

## Reading and Using Binary Data in C

*Phys494, Fall 2007 Lab Notes, Instructor: Alpar Sevgen, Assistant: İsmail Ari*

There are three files we use in this process.

### **readingSequentially.c**

This is an example program that shows how to read numbers sequentially from a file, apply simple operations (multiplying all numbers in the file by 2) and write the outputs to another file again sequentially.

There's also a function in the code named as "createTestFile" which you can use to test the code with initial data involving numbers from 1 to the given number.

The objective of this code is to teach you how to read and write binary data using `c`, and familiarize you to use `FILE` struct, `fread` and `fwrite` functions.

### **readingInBlocks.c**

This is a modification of the previous example. In this version, blocks are introduced. The file is not read element by element, instead, it is read in blocks. For example, the first 10 elements are read and processed initially. Then the next 10 elements are processed, and it continues till the end in this manner.

The objective of this function is to consider efficiency in your code. There are two approaches when we need to read files. First, we can read the entire file into an array but this way requires a very large memory to handle all of the data. In real life, there are very large files like video files. On the other hand, we can read the elements one by one, but it will be very time consuming because file operations will dominate the run time. As seen, these ways are not so efficient. Combining these two approaches will be a better approach. For example, the file is 1 GB and we read it in 1MB blocks. By this way, we take a block of the file to the memory and do the calculations faster.

### **plotData.m**

This function plots the input and output data with the given number of elements. It is written in Matlab.

The objective here is to consider another efficiency issue. The efficiency we considered in the previous example was the run-time efficiency. Here, we consider the development efficiency, i.e. our efficiency in writing codes. We mostly write the code in Matlab to develop algorithms faster, but in some parts of the code, we need faster implementations. We did them in `c` as in the previous examples. Now, we use Matlab to plot the results of the `c` code executions.

*Note: For a generic implementation of block usage, please read "Reading and Processing Binary Data in Blocks" part in the course webpage.*