COE 308 – Fall 2011
Computer Architecture

Programming 2: Floating-Point Computation and Counting Instructions

Problem 1: Exponential Function

The following series can be used to approximate e^x

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \cdots$$

Write a MIPS assembly language function to compute the value of e^x and the estimated error. Ask the user to enter a double-precision floating-point number x and the number n of terms to be used in the above series. Display the estimated value of e^x and the estimate error. The estimated error (in absolute value) using the first n terms is: $| x^n / n! |$. All computation should be in double-precision.

As sample run is shown below:

```
Enter x: 1.2
Enter number of terms: 10
Estimated value of e^x = . . .
Estimated error = . . .
```

Problem 2: Program Analysis and Counting Instructions

Analyze the MIPS code of Problem 1, to have a better understanding of instruction frequencies. You will count the dynamic number of instructions that are executed at runtime to compute the estimated value of e^x. You need a total of four counters to count instructions for the following classes of instructions:

- Class 1 is for ALU instructions.
- Class 2 is for Floating-point instructions.
- Class 3 is for load and store instructions.
- Class 4 is for branch and jump instructions.

You will augment the code of the exponential procedure with additional instructions to count the original number of instructions. At the beginning of the procedure, initialize all counters to zeros. For each group of instructions, insert additional instructions to increment the counters. If an instruction is executed 100 times, it will be counted as 100. For pseudo-instructions, count the equivalent real instructions. Make sure that your additional code does not interfere with the original program code. Count only the original instructions, not the new ones that you have added.

At the end, display the statistics for the exponential procedure. A sample run is show below:

```
Enter x: 1.2
Enter number of terms: 10
Estimated value of e^x = . . .
Estimated error = . . .
```

```
Statistics for the Exponential Procedure:
Total               instructions = ???
ALU                 instructions = ??, Percentage = ?%
Floating Point      instructions = ??, Percentage = ?%
Load & Store        instructions = ??, Percentage = ?%
Branch & Jump       instructions = ??, Percentage = ?%
```

Problem 3: Matrix Traversal

Write a MIPS assembly language program to do square matrix computation. Ask the user to enter the size *N* of a square matrix and to enter the matrix elements. The square matrix size should be restricted to 10 rows by 10 columns. The matrix elements should be single-precision floating-point numbers. Compute the sum of the elements along each row, the sum of the elements along each column, and the sum of the elements along the main diagonal. For a 5×5 square matrix, this should produce 5 row sums, 5 columns sums, and one diagonal sum.

A sample run is show below for a 2 by 2 matrix:

```
Enter Matrix Size N (range 2 to 10): 2
Enter Element [0,0]: -0.5
Enter Element [0,1]: 2.3
Enter Element [1,0]: 1.0
Enter Element [1,1]: 3.7

Sum of values along row 0 = 1.8
Sum of values along row 1 = 4.7
Sum of values along col 0 = 0.5
Sum of values along col 1 = 6.0
Sum of values along main diagonal = 3.2
```

Submission Guidelines:

All submissions will be done through WebCT.

Submit the source code of all programs. Make sure that all programs are well documented.

Grading Policy:

The grade will be divided according to the following components:

- Correctness of code: program works properly and produces correct results
- Design and Coding: program is well designed and divided into procedures
- Documentation of code: program is well documented