CMIS 2720 Data Structures and Algorithms for Games

Instructor: Ying Zhu

Assignment #5

Due date: April 14, 2023

In this assignment, you will learn the following:

- How to implement a sorting algorithm
- How to implement a binary search algorithm

General information

1. You must write your programs in C#.
2. This assignment contains two related parts (A5a and A5b), and you can combine them into one program.
3. Submit a zip file to iCollege under the folder Assessments → Assignments → Assignment5. The zip file should contain one or two C# files:
    a. firstname_lastname_A5a.cpp
    b. firstname_lastname_A5b.cpp
4. Write comments in your code. This part will be graded. If there is no comment, I will deduct 5 points.

A5a requirements (70 points)

1. In this assignment, you will learn to implement a slightly more sophisticated sorting algorithm.
    a. The program must read a CSV file that contains some gamer analytics data. The CSV file can be found on iCollege alongside this document.
        i. I will provide a sample program to show how to read a CSV file.

    b. The program should provide a console-based UI that allows users to request sorting the data by one column or two columns. Your program should be able to process the following requests.
        i. **"sort_by name time"** (Sort by name first and then by time. So you see which game each player spends the most time on.)

        ii. **"sort_by name IAP"** (Sort by name and then by IAP. So you see which game each player spends the most money on.)

        iii. **"sort_by game time"** (Sort by game and then by time. So you see for each game who spends the most time on it.)

iv. **“sort_by game IAP”** (Sort by game and then by IAP. So you see for each game who spends the most money on it.)

v. **“sort_by name”**

vi. **“sort_by game”**

vii. **“sort_by time”**

viii. **“sort_by IAP”**

2. The output for each query should be the entire spreadsheet with the rows sorted properly.

3. You must implement the sorting algorithm yourself. Do not call a sorting function from a C#/.Net library.

4. You can only use one of the three sorting algorithms: **quick sort, merge sort, or heap sort**.

5. Do not try to implement a separate sorting algorithm for each sorting request (e.g., one sorting method for “sort by name IAP” and a separate one for “sort_by name”). Instead, try to implement a sorting algorithm that can handle different sorting requests.

A5b requirements (30 points)

1. Implement a **binary search algorithm** so that your program can process the following requests:
   a. **“find_name aaaa”** (Display the records for player xxxx.)

   b. **“find_game bbbb”** (Display all the records for game yyyy.)

   c. **“find_time c”** (Display all the records with the playtime greater or equal to c.)

   d. **“find_IAP d”** (Display all the records with the IAP amount greater or equal to d.)

2. You must implement the search algorithm yourself. Do not call a search function from a library.

3. You must use binary search. **Do not use linear search because it is not efficient.**

4. Note that you must sort the records first before using the binary search.

5. Do not try to implement a separate search algorithm for each search request. Instead, try to implement a search algorithm that can handle different search requests.