

**DUE: 5/15/2012**

## *Project #2—Golf Database*

Your golf coach has asked you to design a quick and easy program to keep track of all the players on the golf team. Currently no such program exists and it is difficult to contact golf players and to place them with appropriate ranking.

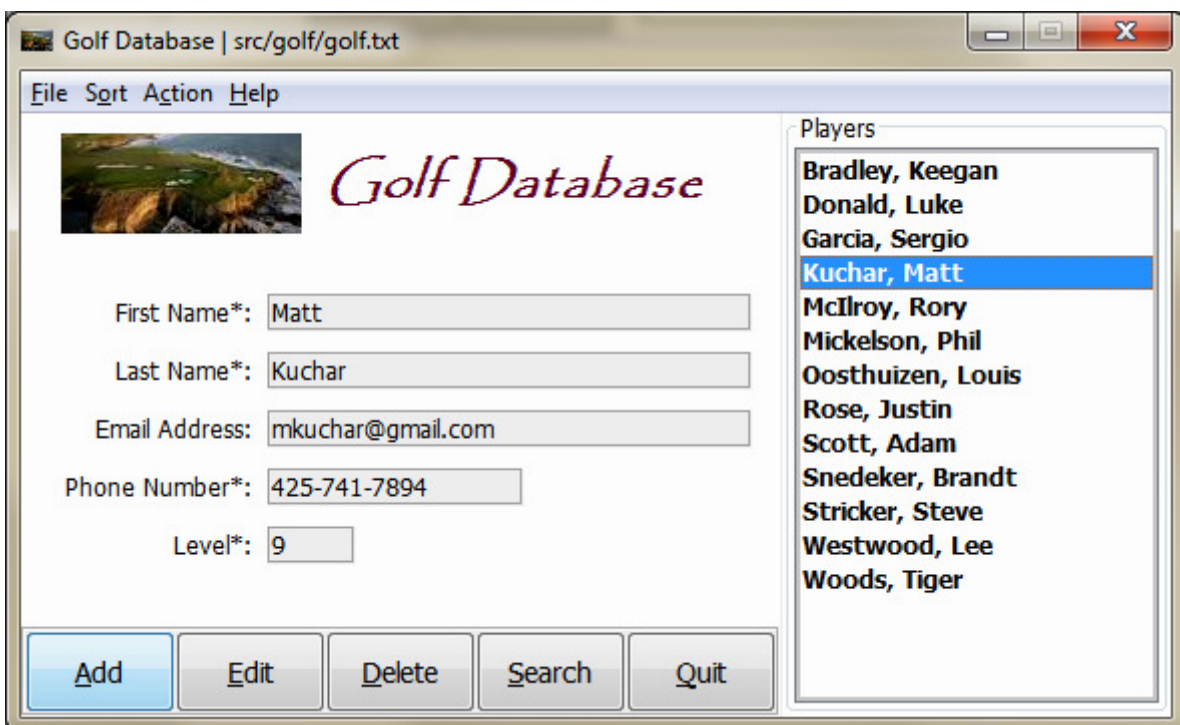
Write a Java program that allows the user to manage a simple golf database with ability to display, add, edit/modify, and delete members of the golf team. Sample GUI is provided as one example of an implementation. There is a lot of flexibility however for your vision of how to design and implement this database, as long as your implementation has all of the required functionality.

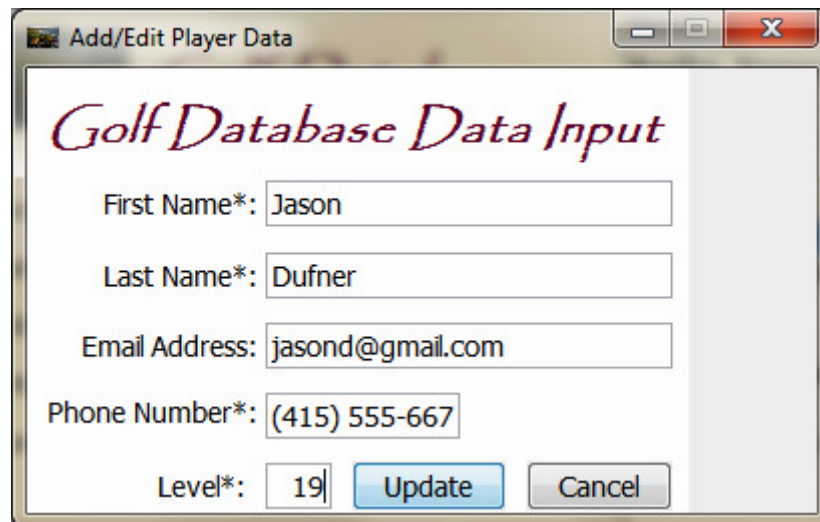
Your program must contain/do:

