**CSCI 232 Operating Systems**

**Homework 2 – System-call level file I/O**

This assignment will take you further into manipulating files with system calls. You will be dealing with PNG files, which are binary. You will need to read about the png format, which is one of the simplest image formats:

<https://en.wikipedia.org/wiki/Portable_Network_Graphics>

The PNG format consists of a series of “chunks”, and each chunk consists of a size, a type, its ***data***, and a CRC code. The ***image data*** itself is stored in one or more chunks of the type “IDAT”. A chunk of type “IEND” follows the last IDAT chunk.

In this assignment, you must take a “corrupted” PNG image and restore it. You will be provided with 2 corrupted files for your testing. The images are corrupted in a uniform way:

* Every byte of ***image data*** has been XORed with the integer 232.

Your program has the following requirements:

* Your C program must take one command-line argument: the png image filename.
* Your program should operate on the input file itself, not use a temporary file.
* You must use the file I/O system calls (open, close, read, write, lseek, etc.), not the C library functions, but you may use the C string functions and printf.
* You should do a medium amount of error handling: check for the correct number of arguments, file permissions, and badly formatted files (i.e., you don't find chunks of the right type).

Other remarks:

* As always, this is an individual assignment. If you get help from a web resource or a person, please give credit in a code comment.
* The solution to this problem is not long or complicated code, but to solve it requires some interesting handling of the mixture of binary and textual data in the png format.
* Recall the function htonl that I discussed in the lab, which is needed regardless of which way your computer stores the bytes. See ***endianness***.
* There are two useful programs that can help you examine the contents of binary files: od and hexdump. You can use them to look at the png files and see the chunk structure.