Value (+4)
        Value (+8)
        Value (+c)
        Value (+10)
        Value (+14)
        Value (+18)
        Value (+1c)

        0x10010000
        0x00000000
        0x00000000

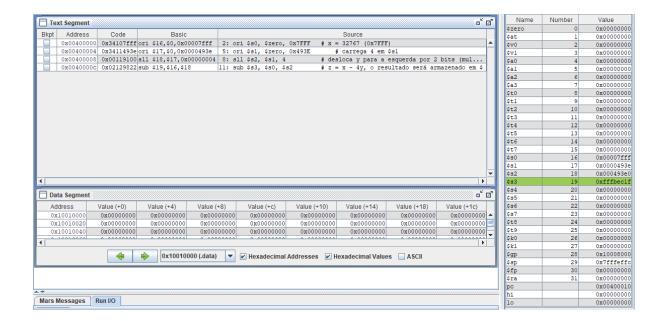
        Value (+c)
        Value (+10)
        Value (+14)
        Value (+18)
        Value (+1c)

        0x00000000
        0x00000000
        0x00000000
        0x00000000
        0x00000000

        Address
```

```
// programa 5 {
          x = 100000;
          y = 200000;
          z = x + y;
}
   .text
    ori $s0, $zero,0x186A
2
3
    sll $s0, $s0, 4
    ori $sl, $zero,0x30D4
4
   sl1 $s1, $s1, 4
5
    add $s2, $s0,$s1
6
                                                                                                  oʻ 🛮
                                                                                                          Name Number
 Text Segment
 Source
                                                                                                                             0x00000000
                                                                                                                             0x00000000
0x00000000
0x00000000
  0x0040000c 0x00118900 sll $17,$17,0x00000004 5: sll $sl, $sl, 4
0x00400010 0x02119020 add $18,$16,$17 6: add $s2, $s0,$sl
                                                                                                         $a3
$t0
                                                                                                                             0x00000000
                                                                                                                            0x000000000
                                                                                                         $t5
$t6
                                                                                                         $t7
$s0
                                                                                                                             0x000186a0
                                                                                                                             0x00030d40
0x000493e0
 Data Segment
                                                                                                  o" 🗹
                                                                                                         $85
$86
                                                                                                                             0x00000000
                                             Value (+c)
0x000000000
                                                         Value (+10) Value (+14) 0x000000000 0x00000000
   Address
              Value (+0)
                         Value (+4)
                                    Value (+8)
                                                                                Value (+18)
                                                                                           Value (+1c)
                                                                                                                             0x00000000
                                                                                                         St8
                                                                                                                             0x0000000
                $gp
                                                                                                         $sp
$fp
                                                                                                                             0x00000000
0x00400014
 Mars Messages Run I/O
```