1. A Member class that contains in order at least first name, last name, email, phone and level (team rating of 1-5, with 1 being the top seed).
2. Read and display the information about the members in a GUI-based driver. You can decide for yourself the back end: it can be comma delimited text file, Access or Excel file, SQL, XML, or anything else you wish. An example of data with comma delimited text file might be:  
  

```
Tiger,Woods,tiger@gmail.com,206-111-1111,1  
Rory,McIlroy,theroy@yahoo.com,425-799-1482,2  
Adam,Scott,ascott@yahoo.com,206-559-3398,3  
Justin,Rose,,206-425-4775,4  
Brandt,Snedeker,lovetogolf@hotmail.com,206-849-2592,5
```
3. Ability to add new members, modify existing ones, delete and search members. Checks should be performed not to add duplicate members, and that all of the 4 required fields (indicated with an asterisk (\*) in the provided GUI) are valid and included when adding/modifying a member. Note that email is not a required field; all the rest are required.
4. A data structure of Members to hold the members data. This could be an array, ArrayList (preferable), vector, or any other structure you desire.
5. Javadocs, description of the program, and comments.
6. Menus that synchronize with corresponding buttons and with at least the following menu choices:
  - File with New, Open, Save, Save As, Clear, Print, and Exit menu items.
  - Sort with Sort by Last Name and Sort by Rank menu items.
  - Action with Add, Delete, Edit (with choice to save or cancel changes), and Search menu items.
  - Help with About menu item for an About form.

7. The project should start with a Splash Screen that closes itself after so many seconds and it should contain an About form activated from the Help menu.
8. Display the members sorted by last name or sorted by rank.
9. Well designed and efficient GUI with images and ease to use.
10. Print the information for a selected member.





