CSCI 2330 – GDB & (More) Assembly Exercises

1. What GDB command (just one) should you use for each of the following situations when debugging an assembly program (without the source code)?

(a) You are paused on **callq foo**, and you want to execute the entire function and then pause after returning.

(b) You are paused on **callq foo**, and you want to step into the function and then pause execution again.

(c) You accidentally stepped into a call to **malloc** and want to return to the calling function (i.e., back into your own code).

(d) You want to know what calling **foo(20)** would return (but the program isn't about to make that call).

(e) You are at a breakpoint within a loop and want to run the next loop iteration (you can assume there is only the one breakpoint set).

2. Write a single GDB "**x**" command ("examine memory") to do each of the following (you must use the **x** command, not **print**):

(a) Print a 4-byte int stored in memory at address **%rax**, in decimal.

(b) Print an 8-byte int stored in memory at address **%rax**, in hex.

(c) Print a string stored in memory at address **%rax**.

(d) Print a string stored in memory at address 0x123456.

(e) Print an array of 5 chars starting at address **%rax**, showing their decimal values.

(f) Print an array of 5 chars starting at address **%rax**, showing their textual values.

(g) Print an array of 5 pointers starting at address **%rax**.

3. Rewrite the x86-64 instructions below as a C function, which you can assume is non-void and takes one argument. Remember to specify the appropriate types of the argument and return value. You can use any local variables you wish. **Do not use any goto statements in your function.**

```
compute:
     movq      $0, %rax
     movq      $1, %rbx
     jmp       .L2
.L1: addq      $1, %rax
     salq      $1, %rbx        # left shift
.L2: cmpq      %rdi, %rbx
     jl        .L1             # jump if less
     ret
```