```
// programa 6 { 
 x = 0 maior inteiro possível; 
 y = 300000; 
 z = x - 4y }
```

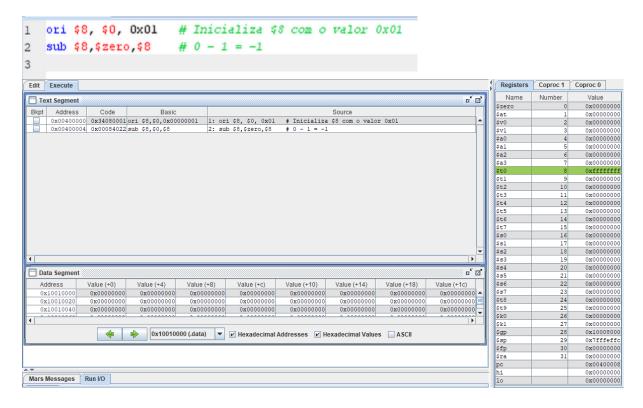


Considere a seguinte instrução iniciando um programa:

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 = 0xFFFFFFF



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

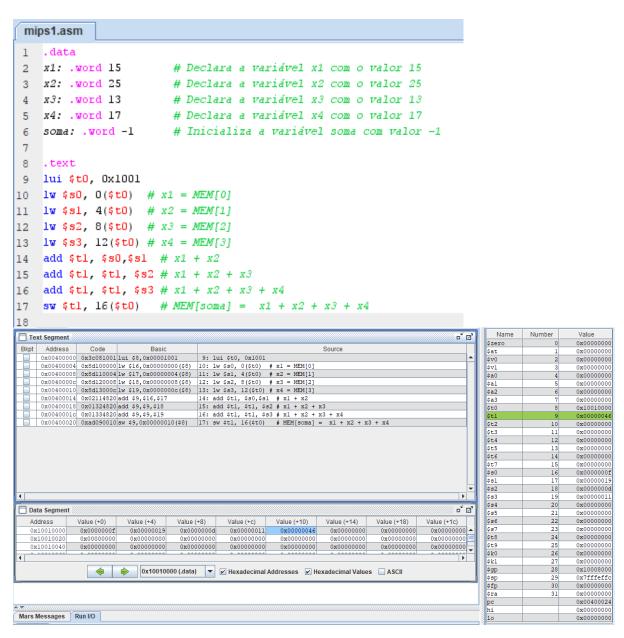
$12 = 0x78
```

```
mips1.asm
                # atribuir $8 = 0x12345678
                  ori $8, $zero, 0x1234
                   sll $8,$8, 16
    3
                    ori $8, $8,0x5678
    4
    5
    6
                   # obtem $9 = 12
    7
                   srl $9,$8,24 # $9 = 12
    8
                  # obtem $10 = 34
    9
                  srl $10, $8, 16
10
                    andi $10, $10, 0x00FF
11
12
                 # obtem $11 = 56
13
14
                 srl $11, $8, 8
                   andi $11, $11, 0x00FF
15
16
17
                  # obtem $12 = 78
                   andi $12, $8, 0x00FF
18
19
 Text Segment
                                                                                                                                                                                                                                                                                                                   0x00000000
0x00000000
0x00000000
  Bkpt Address
                                                                                                                                                                                                                      Source
                Address Code Basic 2:00158,520,000001234 2: 01158,52ero, 0x1234 0x00400000 0x34081234 0r1 58,50,0000001234 2: 01158,58,16 0x00400004 0x00084400 sl1 68,68,000000010 3: sl1 68,68, 16 0x00400000 0x00085678 or1 58,68,000000018 7: sr1 59,59,24 # 59 = 12 0x00400010 0x00085402 sr1 59,58,000000018 7: sr1 59,59,24 # 59 = 12 0x00400014 0x1400055678 0x1 610,58,000000018 11: and 510, 58, 16 0x00400014 0x14400055678 0x10400014 0x1400055678 0x10400014 0x14005578 0x10400014 0x14005578 0x10400014 0x14005578 0x10400014 0x1040578 0x104
                                                                                                                                                                                                                                                                                                                                                                                                       0x00000000
                                                                                                                                                                                                                                                                                                                                                                                                       0x00000000
                                                                                                                                                                                                                                                                                                                                                                                                        0x12345678
                0x0040001c 0x316b00ff andi $11,$11,0x0000... 15: andi $11, $11, 0x00FF 0x00400020 0x310c00ff andi $12,$8,0x000000ff 18: andi $12,$8,0x000000ff
                                                                                                                                                                                                                                                                                                                                                                                                       0x00000012
                                                                                                                                                                                                                                                                                                                                        $t4
                                                                                                                                                                                                                                                                                                                                         $s1
$s2
                                                                                                                                                                                                                                                                                                                                                                                                       0x00000000
                                                                                                                                                                                                                                                                                                                                                                                                       0x00000000
0x00000000
0x00000000
  Data Segment
         Address
                                          Value (+0)
                                                                           Value (+4)
                                                                                                               Value (+8)
                                                                                                                                                  Value (+c)
                                                                                                                                                                                   Value (+10)
                                                                                                                                                                                                                       Value (+14)
                                                                                                                                                                                                                                                          Value (+18)
                                                                                                                                                                                                                                                                                             Value (+1c)
                                                                               0x00000000
                                                                                                                                                    0x00000000
                                                                                                                                                                                                                                                                                                                                         $87
$t8
                                                                                                                                                                                                                                                                                                                                                                                                       0x000000000
                                                                                                                                                                                                                                                                                                                                         $k0
                                                                                                                                                                                                                                                                                                                                                                                                      0x00000000
                                                                                                                                                                                                                                                                                                                                                                                                     0x00000000
0x10008000
0x7fffeffc
0x000000000
                                                   $sp
$fp
                                                                                                                                                                                                                                                                                                                                                                                                      0x00000000
0x00400024
                                                                                                                                                                                                                                                                                                                                                                                                        0x00000000
0x00000000
 Mars Messages Run I/O
```

