Programming Assignment # 2

Code Bumming!

Session 1

1. Objectives

Write assembly code of some C functions. The resulting code must be efficient and made with as few instructions as possible. Your code must be compiled successfully to an object file (`*.o`) and must work correctly when they are called from C source files.

2. Details

A. You are given a set of skeleton assembly files and the C files that call the assembly procedures. Each directory is for each problem. Those files can be compiled into executable files by invoking '`make`' command in each directory.

B. These files are for IA32 Linux systems. If you have installed a x86-64 Linux distribution, install `gcc-multilib` support to your system. It will enable your system to compile IA32 assembly codes.

C. You are supposed to complete the skeleton assembly code so that they work correctly as written in the problem description. Code with fewer instructions will earn better score.

C. Any reverse engineering (from C code) is strictly prohibited. Think and write only in assembly.

D. The skeleton assembly files can be downloaded at
http://csl.skku.edu/uploads/CSE2003S12/skeleton.tgz

3. Problems

A. `int factorial(int n)`

This function gets an integer, `n`, as its parameter and returns `n!`. If you initiate '`make`' program in the directory, you will get the caller program named as '`factorial`'. The caller program calls this function with an argument that is obtained from the command line argument. For example, the command line input '`factorial 443`' will obtain and print out the factorial of 443 by calling the function with an argument 443. The input is assumed to be always a positive integer.

B. `int reverse(int n)`

This function reverses the integer parameter and returns the reversed number. For example, if n is 123 then 321 will be returned. If you initiate '`make`' program in the directory, you will get the caller program named as '`reverse`'. The caller program calls this function with an argument that is obtained from the command line argument. For example, the command line input '`reverse 321`' will obtain

and print out the reverse of 321, which is 123, by calling the function with an argument 321. The input is assumed to be always a positive integer.

C. `int is_armstrong(int n)`

This function checks whether a number is armstrong or not. It returns 1 when n is Armstrong, otherwise 0. A number is armstrong if the sum of cubes of individual digits of a number is equal to the number itself. For example 371 is an armstrong number as $3^3 + 7^3 + 1^3 = 371$. Some other armstrong numbers are: 0, 1, 153, 370, 407. If you initiate '`make`' program in the directory, you will get the caller program named as '`armstrong`'. The caller program calls this function with an argument that is obtained from the command line argument. For example, the command line input '`armstrong 371`' will obtain the result by calling the function '`is_armstrong`' with 371 as its argument and print out a message '`371 is an armstrong number`'. The input is assumed to be always a positive integer.

4. Logistics

A. The completed source files should be tar-and-gzipped into a file. The name of the zipped file should be "studentid.tgz" (e.g. 2011310123.tgz)

C. Prepare a separate document in PDF format, which explains the design and implementation of your code. The document file name should be "studentid.pdf" (e.g. 2011310123.pdf)

D. Send a mail to [*homework.skku@gmail.com*] with attaching the two files, a zipped source code file and documentation. The subject of the mail should be *[PA#2] studentID*.

E. The submission status will be posted on the course home page at http://csl.skku.edu/CSE2003S12

F. Only the assignments submitted before the deadline will receive the full credit. 25% of the credit will be deducted for every single day delay.