CS19101 Programming and Data Structures
Test 2

Part-I

Submit one (single) C program.

Write a C program to perform the following tasks.

1. Define a structure *Player* that will contain the following information of a cricket player (assume that they are all currently playing): players name, country, date of birth, date of debut first class match, number of tests played, number of one day internationals (ODIs) played, number of centuries scored, number of half centuries scored, total runs scored in tests, total runs made in ODIs, number of wickets taken in test cricket, number of wickets taken in ODIs, etc. (you may add some more fields).

2. Write a menu-based program to do the following to a list L of n elements of type *Player*. You may decide how to store the list L, and you may store L in more than one data structures if required:

   ▷ Given a list of Players, sort the structures according to the dictionary order of the player names.
   Take an input integer as the new number of elements for the list. If the new number is larger than the old number by m, then the data of m new players have to be inserted and the final list should be in dictionary order of the player names.
   If the new size is less than the old size by m, then the last m player information have to be deleted.
   If the new size is equal to the old size, then no changes are needed.

   ▷ Write a recursive function to take as input a positive integer, and return the number of players who have played at least those many one day internationals.

   ▷ Write a function to take as input a positive integer, and print the number of players who have played at least those many one day internationals.

   ▷ Print (with proper indentation) the current list sorted in dictionary order of player names.

Depending on what is chosen by the user, the required operation is executed by the main function.

You can write your own functions wherever necessary, but proper commenting is required to explain the purpose of the function.

Always minimize the time complexity of the program. Keep in mind that the list can be edited at any time and may require recomputation unless stored smartly.

Functions from string.h can be used.

Policy on Plagiarism

Academic integrity is expected of all the students. Ideally, you should work on the assignment/exam consulting only the material we share with you. You are required to properly mention/cite anything else you look at. Any student submitting plagiarised code will be penalised heavily. Repeated violators of our policy will be deregistered from the course. Read this to know what is plagiarism.