UFPR – Bacharelado em Ciência da Computação
CI064 – Software Básico — 2019-1

Primeira Prova Semestral

1. Esta questão tem três itens. [5 pontos]

(a) Desemhe um diagrama que indique claramente as dependências de dados no código;

(b) indique claramente quaisquer modificações no código para garantir que ele seja executado corretamente num processador que não possua circuitos de adiantamento (*forwarding*);

(c) desenhe um diagrama simplificado do processador e indique quais os circuitos de adiantamento são necessários especificamente para este trecho de código.

```
    li   r8, 0
    la   r9, reduc
    la   r2, (vetor + 99*4)
L:  lw   r6, 0(r2)
    add  r8, r8, r6
    addi r2, r2, -4
    bne  r2, zero, L
    sw   r8, 0(r9)
```

Considere que o texto abaixo é o conteúdo do arquivo `R3`.

Now is the winter of our discontent
Made glorious summer by this sun of York;
And all the clouds that lour'd upon our house
In the deep bosom of the ocean buried.
Now are our brows bound with victorious wreaths;
Our bruised arms hung up for monuments;
Our stern alarums changed to merry meetings,
Our dreadful marches to delightful measures.
Grim-visaged war hath smooth'd his wrinkled front;
And now, instead of mounting barded steeds
To fright the souls of fearful adversaries,
He capers nimbly in a lady's chamber
To the lascivious pleasing of a lute.

2. Esta questão tem cinco itens. Suas respostas devem ser as saídas dos programas indicados em cada item. [10 pontos]

(a) `grep -i our R3`

(b) `head -5 R3 | cut -c 1-3`

(c) `grep Now R3 | tr a-z A-Z`

(d) `tail -3 R3 | sort`

(e) `grep 'ful ' R3 | cut -d' ' -f 1,3,5`

3. Esta questão tem dois itens.

(a) Re-escreva o código fonte do laço `for`, para obter um tempo de execução menor. Justifique sua(s) escolha(s). [3 pontos]

(b) Traduza seu código C otimizado para *assembly* do MIPS e o otimize para execução no processador segmentado de 5 estágios. Seu código *assembly* deve empregar as convenções de programação do MIPS. O processador possui adiantamento, e necessita de *branch* e *load delay slots*. Multiplicações demoram o mesmo que adições. [7 pontos]

Para facilitar a correção indique os registradores como `ri`, `ra`, `rb`, etc.

```
#define N 1024
int A[N], B[N], C[N];
...
s = 0;
for (i=0; i < N; i=i+1) {
   s = s + C[i] + (A[i] * B[i]);
}
...
```

Segunda Prova Semestral

4. Esta questão tem três itens e todos referem-se à ligação com bibliotecas compartilhadas estáticas.

(a) Mostre, e explique, a sequência de instruções de *assembly* necessária para a invocação de uma função exportada pela biblioteca – ignore os argumentos e o valor de retorno; [5 pontos]

(b) considere a variável exportada `bibVar` e suponha que a tabela de saltos e de variáveis contém o endereço interno à biblioteca daquela variável. Mostre, e explique, a sequência de instruções de *assembly* necessárias para uma referência de leitura em `bibVar`; [5 pontos]

(c) o segmento de código de uma biblioteca é compartilhado entre todos os processos; o que ocorre com o compartilhamento do segmento de dados? [5 pontos]

5. Numa tabela de símbolos com *hashing*, o valor da função de *hash* do símbolo (H(`str`)) é armazenado junto com o (apontador para a *string* do) símbolo. Explique a razão para isso. [5 pontos]

6. Explique como o ambiente de execução (variáveis de ambiente da *shell*) são transferidos para a função `main()`. [5 pontos]

Exame Final

7. Esta questão tem três itens.

(a) Desemhe um diagrama que indique claramente as dependências de dados no código – copie o trecho para a folha de respostas e deixe uma ou duas linhas entre cada instrução;[10 pontos]

(b) indique claramente quaisquer modificações no código para garantir que ele seja executado corretamente num processador que não possua circuitos de adiantamento (*forwarding*) externos ao bloco de registradores; [10 pontos]

(c) desenhe um diagrama simplificado porém claro e limpo do processador e indique quais os circuitos de adiantamento que são necessários especificamente para este trecho de código. [10 pontos]

```
    lui  r2, %hi(vetLim)
    ori  r2, r2, %lo(vetLim)
    add  r8, zero, zero
L:  lw   r6, 0(r2)
    add  r8, r8, r6
    addi r2, r2, -4
    bne  r2, zero, L
    lui  r9, %hi(reduc)
    ori  r9, r9, %lo(reduc)
    sw   r8, 0(r9)
```

8. Re-escreva o código fonte do laço **`for`**, para obter um tempo de execução menor. Traduza seu código otimizado para *assembly* e o otimize para execução no processador segmentado de 5 estágios. Seu código *assembly* deve empregar as convenções de programação do MIPS.[40 pontos]

O resultado da multiplicação só pode ser usado por **`mfhi,mflo`** após um ciclo de espera. O processador possui adiantamento, e necessita de *branch* e *load delay slots*.

Para facilitar a correção indique os registradores como `ra`, `rb`, etc.
Escreva seu programa nas páginas do meio da folha.

```
#define N 1024
int A[N], B[N], C[N];
...
s = 0;
for (i=0; i < N; i = i+1) {
   C[i] = (A[i] * B[i]);
}
...
```

9. Esta questão seis itens. Suas respostas devem ser concisas e precisas.

(a) explique, na programação em *assembly*, as diferenças entre *pseudoinstruções* e *diretivas*; [5 pontos]

(b) enuncie o conceito de *abstração* e dê três exemplos de abstrações providas pelo sistema operacional Unix e seus derivados; [5 pontos]

(c) descreva a operação de um *pipeline* no Unix e justifique sua utilidade; [5 pontos]

(d) quais as diferenças entre o registrador STATUS do MIPS e o registrador de status da UART? Para que servem *dois* registradores de status? [5 pontos]

(e) mostre como as interrupções no processador MIPS são mascaradas com os registradores STATUS e CAUSE; [5 pontos]

(f) enuncie a função do programa `cpp`. [5 pontos]