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

```
// 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.



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.

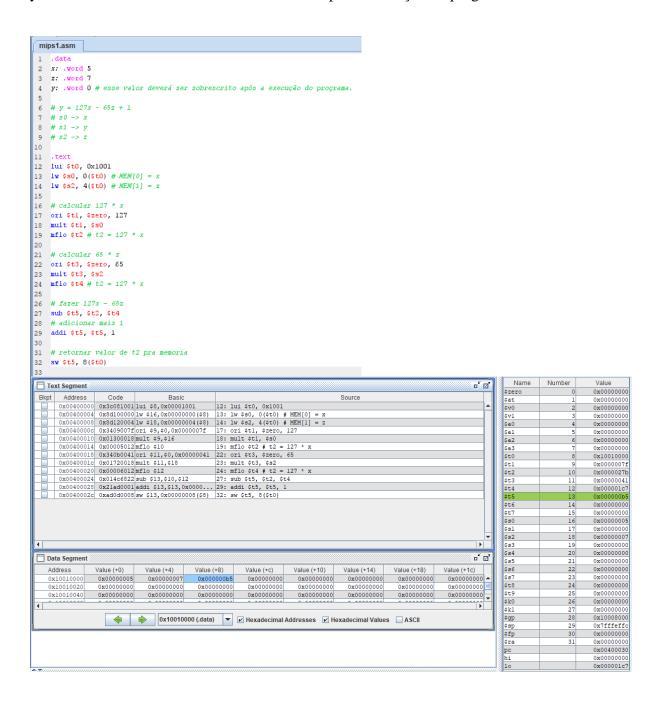
estão armazenados na memória e, na posição imediatamente a seguir, o valor de y deverá ser escrito, ou seja:

.data

x: .word 5

z: .word 7

y: .word 0 # esse valor deverá ser sobrescrito após a execução do programa.



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.

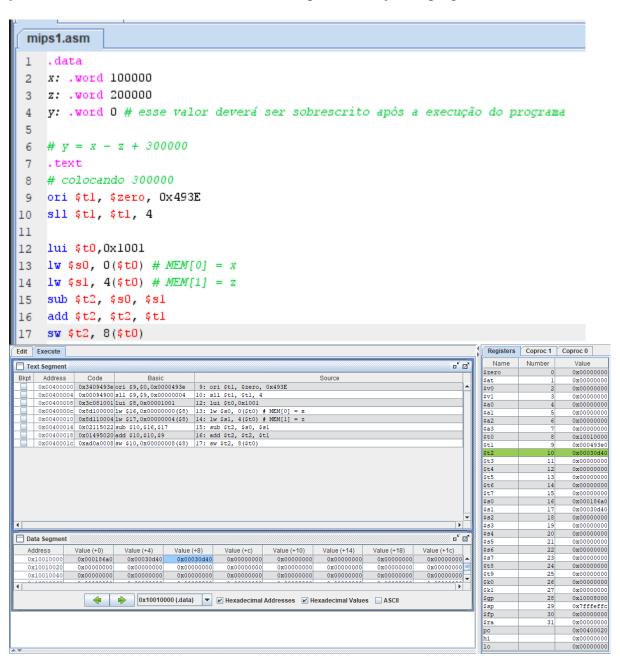
estão armazenados na memória e, na posição imediatamente a seguir, o valor de y deverá ser escrito, ou seja:

#### .data

x: .word 100000

z: .word 200000

y: .word 0 # esse valor deverá ser sobrescrito após a execução do programa.



