Metropolitan State University, St. Paul
Department of Information and Computer Science
ICS 240 Data Structures
Fall 2017

Program 1 Lists of Things

50 points

Requirements

For this program you will be asked to write a list management program. Start by choosing what sort of list you would like to keep. For example,

- a class list (list of students)
- a bibliography (list of books and articles)
- a list of dream vacations
- a transcript (list of courses)
- a list of top ten films
- pretty much any sort of list

You can be creative about your list. Feel free to search the internet for examples of lists if you like. Do not choose a Grocery list since parts of a working example of a Grocery list will be provided for you. It may be possible to use this idea and reuse code across more than one programming assignment, so try to choose something that interests you.

Your list will contain the Things of your choice.

1. Implement the sort of Thing to be stored on your list as a class. The object/class must have the following characteristics:
   - At least 5 attributes/characteristics
   - At least one numeric attribute
   - At least one String attribute
   - One String attribute should be usable as a search key
2. Create the list by implementing a collection of the objects using an array. You may NOT use the library class ArrayList.
3. Be able to choose the size of the list. That is, implement a constructor which accepts as an argument the list size.
4. Allow the user to enter data and store it in the list.
5. Display the complete list. Reflect all changes. That is, deleted items should no longer be displayed.
6. Given a search key by the user, search your list for a matching object. Retrieve and display all information for the matching object. The search key should be CASE INSENSITIVE. For example, “ITEM” should be a match for “item” or “IteM”.
7. Given a search key by the user, search your list for a matching object. Delete the matching item.

8. Given a search key by the user, search your list for a matching object. Present the data and allow the user to change any field or fields for that item. You must NOT require that the user retype information which has not been changed. (Hint: Find, delete, add.)
9. Check for errors such as entering text in a numeric field or entering more values than can be held in the array of objects. Gracefully recover from any problems without crashing the program. (See Appendix C for more on exception handling.)
10. Use of a graphical user interface such as swing is required.

Grading Criteria

1. Design (10 points)
   - Provide a UML class diagram for the project which includes a detailed diagram for each class as well as arrows and lines which correctly show the relationships between classes.
   - Methods are of appropriate length
   - Classes are factored appropriately
   - Appropriate private and public access modifiers are used
   - Other

2. Correctness (30 points)
   - User interface is intuitive, easy to use, attractive.
   - Meets program specifications as described above
   - The program is robust with no runtime errors or problems
   - Other

3. Style (10 points)
   - Program is readable
   - Javadoc comments (see p. 7 and Appendix H)
     - Comments on the class as a whole including
       - Description of program
       - Your name (use @author)
       - Date (due or date you started or last modified)
       - Source of any “borrowed” code
     - Comments on each method including
       - Concise description of what method does
       - Arguments for each method (use @param)
       - Returned value (use @returns)
       - Exceptions which are thrown (@throws)
   - Consistent and correct indenting
   - Meaningful identifiers
   - Normal capitalization conventions are used
   - The source of any "borrowed" code is clearly identified
   - Other

Upload your Solution via D2L

There is a description of how to do this in the Supplemental materials section if you need information on how to do this.