UNIVALI – CTTMAR Disciplina: Arquitetura e Organização de Computadores Prof. Douglas Rossi de Melo
Avaliação 03 – Programação de Procedimentos

Instruções:

1. Esta atividade pode ser realizada individualmente ou em grupo de **até DOIS** alunos.

2. A atividade consiste na implementação de um procedimento utilizando a linguagem de montagem do MIPS (conforme as instruções a seguir).

3. Deve ser postado um relatório, com uma capa identificando a Instituição, o curso, a disciplina, o professor, o nome da atividade, os autores do trabalho e a data em que o mesmo for entregue. O corpo do relatório deverá conter a resolução dos exercícios, incluindo: **código-fonte em linguagem de alto nível** (preferencialmente C ou C++), **código-fonte em linguagem de montagem do MIPS**, **quadro de análise das instruções** e **capturas de tela** que ***demonstrem claramente a execução correta das entradas e saídas realizadas via console do simulador e os resultados da execução dos programas***.

4. O código-fonte deve ser escrito em um arquivo com extensão .asm e, **obrigatoriamente**, com um nome que identifique os membros do grupo e o nome do programa (ex. *RingoStarr_JohnLennon_Programa_02.asm*). Obs: Todos os arquivos de todos os grupos serão reunidos em uma mesma pasta e, por isso, não use nomes como Programa_02.asm.

5. O código fonte deve conter um cabeçalho comentado identificando a disciplina, a atividade, o programa e os nomes dos membros do grupo. Ex:

```
# Disciplina: Arquitetura e Organização de Computadores
# Atividade: Avaliação 02 – Programação em Linguagem de Montagem
# Programa 02
# Grupo: - Ringo Starr
#        - John Lennon
```

6. O arquivo ASM e o relatório em formato PDF deverão ser postados no ambiente Material Didático, em um único arquivo compactado em formato ZIP.

7. O prazo para postagem do relatório e dos códigos fonte é 08h00 do dia **16/05/17**.

8. Não serão aceitos trabalhos entregues em atraso.

9. O professor poderá solicitar a qualquer momento que **qualquer aluno** do grupo faça uma **demonstração explicativa da execução do código no MARS**.

10. **A implementação deverá apresentar resultados corretos para qualquer conjunto de dados.** Uma solução que **não execute corretamente** terá, automaticamente, um **desconto de 50% na nota**, sendo que o professor também avaliará a correção de segmentos específicos do código (controle de execução, acesso a memória,...).

11. Se forem identificados trabalhos com grau de similaridade que caracterize cópia (autorizada ou não) ou adaptação, a nota dos grupos será a nota de um trabalho dividida pelo número de grupos que entregou esses trabalhos similares.

EXERCÍCIOS DE AQUECIMENTO

Os exercícios propostos abaixo deve ser feitos para fins de aquecimento (não valem nota). Tente(m) fazê-los antes de iniciar a implementação do programa desta avaliação. Isso os ajudará a consolidar os fundamentos da programação de procedimentos na linguagem de montagem do MIPS antes da implementação de um procedimento mais completo.

Exercício 01 – Chamada de procedimento
Implemente um programa que:

a) Leia dois números (X e Y) usando a instrução `syscall`;
b) Chame um procedimento chamado SOMA, passando X e Y nos registradores de argumento;
c) Imprima o valor retornado pelo procedimento, o qual deve fazer a soma de X e Y.

O código do procedimento deve ser escrito no início do segmento de código (.text), antes da função principal (main). No entanto, no MARS, o procedimento deve ser precedido por um desvio incondicional para a função principal. Exemplo:

```
      .text
      j main
soma :  ...
      jr $ra

main : ....
      jal soma
```

OBS: Este exercício aborda o uso das instruções `jal` e `jr`.

Exercício 02 – Uso de pilha
Implemente um programa que:

a) Inicialize os registradores `$s0`, `$s1` e `$s2` com os respectivos valores 7, 8 e 9;
b) Estenda o procedimento anterior (Exercício 01) de tal forma a:
   a. Salvar `$s0`, `$s1` e `$s2` na pilha;
   b. Executar as seguintes operações:

```
$s0 <- $a0
$s1 <- $a1
$s2 <- $a2
```

   c. Recupere os valores salvos na pilha e retorne com a soma.

c) Imprima o valor retornado pelo procedimento

Exercício 03 – Vetor
Implemente um programa que:

a) Solicite a entrada de um vetor com oito (8) elementos;
b) Chame um procedimento fornecendo o endereço base e o tamanho do vetor;
c) Imprima o valor retornado pelo procedimento que deve conter o número de elementos nulos (zero) no vetor. Exemplo:

```
Vetor_A: .word 9 0 7 0 3 0 4 1
```

d) Para o vetor apresentado o resultado do programa deverá ser 3, ou seja, o vetor A possui 3 elementos nulos (iguais a zero).

ESPECIFICAÇÃO DO PROBLEMA

Enunciado:
Utilizando a linguagem de montagem do MIPS, implemente um programa sobre dados geográficos de países que realize a classificação dos mesmos sob diferentes indicadores (População, PIB per capita e IDH) utilizando o algoritmo de ordenação *bubblesort* implementado na forma de um procedimento.