```
// programa 12
```

Considere a seguinte situação:

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

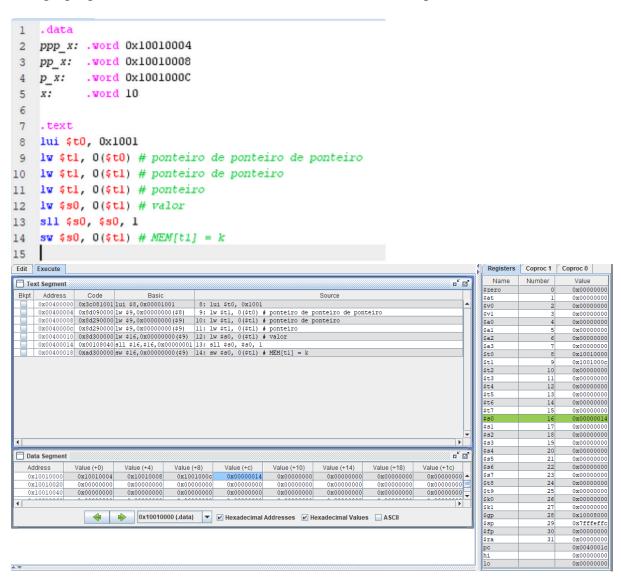
onde x contem um ponteiro para um ponteiro para um ponteiro para um inteiro.

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 = MEM [MEM [MEM [x]]].
```

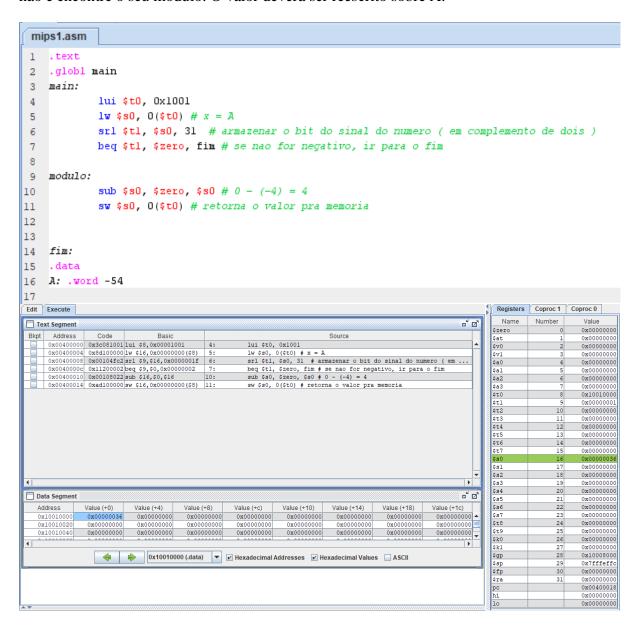
Crie um programa que implemente a estrutura de dados acima, leia o valor de K, o multiplique por 2 e o reescreva no local correto conhecendo-se apenas o valor de x.



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

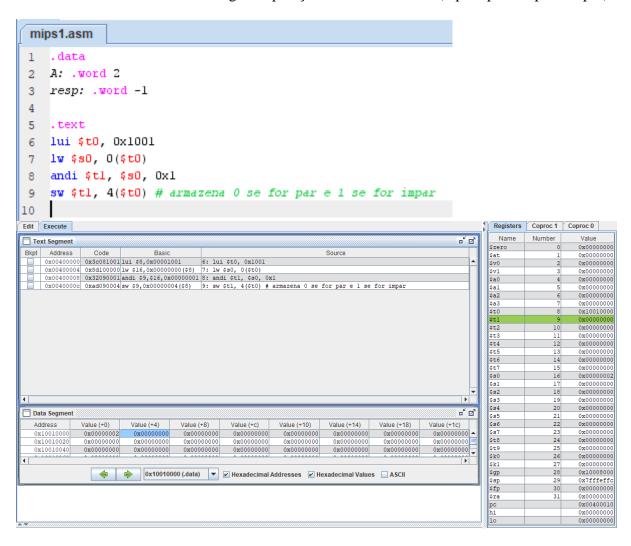
#### // programa 13:

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.



