Universidade Federal Rural de Pernambuco
Departamento de Computação
Disciplina Arquitetura e Organização de Computadores
Professor: Andson M Balieiro.
Pontuação: 2,0 pontos
Entrega: Submissão pelo AVA até 23:55 do dia 17/11/2018
Entregáveis: códigos-fonte das questões em arquivo compactado (os códigos-fonte devem ser nomeados da seguinte forma: "questaoX.asm", onde "X" indica o número da questão. Ex. "questao1.asm", "questão2.asm",etc). Além disso, submeter um arquivo de texto indicando os nomes dos integrantes do grupo.
Equipe: máximo 3 alunos (1 submissão por grupo).
Obs.:

- **Grupos com nenhum aluno inscrito no AVA da disciplina deve enviar os entregáveis por email: andson.balieiro@ufrpe.br**
- **O peso de cada questão está indicado entre parênteses.**

Exercícios MIPS - Simulador MARS
Download do MARS em http://courses.missouristate.edu/kenvollmar/mars/

Obs. Observe na janela "Execute" do Simulador se os valores estão armazenados na posição correta na memória e na aba de Registradores os valores deles estão de acordo com as instruções executadas nos programas elaborados.

1. (0,5 pt) Faça um programa que carregue o valor **0x10010000** para o registrador **$s0** (endereço base do vetor) vetor e salve o seguinte vetor **v= [1,3,2,1,4,5]** em memória, isto é, que contenha a seqüência de instruções MIPS correspondentes às instruções C seguintes **(obs. não use variáveis)**

```
v[0] = 1;
v[1] = 3;
v[2] = 2;
v[3] = 1;
v[4] = 4;
v[5] = 5;
```

2. (0,5 pt) Construa um programa que tenha acesso à memória de forma indexada. Isto é, carregue o valor **0x10010020 no registrador $s0 (**registrador com o endereço base do vetor) e que contenha a seqüência de instruções MIPS correspondentes às instruções C abaixo. Armazene o valor da variável **i** no registrador **$s1** e de **num** em **$s2**

```
num =100;
  i = 0;
v[i] = 1;
i = i + 1;
v[i] = 3;
i = i + 1;
v[i] = 2;
i = i + 1;
v[i] = 1;
i = i + 1;
v[i] = 4;
i = i + 1;
v[i] = 5;
  i=i+1;
```

```
v[i]=v[2]+v[4]+ num;
```

3. (1,0pt) Faça um programa que, considerando o vetor da questão anterior faça:
a) Substitua todos os elementos pelo seu dobro.
b) Copie os elementos do **vetor *v*** para outro vetor **u,** cujo primeiro elemento tem o endereço **0x10010040**.

4. (0,5pt) Implemente em MIPS assembly para o seguintes código-fonte:

```
a = 1;
b = 2;
for (i = 0; i<10; i++){
      if (a < b){
            a = a + 2;
      }else{
            b = b + 2;
      }
}
```

5. (1,0 pt) Faça um programa MIPS que implemente o seguinte programa C.

```
#include <stdio.h>

main(void)
{
  int num;

  printf("Entre com um inteiro: ");
  scanf("%d", &num);

  /* Imprime uma mensagem dizendo se o numero e positivo ou
     negativo, positivo ou negativo. */
  if (num >= 0) {
    if (num % 2 == 0)
      printf("O numero e par e positivo\n");
    else
      printf("O numero e impar e positivo\n");
  }
  else {
    if (num % 2 == 0)
      printf("O numero e par e negativo\n");
    else
      printf("O numero e impar e negativo\n");
  }
}
```

6. (1,0 pt) Faça um programa MIPS para uma calculadora simples (4 operações básicas). O programa deve receber dois números do usuário (via console) e o código da operação aritmética a ser realizada (1-Adição, 2-Subtração, 3-Multiplicação, 4-Divisão) e imprimir o resultado da operação no console.
Obs. O programa deve usar switch/case com a abordagem de **jump table**.

7. (0,5 pt) Escreva o código C correspondente ao seguinte código MIPS abaixo. Considere que as variáveis **f, g, h, i**, e **j** são associadas aos registradores **$s0, $s1, $s2, $s3** e **$s4**, respectivamente, e que endereço base dos vetores A e B estão nos registradores **$s6** e $s7, respectivamente.

```
sll $t0, $s0, 2        # $t0 = f * 4
```

```
add $t0, $s6, $t0       # $t0 = &A[f]
sll $t1, $s1, 2         # $t1 = g * 4
add $t1, $s7, $t1       # $t1 = &B[g]
lw $s0, 0($t0)          # f = A[f]
addi $t2, $t0, 4
lw $t0, 0($t2)
add $t0, $t0, $s0
sw $t0, 0($t1)
```

8. (1,5 pt) Faça um programa que:

- Leia 10 valores inteiros inseridos pelo usuário
- Armazene os valores lidos em um vetor em memória.
- Ordene o vetor em ordem crescente utilizando o método Bubble Sort.
- Calcule a soma dos elementos do vetor.
- Imprima o vetor ordenado e a soma dos elementos
- Obs: **Faça uso de laço de repetição para leitura e escrita dos valores do vetor ao usuário e soma dos valores do vetor.**

9. (1,5 pt) Faça um programa MIPS que receba um número inteiro do usuário no procedimento **main** e calcule o seu fatorial de forma iterativa (direta) e imprima o valor do fatorial no console. Para isso, o programa deve possuir um procedimento **fatorial** que receba o valor de **n** inserido pelo usuário através de chamada feita no procedimento **main**. Antes da chamada ao procedimento fatorial, verifique se o número inserido é negativo. Caso afirmativo, indicar ao usuário "Entrada Inválida". Finalize o programa com uma chamada de sistema (syscall)

10. (2,0 pt) Faça um programa em MIPS que receba um número inteiro **n** através do console e tenha um procedimento recursivo **soma** que calcula o somatório dos **n** primeiros números inteiros positivos e imprima ao usuário. Caso o número informado seja negativo, indicar mensagem ao usuário "Entrada Inválida". Finalize o programa com uma chamada de sistema (syscall). Como no item anterior, crie um procedimento **main** para receber, checar o valor de entrada e imprimir o resultado. Além de chamar o procedimento **soma**.