CSC 241 Assignment 5
Abstract Data Types and Programming Methodology
Due: Thursday, April 14

1 Introduction

In this semester, we are going to develop a Java program called "***GradeManager***", which manages students' grades. This program provides functions such as adding/deleting/editing grades of certain students. All the grades are stored in designated file(s) and updated as grading data change. It also supports to manage students, *e.g.*, adding a new student and his/her grading data or deleting a grading record of a particular student. This program will be built up through several assignments, in each of which you will be asked to apply what you will learn in lectures. At the end of semester, you will have a Java program which utilizes various OOP techniques and diverse data structures.

The assignments will be handed in an order for completing a final program. So, you MUST follow instructions and achieve requirements when you work on an assignment. Java code for each assignment should be errorless and submitted in the Blackboard course shell. Since a next assignment usually asks you to add more functions or edit what have been made in the prior assignment, you should keep the previous Java code(s). If the prior program is submitted with errors or runs unsuccessfully, it must be corrected before it goes to the next assignment.

2 Goal for This Assignment

In assignment 4, you implemented two searching methods – linear search and binary search to find a student by name, and then benchmarked them to see which method was better. As you learned from lectures, the benchmark has resulted that binary search defeated linear search in average cases (linear search may win the competition in some cases, but not frequently). However, binary search should work only with ordered data. To maintain data listing in order, you will implement one of five sorting algorithms introduced in class in this assignment. And you will develop two more functions – add/remove student's grade while you are running the *GradeManager*. For this purpose, you will need to construct an ***ArrayList*** which is the first dynamic data structure you learned this semester. You need to review lecture 9 which addressed ***ArrayList*** and gave you an example. Binary search method can be used to maintain order of data, as data add up or disappear. Extra work is provided in this assignment, thus read the section 4 carefully.

3 Instructions

A. Template file

Each assignment should be built in a package. The names of package and class for this assignment are below.

Package: Assignment5

Class: GradeManager

```
package Assignment5;
        …
public class GradeManager {
        …
```

Java files and data files are zipped under the name of this package. You first unzip the **Assignment5.zip** which is uploaded in the Blackboard course shell, and check all necessary files. DO NOT change the package name or class name if you want to use the template files.

However, you may want to use the files you previously made, since this assignment is extended from the prior assignment. In this case, please duplicate them and start working on them. Do not forget that change the name of the package.

B. Data File / Properties File

Json file includes 20 students' grade data. But it will increase or decrease as a new student's data are added or deleted.

From the assignment 3, `config.properties` file has been used to determine a full path for json file. Once you set the file path correctly, you do not need to update.

C. Developing Environment

Your program should be **implemented in Java only**. The program in another language will not be graded.

D. Submission

You will submit your Java package. Zip the package **Assignment5** and upload it in the Blackboard course shell. DO NOT copy and paste your code into text files such as rtf, doc, or txt. You MUST submit java files, not text files! The assignment will give you **two weeks** so it is **due on Thursday, April 14**. All submission **by 11:59 PM** on that day will be accepted without any penalty. On the due date, Blackboard may be suffering of too much network traffics and be unstable. There is no excuse about the issue, therefore you are strongly recommended to submit earlier than the due date.

4 Requirements

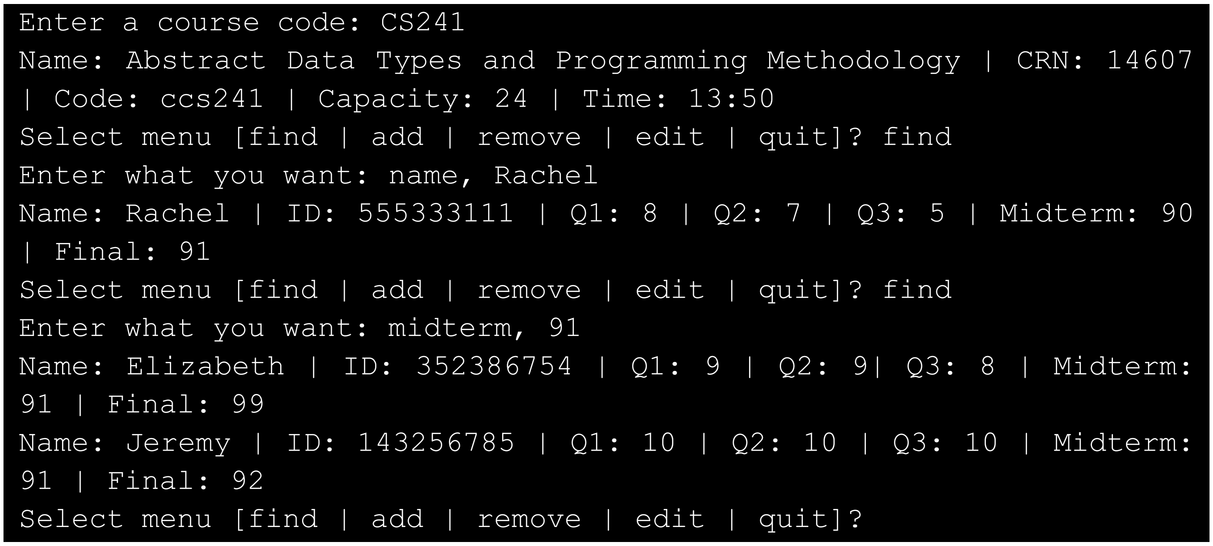
A. Complete all the requirements in the assignment 3 and 4