#### // 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).



#### // programa 15:

Escrever um programa que crie um vetor de 100 elementos na memória onde vetor[i] = 2\*i +1.

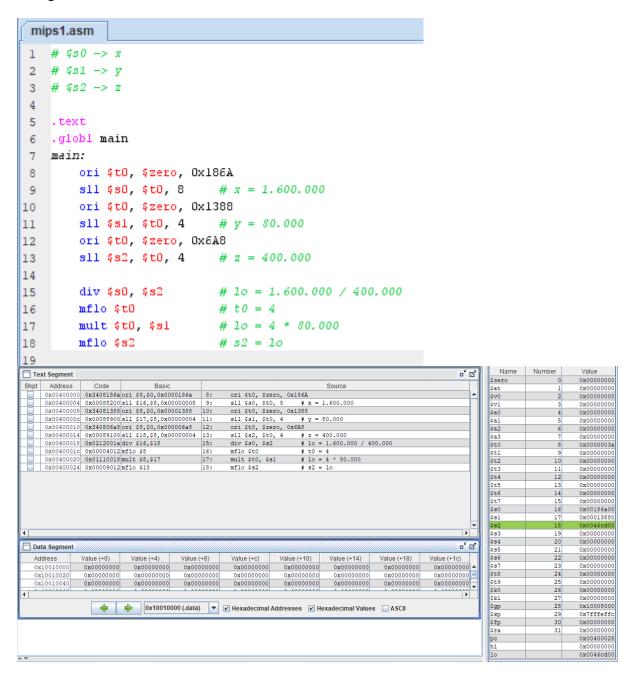
Após a última posição do vetor criado, escrever a soma de todos os valores armazenados do vetor.

Use o MARS para verificar a quantidade de instruções conforme o tipo (ULA, Desvios, Mem ou Outras)



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

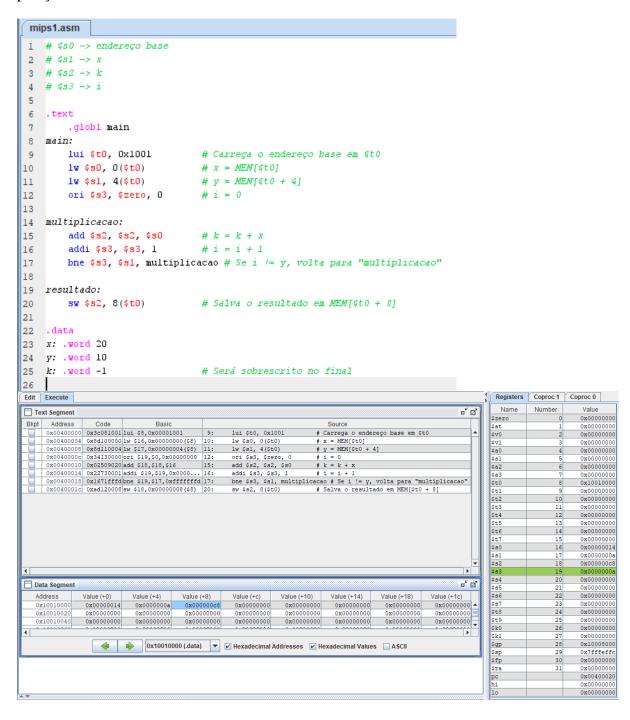
Use x = 1600000 (=0x186A00), y = 80000 (=0x13880), e z = 400000 (=0x61A80). Inicializar os registradores com os valores acima.



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.



