COMP10120 Practical Set 6: File Processing

Please read the questions carefully. Name each program based on your student number, the practical set number and question number. For this set (set6), question 1 should be named 1234567s6q1.c where your student number replaces 1234567. All questions that you are submitting can be zipped into a single file called 1234567s6.zip, where 1234567 is your student number and s6 refers to set 6. Please also include a readme.txt file which says which compiler you used to test your implementation. This zipped file can be submitted via Moodle for grading.

1. Write a C Program which demonstrates copying the string below to a new string using pointer notation. The new string should not contain and consonants and the letters should all be converted to uppercase. Refer to lecture notes for examples.

   char *slogan = “Comp10120 is my favourite module”;

2. Write a C Program which prompts the user for student records and then writes each record to a file called *students.txt (a* ***sequential*** *file).* The user should be prompted to enter each student’s first name, surname, student number, phone number, field of study, and GPA. Be careful to provide code to allow a user to indicate they have stopped inputting data and to check for invalid data. The result should be a text file with several student details. A separate function in your programme should be able to process students.txt and print each student’s student number and indicate if the student’s GPA corresponds to a 1st, a 2.1 a 2.2 or pass. See UCD Grading Guidelines.

3. You have a large collection of 1000 computer games. Your friends like to borrow games from you. While you are happy to do this, you find it difficult to remember who has borrowed what games. Write a sequence of C functions which will allow you to set up a system to record/catalogue the computer games that you have in a file. It should be possible to record details such as, the title, the platform, the developer and the year, but also detail if the computer game has been borrowed and by whom and when. One function should provide functionality to update these borrower details. Use **Structs** and **Random Access Files** to achieve this.