## Submission Instructions and Guidlines

- Your submission must consist of a single zip file that contains the following .java files only: PhonebookEntry.java, and Phonebook.java. No additional files should exist in the .zip file/folder.
- Give meaningful names to your methods and variables in your code.
- Include a comment at the beginning of your program with basic information about yourself and a description of the program. Include also a comment at the start of each method.
- The name of the zip file must adhere to the following naming convention finalExam_netid, where netid stands for your AUBnet user name. For example, if your AUBnetid is abc65, you should submit the following file: finalExam_abc65.zip.
- Failing to follow these guidelines will result in deducting marks form your grade.


## PhonebookEntry.java \& Phonebook.java (100\%)

In this problem, you will need to create a program that emulates a phonebook. Specifically, you will first create the data type PhonebookEntry, which stores and maintains information about a phonebook entry. You will then create the data type Phonebook that keeps track of all entries such that duplicate entries are allowed. Finally, you will create a program that allows the user to manipulate the phonebook through a command line menu, which will test all the methods that you will need to implement. The class PhonebookEntry is represented as follows:

| public class PhonebookEntry |  |
| :--- | :--- |
|  | // Private Fields <br> firstName: String <br> lastName: String <br> phoneNum: int // the phone number would have 8 digits (e.g., 01350000) |
|  | // Full constructor <br> PhonebookEntry (String fName, String IName, int num) |
| void | // Changes the number of the current object to newPhoneNum <br> changeNumber (int newPhoneNum) |
| void | // Changes the last name of the current object to newLastName <br> changeLastName(String newLastName) |
| boolean | // Returns true if the object has the same last name as lastName (case sensitive) <br> hasSameLastName (String lastName) |
| // This method compares this PhonebookEntry object to a passed PhonebookEntry object p. One <br> PhonebookEntry is considered to be less than another PhonebookEntry, if the firstName in the <br> first object is less than the firstName in the second object, and if the lastName in the first object <br> is less than the lastName in the second object, and if the phoneNum in the first object is less than <br> the phoneNum in the second object. Two PhonebookEntry objects are equal if all their private <br> fields are equal. If the first PhonebookEntry object is not less than the second PhonebookEntry <br> object and not equal to it, we consider that the first PhonebookEntry is greater than the second <br> one. Return the number 0 if both PhonebookEntry objects are equal. -1 if the given object is less <br> than the passed object. +1 otherwise. <br> compareTo(PhonebookEntry p) |  |
|  | // String representation of a an entry with the following format: <br> // FirstName LastName: xx-xxxxxx (Note: you need to format the number properly) <br> // Example: Lionel Messi: 71-000017 <br> toString() |
| String |  |

The class Phonebook is represented as follows:

| public class Phonebook |  |
| :---: | :---: |
|  | // Private Fields <br> phoneBook: PhonebookEntry [] // an array of phonebook entries <br> currentEntry: int // keeps track of the next available entry in the phonebook. <br> // Also used to know where to insert an entry. <br> size: int // size of the array |
|  | // Full constructor, takes the size of the array as parameter, and sets it to the corresponding field <br> // Initializes the array accordingly, and sets the currentEntry to 0 <br> Phonebook(int size) |
| boolean | // Returns true if the phonebook is full isFull() |
| boolean | // Asks the user for a phonebook entry <br> // Adds the entry into the phonebook. Returns true if the add was done successfully; <br> // If false is returned, no entry is added to the current object. <br> // Hint: an entry cannot be added if the phonebook is full <br> // The method must create a new entry and fill its information using the Scanner object <br> add(Scanner console) throws IOException |
| void | // Prints out all entries that have the last name lastName with the following format. <br> // For example if the phonebook contains three entries, the following will be printed: <br> // O. Lionel Messi: 71-000017 <br> // 1. Cristiano Ronaldo: 03-923219 <br> // 2. Eden Hazard: 70-111100 <br> searchAndPrint (String lastName) |
| int | // Returns the number of entries with the last name matching lastName. searchNumMatches (String lastName) |
| int | // Returns the index of the first entry in the phonebook with lastName. // If no such entry exists, -1 is returned. <br> getIndex (String lastName) |
| void | // Changes the phone number of a phonebook entry at index, to newNum changeNumber(int index, String newNum) |
| void | // Asks the user for the entry, to change its phone number based on the last name. <br> // If the entry's last name is not available, the method must notify the user <br> // If one entry exists, the method must ask the user to input the new phone number, <br> // and then perform the change <br> // if multiple entries exist, the method must ask the user to specify which index/number <br> // to change its phone number <br> doChangeNumber (Scanner console) throws IOException |
| Phonebook | // Returns a new Phonebook with all its entries sorted in descending order. Do not sort the existing Phonebook sort() |
| String | // String representation of the phonebook with the following format. <br> // For example if the phonebook contains three entries, the sting representation will be as follows // (with one entry per line): <br> // 0. Lionel Messi: 71-922234 <br> // 1. Cristiano Ronaldo: 03-923219 <br> // 2. Eden Hazard: 70-111100 <br> toString() |

Write a main method (in the same class as Phonebook) that provides the user with a menu to create and manipulate a phonebook.

Sample run: the program will first ask the user to enter the size of the phonebook (e.g., 10) as follows:
Enter the size of the Phonebook: 10
It then keeps printing the following menu and waiting for the user's input (as long as the input is not 5):
Menu:

1. Add an entry to the phonebook.
2. Print out all phonebook entries.
3. Search for an entry.
4. Change the phone number of an entry.
5. Quit.
```
If the user enters 1 (and the phonebook is not full), the following gets executed:
What is the first name of the entry?
Lionel
What is the last name of the entry?
Messi
What is the phone number of the entry?
71000017
The entry has been added.
If the user enters 1 (and the phonebook is full), the following gets executed:
Sorry, the phonebook is full, no entry can be added.
```

If the user enters 2, the following gets executed (assuming that the phonebook has three entries):
0. Lionel Messi: 71-000017

1. Cristiano Ronaldo: 03-923219
2. Eden Hazard: 70-111100

If the user enters 3, the following gets executed (assuming two entries for Rona Ldo exist):
What is the last name of your search?

## Ronaldo

Here are the entries that matched your search:
0. Cristiano Ronaldo: 03-923219

1. Luis Ronaldo: 01-125212

Again, if the user enters 3 , the following gets executed (assuming Zidane has not been added):
What is the last name of your search?
Zidane
Sorry, no entries matched your search.

If the user enters 4, and no entries for Ronaldo exist, the following gets executed:
What is the last name for the number you want to change?
Ronaldo
Sorry no entries match that last name.
If the user enters 4, and only one entry for Ronaldo exists, the following gets executed:
What is the last name for the number you want to change?
Ronaldo
Exactly one entry matched.
0. Cristiano Ronaldo: 03-923219

What is the new phone number?
03123219

```
Number changed.
If the user enters 4, and more than one entry for Ronaldo exist, the following gets executed:
What is the last name for the number you want to change?
Ronaldo
Here are the matching entries:
0. Cristiano Ronaldo: 03-923219
1. Luis Ronaldo: 01-125212
Which entry would you like to change?
1
What is the new phone number?
03123219
Number changed.
```

If the user enters an invalid menu option (e.g., 10), the following gets executed:
Sorry, invalid menu choice.