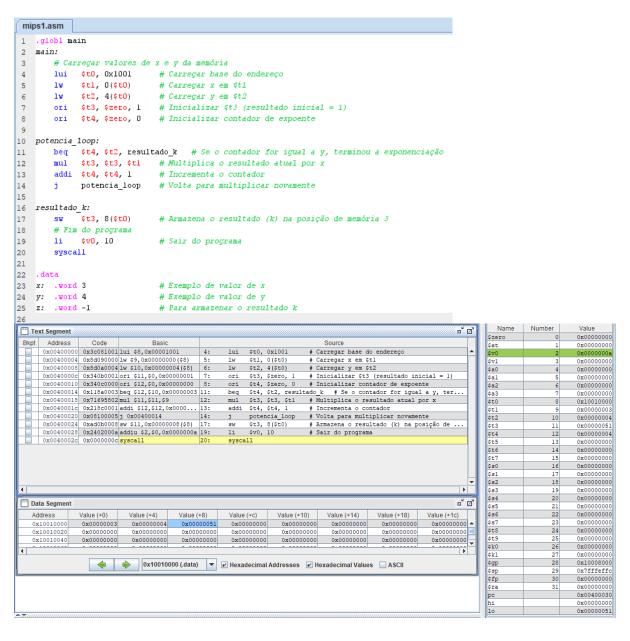
Para a expressão a seguir, escreva um programa que calcule o valor de k:

k = xy

Obs: Você poderá utilizar o exercício anterior.

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.

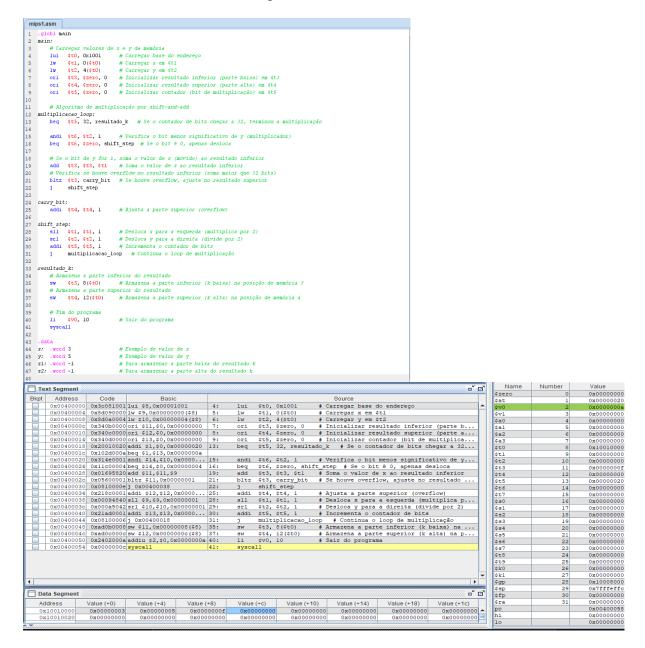
Dê um valor para x e y (dê valores pequenos !!) e use o MARS para verificar a quantidade de instruções conforme o tipo (ULA, Desvios, Mem ou Outras)



#### Desafio:

Todos viram durante a parte aritmética que podemos utilizar a ULA e registradores para multiplicar dois números através de 3 algoritmos.

Você deverá escrever um programa que leia dois números da memória (primeira posição e segunda posição) os multiplique e coloque o resultado na terceira 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 reposta 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).



## Parte 3

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

C. 64

- 2. Quais os registradores que armazenam os resultados na multiplicação? B. hi e lo
- 3. Qual a operação usada para multiplicar inteiros em comp. de dois?
- 4. Qual instrução move os bits menos significativos da multiplicação para o reg. 8?

C. mflo \$8

- 5. Se tivermos dois inteiros, cada um com 32 bits, quantos bits deveremos estar preparados para receber no quociente? B. 32
- 6. Após a instrução div, qual registrador possui o quociente? A. lo
- 7. Qual a inst. Usada para dividir dois inteiros em comp. de dois? D. div
- 8. Faça um arithmetic shift right de dois no seguinte padrão de bits: 1001 1011

B. 0010 0110

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.

