**CSC 406: Systems I**

**Homework 3**

**Due by 11:59 pm on Monday, October 24**

Please write precise and concise answers.

**Reading**

Read sections 3.1-3.6 and and 3.7.1-3.7.4 of BO\* as well as week 5 lecture notes.  
  
BO\* = Computer Systems by Bryant and O'Hallaron  
  
Make sure that you work out every practice problem in BO\*!  Peek at the solutions (presented at the end of the chapter) if you have to.

**Assignment**

Unzip file hw3.zip

$ tar -xvf /home/sparsa2/public/hw3.tar

and move to newly created directory hw3:

$ cd hw3

This directory contains files hw3.c and Makefile. Open file hw3.c for editing and do the homework:

$ emacs hw3.c

When done, submit the homework as follows:

$ make submit

**1**.    Compile the file sum.c, which is in the hw3 folder, using the following command.   
  
gcc -Og -c sum.c   
  
Next, dis-assemble the object file created using objdump:   
  
objdump -d sum.o   
  
Study the generated assembly and find out what registers are used to store variables i and s of the code in sum.c. Explain your reasoning. Write your answer as a comment in hw3.c.   
  
**2.**    For a function with prototype

long decode2(long x, long y, long z);

gcc generates the following assembly code:

decode2:  
        movq    %rdi, %rax  
        subq    %rdx, %rax  
        movq    %rax, %rdx  
        imulq   %rax, %rdi  
        salq    $63, %rdx  
        sarq    $63, %rdx  
        xorq    %rdx, %rdi  
        leaq    (%rdi,%rsi), %rax  
        ret

Parameters x, y, and z are passed in registers %rdi, %rsi, and %rdx. The code stores the return value in register %rax.  
  
Reverse-engineer decode2. (In other words, write C code for decode2 that will have an effect equivalent to the assembly code shown.)  
  
Note: read Section 3.5 first and write your solution in hw3.c. To check your solution, compile hw3.c to assembly code using

$ gcc -Og -S hw3.c

and then view the file hw3.s:

$ cat hw3.s

**3**.    Problem 3.60 in BO but using this assembly code instead:

        movl    %esi, %ecx  
        movl    $1, %edx  
        movl    $0, %eax  
.L3:  
        testq   %rdx, %rdx  
        je      .L5  
        movq    %rdx, %r8  
        andq    %rdi, %r8  
        orq     %r8, %rax  
        salq    %cl, %rdx  
        jmp     .L3  
.L5:  
        ret

Your solution should be an implementation of C function loop(). You can check your solution as you did for problem 1. Note that there is a mistake in the book in line 1 of page 313. The function signature should be long loop(long x, int n).