**Design and analysis of data structures and algorithms: Coursework Part A**

This implementation utilizes parallel two-dimensional arrays (lists in Python), each storing a particular aspect of a season, read from and written back to separate files. The main four arrays deal with accumulated points and prize money for all players over the various competitions, one of each type for male and female players. The decision to use this approach was formed from the idea that the problem was a simple one, and should have an equally simple solution. The aspects of the tournament that are to be recorded, are interrelated, and as such parallel.

Multiple temporary arrays and small stacks are used to compare the main parallel arrays, and to award points and prize money accordingly. Linear search is employed heavily during this process. An integral part of the system is a one-dimensional array (named currentWinners) which is used both in the formation of all rounds following the first round of every tournament, and in the allocation of points via comparison. The latter is achieved by borrowing the formula A **∩** B = C from set theory, where A is the set of players who have won a match in a particular round, and B is the set of all players of a particular gender. C would in this case be used as the first column ([i][0]) in a temporary array/stack, which would have previous scores combined with new scores, inserted into the second column ([i][1]).

The points are awarded at the end of each round, with the loosing players retaining the points they have been allocated, and the winners having the results of that round deducted (popped from the stack) and new points added (inserted into the stack at a given index) at the end of the next round. These points are stored in a temporary file, which is cleared at the end of the tournament. Overall rank points are calculated in the same manner, but are multiplied by the difficulty rating associated with whatever tournament they are playing in, with any new results are added to their previous results, and saved in a permanent file.

Prize money is also allocated in a similar manner, with winning players being allocated one of four pay brackets (low, mid, runner-up, and winner) depending on where they leave the tournament. As above, the money they win is added to any money already won.

To display the information, a quick sort algorithm was adapted for use with two-dimensional arrays. This was used for displaying both sets of rank points, and prize money, in descending order. Quick sort was used because of its average case time complexity of O(n log n). Although bubble sort and insertion sort are arguably easier to implement, and share a worst case complexity of O(n2) with quick sort, their average complexity is also O(n2), making quick sort more agreeable. However, as the input is quite small, it perhaps makes little difference.

Input is dealt with in two ways. The first is through automatically generated random results within a certain range, the second uses user input. The latter is achieved via a stack, containing all the players of a chosen gender. When the user enters the player they want, that player is popped from the stack of available players. This process is repeated until the stack is empty, and the round may continue. The following round, sees the stack re-populated by the winners of the previous round