Regardless of new requirements for this assignment, your program should be able to do what were requested in the assignment 3 and 4. If you have already completed all requirements in the assignment 3 and 4, you would concentrate on the new requirements. Otherwise, you must complete them first because they are required for the assignment 5.

However, you do not need to do the benchmark for searching algorithms which was requested in the assignment 4. You will use only binary search for finding information.

B. Implement one sorting algorithm

Binary search algorithm, which was implemented in assignment 4, needed ordered data. In this assignment, thus, you will implement a method to sort random data for binary search method. The sorting method will be called first when you find a student's information. For example, when your program receives a request to find a student by name, it needs to sort data first by name and then run binary search method. If you want to find student(s) who get a particular score for a course work, then you must sort the data by the course work as a key. **Figure 1** illustrates how to find a student by a key and its value. Key could be any of information for the student, i.e., id, name, q1, q2, and so on. Note that searching method may return more than one student's information. In **Figure 1**, there are two students who get 91 points for midterm: Elizabeth and Jeremy.



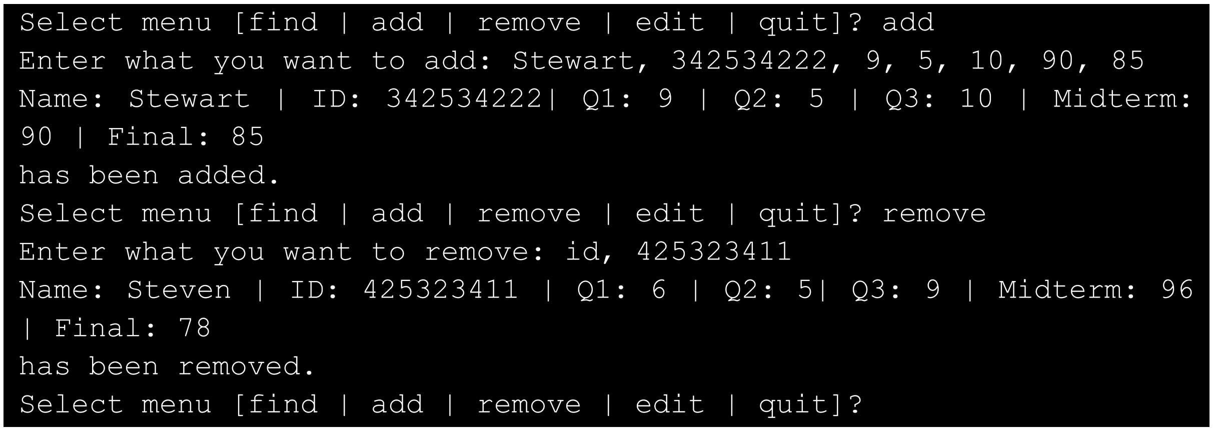
```
Enter a course code: CS241
Name: Abstract Data Types and Programming Methodology | CRN: 14607
| Code: ccs241 | Capacity: 24 | Time: 13:50
Select menu [find | add | remove | edit | quit]? find
Enter what you want: name, Rachel
Name: Rachel | ID: 555333111 | Q1: 8 | Q2: 7 | Q3: 5 | Midterm: 90
| Final: 91
Select menu [find | add | remove | edit | quit]? find
Enter what you want: midterm, 91
Name: Elizabeth | ID: 352386754 | Q1: 9 | Q2: 9| Q3: 8 | Midterm:
91 | Final: 99
Name: Jeremy | ID: 143256785 | Q1: 10 | Q2: 10 | Q3: 10 | Midterm:
91 | Final: 92
Select menu [find | add | remove | edit | quit]?
```

Figure 1. Your program has five menu – find, add, remove, edit, quit. When you want to find a student's information, you will provide key and value which will be used for searching.

There are many options for sorting algorithms. You learned 3 algorithms – selection, insertion, and bubble which conducts sorting in $O(n^2)$ and 2 algorithms based on divide-and-conquer paradigm which runs in $O(nlogn)$ – merge and quick. You are free to choose any of the algorithm, but those who implement a divide-and-conquer algorithm, will have extra credits of 10% (3 points).