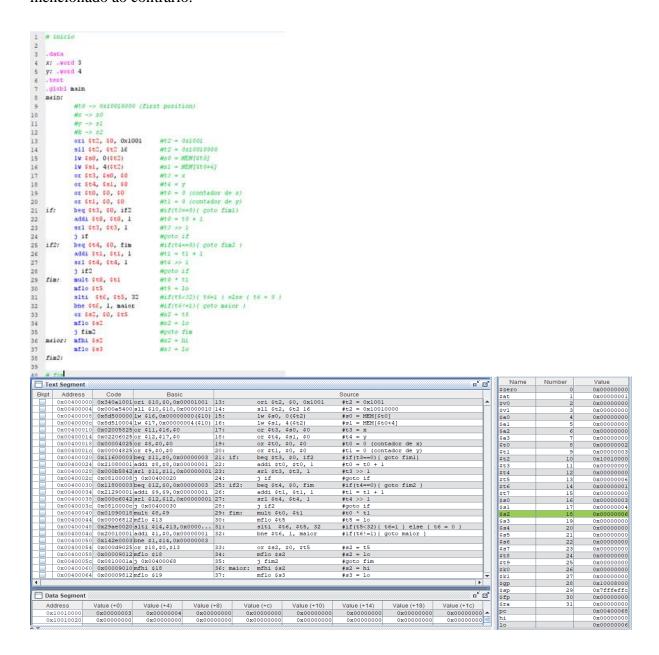
10. Qual sequencia de instruções avalia 3x+7, onde x é iniciado no reg. \$8 e o resultado armazenado em \$9?

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

#### Parte 4

#### // programa 19

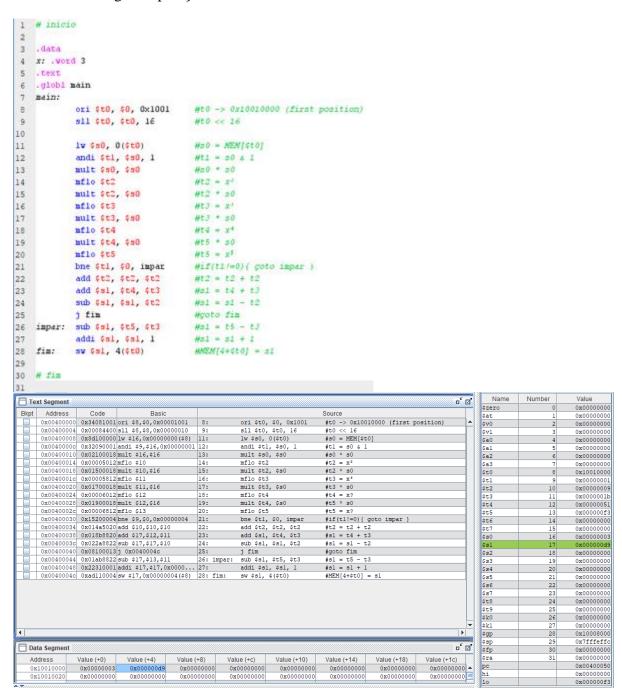
Escrever um programa que leia dois números da memória, a primeira e segunda posições respectivamente (os coloque em \$s0 e \$s1) e determine a quantidade de bits significantes de cada um. Coloque as respostas em \$t0 e \$t1, a partir desse resultado faça a multiplicação. Caso o número de bits significantes de ambos seja menor do que 32 a resposta deverá estar apenas em \$s2, caso contrário a resposta estará em \$s2 e \$s3 (LO e HI respectivamente). Para os exercícios a seguir, considere as variáveis com números abaixo de 16 bits, salvo se mencionado ao contrário.



```
// programa 20

y = \begin{cases} x^4 + x^3 - 2x^2 & \text{se x for par} \\ x^5 - x^3 + 1 & \text{se x for impar} \end{cases}
```

Os valores de x devem ser lidos da primeira posição livre da memória e o valor de y deverá ser escrito na segunda posição livre.



$$y = \begin{cases} x^3 + 1 & \text{se } x > 0 \\ x^4 - 1 & \text{se } x \le 0 \end{cases}$$

Os valores de x devem ser lidos da primeira posição livre da memória e o valor de y deverá ser escrito na segunda posição livre.

