CS220 - Computer System II
Assignment 7 (100 points)

Due: 10/23/2016, 11:59pm

1 Instructions

- Answer the questions individually. Group effort is not allowed.
- Reading: `http://c-faq.com`, K&R C, chapter 1-6.

2 Questions

1. Programs operate on files with specific format. For example, Linux executes elf binaries. Microsoft Windows executes PE binaries. Elf is a file format that tells the loader where and what to find inside the file. Similarly, a Word processors can open .doc files, Zip applications can open .zip, .gz, etc. files, and so on. In order to aid in detecting if a file is of a particular type, a signature is embedded within the file to indicate the type of a file. Often, the signature is the first few bytes within a file. This will enable an application to quickly identify if it can operate on the file. In this problem, you will implement a program `file_recognizer` that accepts a file as command line input, and prints the format of the file and the program that can open it. NOTE: Although file extensions can sometimes tell what format a file is, it is an unreliable indicator. The file signature is the true indicator of a file type. **(100 points)**

Table 1: Signature to type mapping. This is only a sample for the assignment. For curious minds, more can be found here: (https://en.wikipedia.org/wiki/List_of_file_signatures)

(a)

Signature (first few bytes in Hex)	**File type**	**Expected output**
47 49 46 38 39 61 or 47 49 46 38 37 61	GIF Image	This is a GIF file.
7f 45 4C 46	ELF file	This is an ELF file.
25 50 44 46	PDF file	This is a PDF file.
50 4B	ZIP file	This is a ZIP file.
CA FE BA BE	Java class file	This is a Java class file.
89 50 4E 47 0D 0A 1A 0A	PNG file	This is a PNG file.
Anything else	Unknown file	File type unknown.

(b) All the above signatures will start from the beginning of the file.

2. You will implement file_recognizer.c and file_recognizer.h (optional) that contains all your implementation. You will write a Makefile that generates file_recognizer executable. Your program must accept exactly 1 command line argument (i.e., file name) other than the program name. If there is more than 1 argument or if the input file can not be opened/read from, you are to print ”Invalid input.” and exit.

3 Useful resources

1. Examine fopen() and fscanf() functions to open and read from files.
2. Run the find command (e.g., `find / -name "*.png"`) on remote to find files of specific types to test.

4 Testing your code

In order to test your code, we will run the make command to generate the program. We will then call the program in a loop on a set of known file types, and in each case, we will check if the output is as exepected in Table 1. We will also provide invalid number of arguments and nonexistent files as arguments. NOTE: The testing process may be automated. So, it is your responsibility to ensure that the files are named correctly. You are advised to test the programs before submitting. Each valid input file is guaranteed to be at least 10 bytes in size.

5 Submitting your code

You will create a folder assn7 with file_recognizer.c, file_recognizer.h (optional) and Makefile. Create assn7.tar.gz that contains assn7 and submit to blackboard.