Requisitos:

1. **Declaração dos vetores:** Na seção de declaração de variáveis (.data), devem ser declarados vetores para armazenar os nomes e os indicadores geográficos de até 06 países, incluindo:

    a. Vetor de inteiros com Código DDI dos países (identificador inteiro do País)
    b. Vetor de inteiros com a População dos países expressa em Milhões de Habitantes
    c. Vetor de inteiros com o IDH dos países multiplicado por 1000
    d. Vetor de inteiros com o PIB per capita dos países

2. **Entrada do tamanho dos vetores:** A função principal (MAIN) deve solicitar o número de países aceitando no mínimo 2 e no máximo 6 países. Para leitura, deve ser apresentada uma mensagem solicitando a entrada desse valor, indicando o seu limite máximo.

3. **Verificação do tamanho do vetores:** A função principal (MAIN) deve solicitar a entrada do número de elementos até que ele seja **maior que 1** e **menor ou igual a 6**. Ou seja, deve implementar um mecanismo de filtragem que não aceite entrada diferente da especificada. No caso de entrada inválida, a função principal deve imprimir uma mensagem de advertência antes de solicitar novamente a entrada.

4. **Entrada dos elementos dos vetores:** Para leitura, a função principal (MAIN) deve solicitar ao usuário a entrada de cada elemento do vetor, um a um, com mensagens do tipo:

```
DDI[0] =
DDI[1] =
...

POP[0] =
POP[1] =
...

PIB[0] =
PIB[1] =
...

IDH[0] =
IDH[1] =
...
```

Exemplo: 03 países

País	DDI	População (Milhões)	PIB per capita (US$)	IDH (x1000)
Brasil	55	191	11.067	744
França	33	65	45.383	884
China	86	1.338	12.893	687

Obs: Dados obtidos na Embratel e na Wikipedia

5. **Solicitação de indicador:** A função principal (MAIN) deve solicitar ao usuário o indicador a ser usado para classificar os países com mensagem do tipo:

```
Selecione o indicador a ser utilizado para a classificação dos países:
- 1. População
- 2. PIB per capita
- 3. IDH

Opção:
```

6. **Interface da função principal com o procedimento:** A função principal (MAIN) deve copiar o número de países (tamanho do vetor) para o registrador $a0 e o endereço base do vetor correspondente ao indicador selecionado para o registrador $a1 e o vetor endereço base do vetor DDI para o registrador $a2 antes de chamar o procedimento BUBBLESORT.

7. **Funcionalidade do procedimento:** O procedimento BUBBLESORT deve realizar a ordenação do vetor do indicador selecionado, com base no conteúdo dos seus elementos, e também do vetor DDI (identificado do país), de acordo com a ordem estabelecida para o primeiro vetor.

8. **Uso de registradores e da pilha pelo procedimento:** Caso seja necessária a utilização de algum registrador de uso geral (ex. para implementar laços de repetição), o procedimento deve, **OBRIGATORIAMENTE**, utilizar apenas registradores salvos $s, preservando-os na pilha no começo do procedimento e restaurando-os ao seu final.

9. **Retorno do procedimento:** Como o procedimento recebe os endereços base do vetor, não há necessidade de retornar nenhum valor pelos registradores **$v**. A função principal terá acesso aos vetores ordenados pelos registrador **$a1** e **$a2**.

10. **Impressão do vetor ordenado:** Após o retorno do procedimento, a função principal MAIN deve imprimir os vetores ordenados no terminal.

```
Classificação dos países para o critério selecionado: População
1o lugar - DDI = 86, População = 1338
2o lugar - DDI = 55, População = 191
3o lugar - DDI = 33, População = 65
```

Nota: Alternativamente, ao invés do DDI, pode ser impresso o nome do país (não obrigatório)

11. **Estilo de codificação:** O código deve ser escrito respeitando o estilo de programação ASM, usando tabulação para organizar o código em colunas (rótulos, mnemônicos, operandos e comentários).

12. **Comentários:** Procure comentar ao máximo o seu código. Isto é um hábito da programação *assembly*.

13. **Análise das instruções utilizadas**: No seu relatório, **apresente uma análise** indicando a contagem de instruções executadas para cada de classe utilizando a ferramenta "Instruction Statistics" do MARS (conecte a ferramenta ao MIPS antes de executar a simulação), conforme o exemplo anterior.

NOTA:

Para ter certeza do funcionamento correto do procedimento em qualquer circunstância, durante o seu teste, é recomendável que a função principal inicie todos os registradores `$s` utilizados pelo procedimento com um valor diferente de 0, como, por exemplo:

```
addi $s0, $zero, 0xFFFFFFF0
addi $s1, $zero, 0xFFFFFFF1
addi $s2, $zero, 0xFFFFFFF2
...
```

Após a execução do procedimento, se ao retornar à função principal, os valores originais desses registradores tiverem sido restaurados, então o tratamento da pilha estará correto.