CMSC 216 Quiz 5 Worksheet

The next quiz for the course will be on Mon, Mar 30. The following list provides additional information about the quiz:

- Do not post any solutions to this worksheet in Piazza. That represents an academic integrity violation.
- The quiz will be a written quiz (no computer).
- The quiz will be in lab session.
- Closed book, closed notes quiz.
- Answers must be neat and legible.
- Quiz instructions can be found at http://www.cs.umd.edu/~nelson/classes/utilities/examRules.html
- Make sure you know your section number and your TA's name.

The following exercises cover the material to be included in this quiz. Solutions to these exercises will not be provided, but you are welcome to discuss your solutions with the TA or instructor during office hours. We will include the following cheat sheet in the quiz so you don't have to memorize some Assembly constructs.

Assembly Cheat Sheet

Registers → %eax, %ecx, %edx, %ebx, %esi, %edi
Assembler Directives → .align, .long, halt, .pos
Data movement → irmovl, rrmovl, rmmovl, mrmovl
Integer instructions→ addl, subl, multl, divl, modl
Branch instructions → jmp, jle, jl, je, jne, jge, jg
Reading/Writing instructions → rdch, rdint, wrch, wrint
Ascii code for newline character → 0x0a
Ascii code for space → 0x20

End of Assembly Cheat Sheet

Solutions to the exercises below will not be provided, but you are welcome to discuss your solutions with the TA or instructor during office hours.

Exercises

1. What is the difference between Big Endian and Little Endian? Suppose we have the value 0x02143657. How would the value be represented using Big/Little Endian?

2. What is the purpose of the .align directive?

3. What is the purpose of the .pos directive?

4. Is it possible to have an Assembly (Y86) program with an array of 5000 elements? Briefly explain.

5. What is the difference between **irmovl MyData, %eax** and **mrmovl MyData, %eax**, assuming MyData is a label?

6. Write Assembly code that will define a global variable named x that has some initial value (e.g., 100). The program will read an integer and will print the result of dividing that value by x.

7. Write an Assembly program that reads an uppercase letter and prints the corresponding lowercase. To compute the lowercase just add 32 to the ascii value of the uppercase. Make sure a \n is printed after the lowercase letter.

8. Write Assembly code that will read two integer values and print 0 if the first value is divisible by the second, and any other number otherwise.