CSMC 4180
File Open Program

Write a well commented structured C open file program, not C++, that can accept up to two optional command line parameters input file name and output file name. If no command line parameters are given, the program will prompt for and process the opening of the input file and when completed prompt for and process the opening the output file. If one command line parameter is included, the program will treat this parameter as the input file and once it processed the opening the program will prompt for and process the opening of the output file. If two command line parameters are included, the program will process the first as the input file and the second as the output file.

When the input file does not exist, the program will re prompt for a input file name. If no input file name is entered when prompted for an input file name, the program will terminate. DO NOT USE BREAKS, GOTO, OR EXIT. If no file extension is given for the input file, the program will use a default .IN extension. If the input file name exists, the program will open the file for reading.

If no output file name is given after prompted for an output file name, the program will generate the output file name from the source file name with the .OUT extension. If an output file name is given but no extension is given for the output file, the program will use a default .OUT extension. If the output file exists, the program will offer a choice to overwrite the existing output file, enter a new output file name, or terminate the program. Make this choice selection user friendly. The program will back up an older output file before overwriting the file with the newer output file using the extension .BAK for the backup file. Any previous backup files will be overwritten. The output file will be opened for writing.

When the input file exists and the output file does not exist or it has been chosen to overwrite it, open the files.

In addition to the input and output file, your program will create another output file which will eventually be used as a listing file. This listing file will have the same name as the output file but with a .LIS extension.

The program will also create two temporary files. These temporary files will be used in a future program during program execution and then will be deleted at the completion of the program execution. Put some thought on how you will name this file so it does not interfere with other files existing on the computer. For this program create the files and include the code in the wrap up module to delete the file; but, comment out the code that deletes the files for program 1.

Your open file program will accept zero, one, or two parameters when the program is executed from the command line and return an appropriate enumerated completion code so the main program will know to Continue or Quit.

The program will not allow the target, source, listing, or temporary files to have the same name and same extension. The file handles and file name variables can be global for this assignment. The file close function should close only the files that are opened.

Your program will be written as a project of several files, the file processing and the main, and you will create a header file for your program.

For example:

main.c, will perform operations like calling initialization, file open, the future parser, and the wrap up.

file_util.c, will contain the functions to perform all of the file handling. Make it modular and not just one large flat function.

file_util.h will contain the #includes and constants and function prototypes for the file.c module.

Use modules i.e. file_open, get_file_name, file_close, etc.

Your main will call the procedures to open the files and if successful, the return code, read characters from the input file and write the characters to the output file, listing file, and the two temporary files until the eof is reached. Upon completion the main will call the procedure to close the files, basically a file copy. Appropriate messages will be displayed by main.

Include the program name and number, course name and number, group number, group members names and email addresses at the top of the program.
Name the files appropriately like main.c, file_util.c, file.h, etc.