60-141 – Introduction to Programming II Winter, 2017
BONUS Assignment 6

NOTE: For this assignment you will submit 1 script file. You will be writing a single C program. This assignment is optional – students who choose not to submit the assignment will not lose any marks towards their course grade. However, students who do choose to submit the assignment may receive up to 3 marks (for a fully correct program, less for a program that does not satisfy the requirements) as a BONUS to be added to their raw course mark.

Direct Access File Manipulation:

In this assignment you are asked to develop a C program that will input character data sequentially from a text file, convert the input to machine compatible formats for storage within a structure, store the structures as binary records in a direct access file and then perform a sort of the direct access file records before producing a formatted output to stdout. In addition, you will determine the total number of bytes containing in the input file and in the direct access file and output those values for comparison. You should note that character (i.e. text) files typically require much more storage than binary files due to the fact that binary files store data in the actual machine format, or representation, of various data types.

Steps:

1. In order to test your program, prepare a text file for input – you should call it **assign6.dat**. The input file must consist of at least 10 lines of data. Each line begins with a 6 digit ID that must be unique, as well as 10 arbitrary float values from 0.0 to 100.0. Each value must be separated from the following value by at least one space character, except the last one which is followed by the end-of-line. This file represents a set of student grades for each student in 10 courses (assume the same courses for all students).

The following steps all pertain to the C program you must write.

2. Declare and initialize all variables, data structures and file structures you will need for this program. You will need a structure with members for storing the unique ID (int), the 10 course marks (array of float), and the GPA (float).You will also need variables to count the number of bytes in the input text file and output binary file.

3. For each input text file record, input the entire string – use either gets() or fscanf(). Once the string has been inputted you can determine its length using strlen(). You will need to extract each of the data fields to populate (i.e. transfer data to) the structure members for the ID and marks – you might want to use sscanf() for this task. At this time you can calculate the GPA and store it in the structure.

4. Once you have populated a structure, you should then output the structure to a binary file called **assign6out.dat**. Determine the size of the structure, in bytes, using the **sizeof** pre-processor operator, and add this to the total count of bytes used in the binary file. If there is no more data in the input file you should continue to the next step, but if more data exists then you must repeat steps 3 and 4. You must not use an array to input all data; instead, you must process each input record separately before proceeding to the next.

5. With all of the data outputted to the binary file, you must now perform a sorting operation on the records, so that after the sort is finished, all the records in the binary file are sorted in ascending order by ID. Note that to perform this sort you will likely need at least 2 different structures to read in binary records to do the comparison needed, and then restore the records in required order. You may use any sort algorithm but a selection sort might be simplest. For this part, you must not use any other files – your sort must be in place (i.e. within the binary file itself).

6. The final output from your program must consist of the sorted records of student marks, including their GPA, on one output line for each record. Following this, you must output the total number of bytes inputted to the program from file **assign6.dat**, as well as the total number of bytes required to store all data in structures stored in the binary file **assign6out.dat**. Direct your final output to the file **assign6results.dat**.

Requirements:

- Write and document a complete C program that is capable of satisfying the requirements of this assignment problem.
- UNDOCUMENTED OR POORLY DOCUMENTED code will automatically lose 50% marks.
- PLAGIARIZED work will not be graded and receive a mark of ZERO and reported according to the Senate bylaws.
- The question can use I/O redirection if appropriate. Please review the textbook for an example on using I/O redirection from flat files.
- TO SUBMIT: No later than the submission deadline, your assignment should be uploaded in Blackboard. Late submissions are not accepted and will receive a mark of ZERO.
- Upload your work as an attachment, include both the source file and the script file (assign6.txt) - see below how to create the script file.

To create a script file (one that logs your compilation steps and your output in a text file):

1. script assign6.txt
2. cat assign6.c
3. ls -l
4. gcc assign6.c
5. a.out
6. cat assign6results.dat
7. ls -l
8. exit (DO NOT FORGET THIS STEP!!)

The example script execution presumes that the input was specified in the file input.txt and that all I/O is handled within the program. The commands to list all files in your directory, before and after program execution, is to show the creation of the output file by the program.

If you are compiling your code under Cygwin shell, then you need to change line 5 to:

5. ./a.exe

In line 4, students should use the gcc compiler to ensure improved warning and error diagnostic reports from compiling the program. You must have access to gcc in order to make this change, however.

NOTE: Submissions that are not received correctly by the deadline will automatically receive a ZERO

mark. **Late assignment submissions are not accepted!**

NOTES:

1. Your assignment must be RECEIVED by the due date and time. Late assignment submissions are NOT accepted. Keep your script file, and all your code unmodified as proof of its completion in case it is not received.
2. It is your responsibility to get an early start on the assignment, research and ask questions ahead of time from the due date.
3. Marks will be deducted for unclear code (improper spacing and alignment, hard to read programs and missing outputs).
4. Make sure you turn in a complete script file that clearly shows: your code, your compilation process, a listing of the directory showing your source file(s) and the a.out with the date/time stamps, and the output. DO NOT SUBMIT a.out FILES!

5. **PLAGIARISM:** CHEATING IS NOT TOLERATED – PLAGIARISM IS CHEATING! You must submit your own work. Students who are suspected of copying someone else's work will be reported to the department's chair and the Dean of Science and be dealt with in accordance with the University policies. You should not share your code with others. Codes that are similar to each other will BOTH be reported as potential evidence of copying. It is imperative that you write your own code.

6. Authorized/limited help on this assignment may be provided directly from your Lecture or Lab instructors and Teaching Assistants.