C. Add/remove data

In the previous assignments, your program did not add a new student's information or delete a student's data. It just allowed you to edit an existing information. Since the size of the data (of students) did not change while running the program, static data structure such as array could be used. In the assignment 5, you will add or delete students' data. **Figure 2** shows how to add and how to delete.



```
Select menu [find | add | remove | edit | quit]? add
Enter what you want to add: Stewart, 342534222, 9, 5, 10, 90, 85
Name: Stewart | ID: 342534222| Q1: 9 | Q2: 5 | Q3: 10 | Midterm:
90 | Final: 85
has been added.
Select menu [find | add | remove | edit | quit]? remove
Enter what you want to remove: id, 425323411
Name: Steven | ID: 425323411 | Q1: 6 | Q2: 5| Q3: 9 | Midterm: 96
| Final: 78
has been removed.
Select menu [find | add | remove | edit | quit]?
```

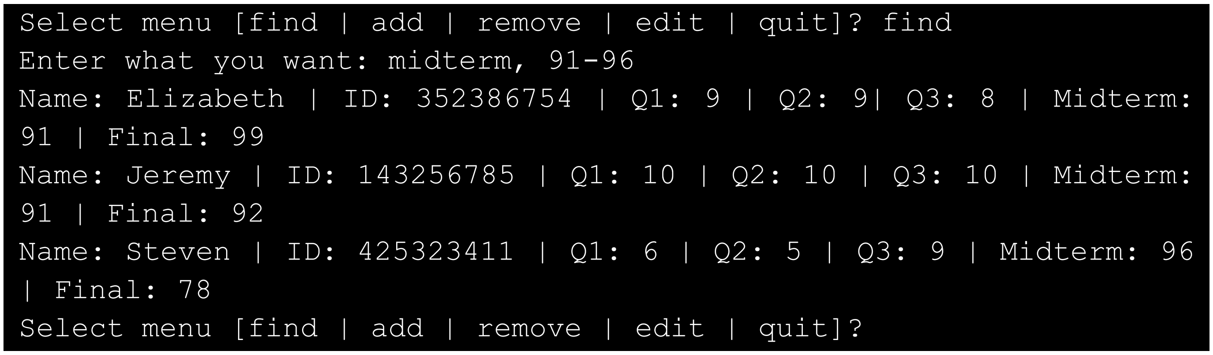
Figure 2. When you add a new student's information, enter all information. When you remove a student's information, find the student first and remove his/her from the data.

When you remove data, you need to provide a key and its value. In the **Figure 2**, a student, whose id was 425323411, has been removed. Note that the remove function may remove multiple students. For example, if you request to remove with (q1, 10), more than one students' information will be deleted.

The addition and removal of a student affect the Json data file. When you add a new student, the Json file should have the student's information. After deleting a student, the student's information should be gone from the Json file.

D. [Extra Work]Enhanced conditional search

In **Figure 1** and **Figure 2**, a student was searched by a specific and single condition, for instance, being searched by midterm score 91. It can be extended to have range of scores such as 91 to 96. The ranged condition can be formed in this way: ***minimum score-maximum score***. If you want to find students whose midterm is greater than and equal to 91 and less than and equal to 96, you need to enter the condition like "91-96" as shown in **Figure 3**.



```
Select menu [find | add | remove | edit | quit]? find
Enter what you want: midterm, 91-96
Name: Elizabeth | ID: 352386754 | Q1: 9 | Q2: 9| Q3: 8 | Midterm:
91 | Final: 99
Name: Jeremy | ID: 143256785 | Q1: 10 | Q2: 10 | Q3: 10 | Midterm:
91 | Final: 92
Name: Steven | ID: 425323411 | Q1: 6 | Q2: 5 | Q3: 9 | Midterm: 96
| Final: 78
Select menu [find | add | remove | edit | quit]?
```

Figure 3. With the ranged condition, searching method may find more students satisfying the condition. In this example, three students

It is not mandatory. If you finish all requirements and want to do more, you may do the extra work. Those who complete it, will have extra credits up to 20% (which is worth 6 points). For this function, you should use binary search.

5 Grading

A. Grading criteria

The lab is assigned **30** points, which is 10% of the final grade. It will be graded by evaluating the requirement. Any missing and unsatisfiable criteria will take off points. The tentative and brief criteria are below.

Compilation:	**5** points
Execution:	**5** points
Proper output:	**20** points (extra **3** points for divide-and-conquer algorithm)
Extra work:	**6** points

B. Late penalty

Late submission will take off **2 points per day** after due date. **Thus, submission after 15 days will not be accepted in any circumstances**.

6 Academic Integrity

Any dishonest behaviors will not be tolerated in this class. Any form of plagiarism and cheating will be dealt with according to the guidelines on the Academic Integrity Policy on-line at http://www.oswego.edu/integrity. For more information about university policies, see the following online catalog at:

http://catalog.oswego.edu/content.php?catoid=2&navoid=47#stat_inte_inte

Student who is against the honor code will not have any credits in this project.