CSCI 2500 — Computer Organization
Homework 3 (document version 1.2)
MIPS Instruction Set and Matrix Multiplication in MIPS

Overview

- This homework is due by 11:59:59 PM on Friday, October 18, 2019.
- This homework is to be completed **individually**. Do not share your code with anyone else.
- You **must** use MIPS for the programming portion of homework assignment, and your code **must** successfully execute on Submitty to obtain full credit.
- Please put all of your MIPS code into the single `hw3.s` source file.

Homework Specifications

For this individual homework assignment, you will work on a mixture of textbook problems and MIPS code.

First, start with some "warm-up" exercises, which you will **not** submit as part of this assignment. In other words, do these "warm-up" exercises as practice and to prepare to work on the actual problems you will submit for credit.

Warm-Up Exercises (for Practice)

1. **Textbook Problem 2.7**
2. **Textbook Problem 2.15**
3. **Textbook Problem 2.24**

Homework Problems (to Submit for Credit)

Use whatever software you like to write your answers to the textbook problems below. You must produce a PDF to submit for this assignment. Please name your PDF `hw3.pdf`. These will be manually graded by our TAs.

1. **Textbook Problem 2.8**
2. **Textbook Problem 2.16**
3. **Textbook Problem 2.18 (all sub-parts)**
4. **Textbook Problem 2.25 (all sub-parts)**

Coding Problem (to Submit for Credit)

Implement matrix multiplication, this time using MIPS. More specifically, you will read in two matrices from the user and multiply them together. As with Homework 1, if you need a refresher in how matrix multiplication works, look in a math textbook or check out Wikipedia!

The first matrix is an $n \times k$ matrix, while the second matrix is a $k \times m$ matrix. Therefore, the result will be an $n \times m$ matrix. Use the `read_int` system call (`syscall`) to read in n, k, and m, as well as each unsigned integer matrix value.

One approach you could take is to store these important values in your `.data` section as follows (with sample hard-coded values shown):

```
        .data
n:      .word 4
k:      .word 3
m:      .word 4
```

Once you have your matrix sizes defined, dynamically allocate memory to store the actual matrices. This would be equivalent to calling `malloc()` or `calloc()` in C to allocate memory on the heap. And remember that each integer is one word (or four bytes) in size.

Example Program Execution

On the next page is an example MIPS program execution that you can use to better understand how your program should work, how you can test your code, and what output formatting to use for Submitty. Also use test cases from Homework 1 to test your MIPS code.

Note that you must input each value on a separate line in MIPS. And you can assume that the input given to your program is valid.

When displaying a matrix, each line must start with '`[`' and end with '`]`' (as with Homework 1), but in this assignment, left justify the columns by using TAB ('`\t`') characters as follows:

```
[12\t34\t5567\t]
[8\t9\t123\t]
[45\t67\t8\t]
[9\t10\t11\t]
```

This will display this 4×3 matrix as follows:

```
[12    34     5567   ]
[8     9      123    ]
[45    67     8      ]
[9     10     11     ]
```

```
(spim) load "hw3.s"
(spim) run
Please enter values for n, k, and m:
4
3
4
Please enter values for the first matrix (4x3):
10
20
30
40
50
60
70
80
90
100
110
120
Please enter values for the second matrix (3x4):
0
10
0
20
30
0
40
0
0
50
0
60

[10    20     30     ]
[40    50     60     ]
[70    80     90     ]
[100   110    120    ]
multiplied by
[0     10     0      20     ]
[30    0      40     0      ]
[0     50     0      60     ]
equals
[600   1600   800    2000   ]
[1500  3400   2000   4400   ]
[2400  5200   3200   6800   ]
[3300  7000   4400   9200   ]
(spim)
```

Error Checking

Given the complexity of this assignment, you can assume that all input values are valid unsigned integers. You can also assume that the correct number of values is given for each matrix. In other words, you do not need to validate the user input.

Submission Instructions

For this assignment, you will submit both your code and your PDF (i.e., `hw3.pdf`) with your answers to the textbook problems to submit.

Before you submit your code, be sure that you have clearly commented your code (this should not be an after-thought). Further, your code should have a clear and logical organization. Use registers appropriately, and create reusable procedures (just be sure to manage the stack properly).

To submit your assignment (and also perform final testing of your code), please use Submitty.

Please put all of your MIPS code into the single `hw3.s` source file.

Note that the test cases for this assignment will be available on Submitty a few days before the due date and will include hidden test cases.

Also as a reminder, your code **must** successfully execute on Submitty to obtain credit for this assignment.