CSCI 3320
Fall 2016
Exam 2

Last Name: _____

First Name: _____

Write all answers on the answer sheets provided. You *may* use the following resources during the exam: *x86 Assembly Quick Reference*, `iomacros.asm` source code, calculator.

1. (2 pts) For each `MOV` command below state whether the command is valid or not. If valid, explain (including size of operands) what the `MOV` accomplishes. If not valid, explain why not. Assume all labels have been declared.

   (a) `mov ax,[a]`

   (b) `mov dword [x], dword [y]`

   (c) `mov rbx,george`

   (d) `mov [rax+12],ecx`

   (e) `mov [x],12`

2. Write the x86_64 assembly language statements that correspond to each of the following high-level language statements. The variables `i`, `j`, `k`, and `n` are all 32-bit signed integers.

   (a) (4 pts) `i= j-3*k;`

   (b) (8 pts) `for (i=n-1; i>=0; i--) System.out.println(i);`

3. (12 pts) Suppose that `a` is an array of `n` 64-bit signed integer values. Write a section of x86 assembly code that will calculate the sum of all of the values. Put the resulting sum in RAX. You may assume that all values are positive and that `n` is a 64-bit signed integer.

4. Suppose you have an array of structured data with each element containing the following information about a person: id number, age, and length (all 32-bit integers) and name (a null-terminated buffer 20 characters). Suppose the array, marked with the label `people`, has a capacity large enough to store information for 1000 people. Further suppose that the label `n` refers to a 32-bit integer that stores the number of elements currently in use in the array.

   (a) (4 pts) Write the code you would use to declare the `people` array in assembly.

   (b) (16 pts) Suppose the array has been created and that the label `n` has been apporpriately initialized. Write code to set the length value for each entry to be the number of characters stored for the name. The length of each name can be determined by looking at each character in the name until the zero byte is encountered.

   (c) (6 pts) Write assembly code that will swap the contents of the first and last elements of the array.

(d) (10 pts) Write assembly code that will reserve and initialize an array (called `tag`) of pointers/references to each array element. The addresses in the `tag` array should point to the individual records in `people`.