	**Fundação Universidade Federal do Vale do São Francisco** **Colegiado de Engenharia da Computação** **Disciplina:** ***Organização e Arquitetura de Computadores*** **I** **Prof.:** ***Rômulo Calado P. Camara*** **Lista de Exercícios** **(Programação MIPS)**

1. Faça um estudo do MIPS destacando os seguintes aspectos:
a. tamanho do dado a ser processado
b. espaço de endereçamento de memória
c. número de registradores
d. formato e tamanho das instruções

2. Tanto na arquitetura projetada em sala de aula como na descrita no livro todas as instruções aritméticas envolvem três operandos (registradores) e possuem o mesmo formato. Na sua opinião de projetista, quais as vantagens e desvantagens desta decisão de projeto?

3. Porque o processador MIPS se caracteriza como uma arquitetura LOAD/STORE? Quais as vantagens e desvantagens desta característica?

4. O que é modo de endereçamento e quais os modos de endereçamento das instruções do MIPS?

5. Qual o suporte do MIPS para funções? Existe suporte para recursividade? Como?

6 . Considere a seguinte parte de programa em linguagem Assembly do MIPS:

```
.data 0x10010000      # segmento de dados
palavra1:  .word 13
palavra2:  .word 0x15
```

Indique, em hexadecimal, quais os valores dos seguintes itens:
Palavra1:
Palavra2:

7. Faça cada programa em um arquivo separado, com nome dado por exercícionumero_questão.s. Exemplo exercicio-1.s

7.1. Codifique um programa correspondente ao seguinte pseudo-código:

```
int a = 3;
int b = 4;
int m = 10;
m = a;
if ( b < m )
m = b;
```

7.2. Codifique um programa correspondente a

```
int a = ...;
int b = ...;
x = 0;
if ( a >= 0 && b <= 50 )
x = 1;
```

7.3. Faça um programa em linguagem de montagem MIPS que receba como entrada uma string com n caracteres e gere como saída uma nova string com a inversão da ordem dos caracteres. Essa nova string também terá a troca das letras maiúsculas por minúsculas e vice-versa. Por exemplo: se a entrada for HArdwArE a saída deverá ser eRaWDRah. A entrada deve ser lida da memória e a saída deve ser escrita na memória. Caso a string possua algum caractere que não seja letra o valor 1 deverá ser armazenado no registrador v1 e o programa deverá ser encerrado. Lembramos que o fim da string é dado pelo caractere nulo e que para manipular string nessa questão recomendamos que sejam usadas variáveis do tipo ASCIIZ.
Atenção: os caracteres deverão ser armazenados em sequência na memória.

7.4. Escreva um programa em linguagem de montagem do MIPS que receba dois números inteiros armazenados na memória e realiza a multiplicação dos dois números. Considere números positivos e negativos. A instrução **mult** não deverá ser utilizada na implementação dessa questão. O resultado deverá ser armazenado em uma variável RESULT na memória.

7.5. Faça um programa em linguagem de montagem MIPS que receba como entrada dois números, n e s, e que tenha como saída o resultado da combinação de n tomados s a s. Os números n e s devem ser carregados da memória e o resultado da combinação deve ser colocado na mesma em uma variável COMB. Caso s seja maior que n, deve ser armazenado o valor 1 no registrador v1. Caso n e/ou s seja menor que zero, o valor 2 deve ser armazenado no registrador v1. Quando n for igual a s o valor 3 deve ser armazenado no registrador V1. Quando n e/ou s for igual a zero o valor 4 deve ser armazenado no registrador v1. Segue abaixo a fórmula da combinação.

$$C_n^s = \frac{n!}{s! \cdot (n-s)!}$$

Atenção: é obrigatório o uso de recursão para a implementação do cálculo de n!, (n - s)! e s!. Sendo que a questão que não for feita recursivamente não será aceita como resposta válida.

7.6. Escreva um programa em linguagem de montagem do MIPS que converta um número inteiro (representação complemento a dois) na sua representação em ASCII. Números positivos e negativos devem ser considerados. O número binário terá 32 caracteres e será lido da memória. Os números estarão representados como string de caracteres ASCII e o fim de string será denotado com o caractere null. O número binário deverá ser armazenado no registrador v0. Caso o número a ser convertido não seja válido, o registrador v1 retornará o valor 1.

7.7. Complete o seguinte código: escreva um procedimento em Assembly do MIPS que percorra o vetor de bytes **vetorX** e conte o número de bytes cujo valor é igual a 1. Armazene o resultado da contagem na variável **numerosDe1**. (Pode assumir que a dimensão do vetor sala é sempre 64 bytes).

```
.data  # declara segmento de dados
vetorX: .space 64 # variável que representa o conteúdo de um vetor
numerosDe1: .word 0  # variável que conta o número de 1
#
# Princípio do Segmento de Texto
#
.text # declara segmento de código
main:
```

7.8 . Pretende-se codificar um procedimento Substitui(string,x,y), em Assembly do MIPS, que dada uma string e dois caracteres, x e y, substitui nessa string todas as ocorrências do caracter x pelo caracter y. O procedimento terá como argumentos:

• string: o endereço da string
• x: o caractere a procurar
• y: o caractere para a substituição
Por exemplo, para a string “Sobstitoi”, se o caracter a procurar for o caracter ‘o’, e o caracter para a substituição for o caracter ‘u’, a string deve ser alterada para “Substitui”.

a. Desenhe um fluxograma para este procedimento.
b. Escreva o programa em Assembly do MIPS.

7.9 Traduza para o assembly do MIPS os código a seguir:
a)

```
int i;
int A[10];

for (i=0; i<10; i++) {
    A[i]=A[i]+1;
}
```

b)

```
int i;
int A[10];

for (i=0; i<10; i++) {
    if (i%2==0)
        A[i]=A[i]+A[i+1];
    else
        A[i]=A[i]*2;

}
```

Atenção: o código de todas as questões deverá estar claramente comentado, pois caso contrário a correção será muito dificultada. Portanto colaborem com a correção!

Os exercícios de 1 até 10 valerá 2,5 pontos.
Obs: Os exercício 6.1 deve ser entregue no mesmo dia da aula (por email).

Os demais exercícios deverão ser entregues até o dia 28 de Fevereiro de 2014.

Enviar exercícios para os endereços:

romulo.camara@univasf.edu.br