Gebze Technical University
Department Of Computer Engineering
CSE 312 / CSE 504

Operating Systems

Homework #03
Due Date: Jun 15th 2018
File Systems
No late submission will be accepted for this homework.

In this homework, we will use our 8080 Emulator with a more advanced OS that supports a simple file system.

Our new OS will offer the following system calls in addition to the calls of HW2. Note that you should update your ASM files to include the definitions of the new system calls. Your new OS calls are

Call	**Params to pass**	**Function**
FileCreate	Register A =12 The file name is at address BC.	It works like the file create call of UNIX systems. On fail, returns 0 in register B. On success, it returns 1. There are no owners, no access permissions.
FileClose	Register A = 13, Register B holds the file handle ID	It works like the close call of UNIX systems. It closes the file. On fail, returns 0 in register B. On success, it returns 1.
FileOpen	Register A = 14 The file name is at address BC.	It works like the file open call of UNIX systems. It returns the file handle ID in register B on success. On fail, returns 0. The file is always opened for read/write mode. There are no owners, the file pointer points to the beginning of the file.
FileRead	Register A = 15, Register BC holds the adress of the buffer. Register D holds the number of bytes to read, Register E holds the file handle.	It works like the read call of UNIX systems. It reads the file. On fail, returns 0 in register B. On success, it returns the number of bytes read. It advances the file pointer by the number of bytes read.

FileWrite	Register A = 16, Register BC holds the adress of the buffer. Register D holds the number of bytes to write, Register E holds the file handle.	It writes to the file. On fail, returns 0 in register B. On success, it returns the number of bytes written. It advances the file pointer by the number of bytes written. It append the file if needed.
FileSeek	Register A = 17, Register BC holds the seek value. Register D holds the file handle.	It moves the file pointer to the desired seek position. On fail, returns 0 in register B. On success, it returns 1.
DirRead	Register A = 18, Register BC holds the adress of the buffer to write the directory information.	We have a single directory in our system: Root. The information about the files will be read and will be put to the memory as a null terminated string.

You will use contiguous layout model for keeping your files on the disk. You should be careful about appending files and not overwriting the other files. Your block size is 256 bytes and your total disk space is 1 MB. The maximum file size is limited by the addressable seek values. You will use a single C file to keep all the file system.

You need to keep an open file table which should keep necessary information about your files. Your file handles are pointers to this table.

Your file attributes should include

- File name
- File creation, last modification and last access time
- File size in bytes.

+ file pointer for all files (open edildiğinde, file pointer dosyanın baslangıcında olucak, handler: open da return edilicek fopen dan gelen retrun gbi

You will write 4 ASM programs for your new system.

- P1: A program that creates a file (file name is given from the keyboard) writes single byte integers from 0 to 50 to this file and closes it.
- P2: A program that opens a file (name is given from the keyboard) and reads all the numbers from the file and prints them to the screen.
- P3: A program that read the root directory and prints all the information to the screen.
- P4: A program that opens a file, appends 10 numbers to the end of the file. These numbers should be the continuation of the existing numbers.

You will provide the main simulation program as follows

- Write a simulation program that runs your systems with some command line parameters. There should be parameters for the program name and debug flag.
    - **`sim8080 exe.com filesystem.dat 1`** : will read the program from exe.com and the filesystem.dat as the file system. In debug mode 1, the contents of the file table will be printed to the screen after each system call. At the end of the simulation, the directory information will be printed.
    - In Debug mode 0, the program will be run. At the end of the simulation, the directory information will be printed.

We will provide the submission instructions in a separate document. You should strictly follow these instructions otherwise your submission may not get graded.
You will submit the following files for this homework

- main.cpp
- gtuos.cpp and gtuos.h
- 4 ASM programs.
- At least 5 filesystem files on which that we can run your ASM files.