Add/Edit Player Data

*Golf Database Data Input*

First Name\*: Jason

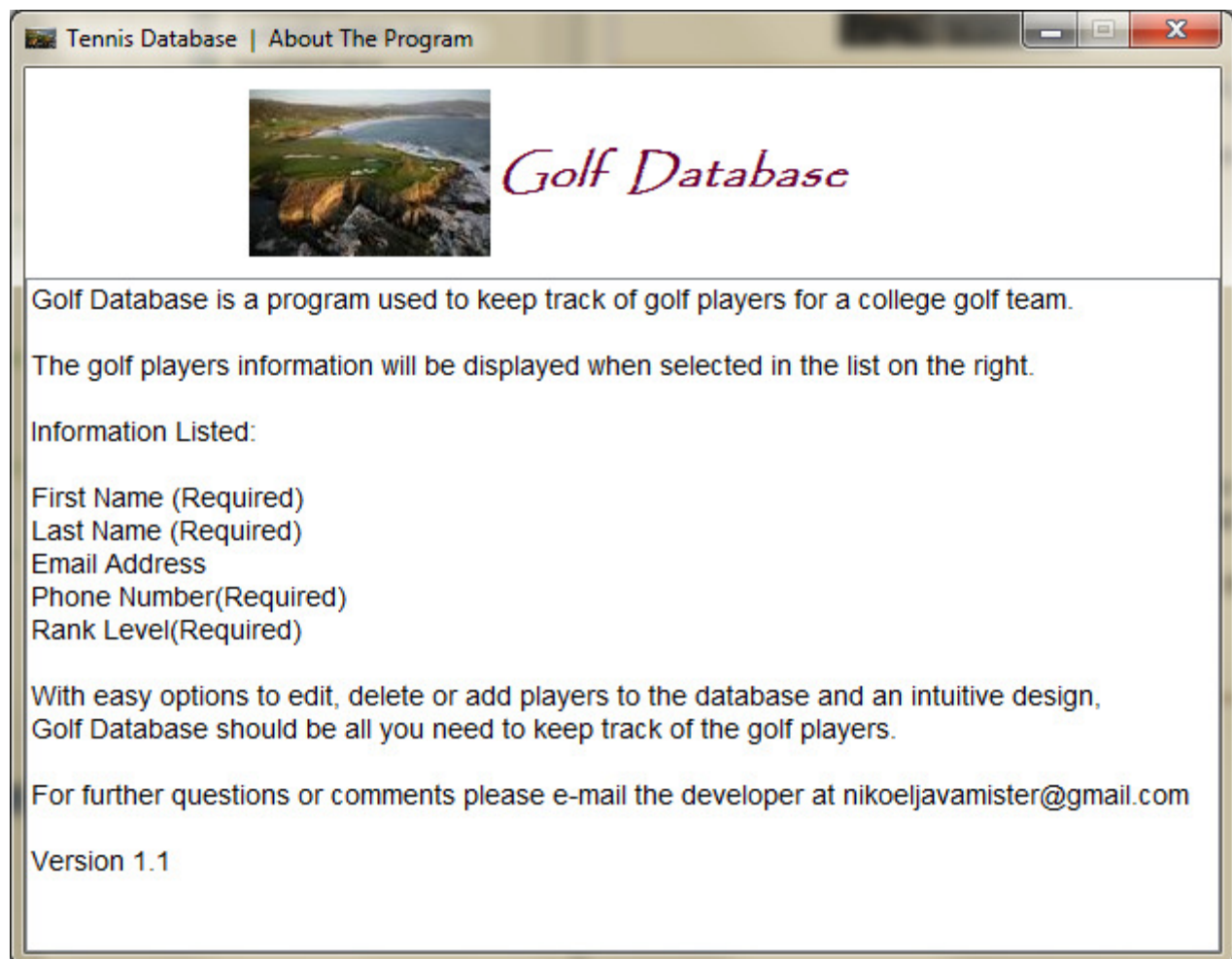
Last Name\*: Dufner

Email Address: jasond@gmail.com


Phone Number\*: (415) 555-667

Level\*: 19

Update Cancel



Tennis Database | About The Program

 *Golf Database*

Golf Database is a program used to keep track of golf players for a college golf team.

The golf players information will be displayed when selected in the list on the right.

Information Listed:

- First Name (Required)
- Last Name (Required)
- Email Address
- Phone Number(Required)
- Rank Level(Required)

With easy options to edit, delete or add players to the database and an intuitive design, Golf Database should be all you need to keep track of the golf players.

For further questions or comments please e-mail the developer at [nikoejvami@gmail.com](mailto:nikoejvami@gmail.com)

Version 1.1

EXTRA CREDIT (Optional). For many additional points improve the project by adding the following features/changes:

- Add additional but pertinent classes (for example, for reading/writing from/to the database, sorting, validating input, searching, etc.).

- Add separate forms for adding/editing members to the database.
- Modify the program not to allow members with same rank.
- Add pictures of each player when selected.

**Project 3 Grading Guidelines:** Your project will be graded on the following:

- Correct solution to the proposed problem.
- GUI design (as outlined above).
- Elegance, simplicity, style, and readability of the algorithms and code.
- Comments:
  - ✓ Use of header.
  - ✓ Description of project.
  - ✓ Description of procedures.
- 30 points maximum. Extra credit points are optional and not required.

The best advice I can give is to ***draw a detailed flowchart prior to coding this project***—it saves a lot of headaches later! Other than that, here are few hints:

- ⇒ Create an ArrayList of Members when reading the members' data from the external text file.
- ⇒ Consider using a JListbox to list the names of the golf members when sorted by last name or name and level when sorted by ranking.
- ⇒ Selecting a member from the JListbox should display all the information about that member on the main form.
- ⇒ Use a JSpinner control for the rank level with appropriate bounds and access.
- ⇒ Make sure you synchronize the menu choices with the corresponding buttons.
- ⇒ Enable and disable buttons and menus with appropriate context.
- ⇒ Make changes to the database dynamic. For example, if a member is deleted, that member should not show up in the JListbox after being deleted.
- ⇒ Check your invoice against known results (see screen captures above and below)
- ⇒ Make the appropriate buttons and menus available and disable them when they should not be available.
- ⇒ Create ReadFile and WriteFile classes to read/write to the database.
- ⇒ Provide two sorting methods for sorting by level and sorting by last name.
- ⇒ Provide displayMembersList() method that displays only Last Name, First Name after the sorting method for last names is called and displays Last Name, First Name, and rank after the sorting method on ranks is called.
- ⇒ Consider makes rankings unique.
- ⇒ Create a Member class and ArrayList of objects from the Member class for your main database.



## *Grading Rubric for Project 2*

```

/*~~~~~
'Comments by the prof:
'Great effort. Here are suggestions for improvement:
'1. Use comments throughout the program, not just at the top heading.
'2. An image of a golf logo (or other) would enhance the look of the form.
'3. Did not implement printing of selected member--it is required.
'4. Make the Add New Member button default.
'5. Make the level JSpinner read-only when displaying the member information.
'6. Validate email, rank and phone number entries. Consider using
JFormattedTextFields.
'7. It is good style to tab code inside a method.
'8. Provide buttons and menus for adding, deleting, editing, printing and exiting the
application.
'9. Declare all constants if there are any.
'10. Clear the JListbox display box before you use it and populate it with members as
the form loads.
'11. Change the tab sequence to indicate correct order of entry.
'12. Name the classes and project appropriately.
'13. Provide check for member duplication--do not allow two members in the database
with identical information.
'14. Follow the Java naming convention for naming variables, methods and classes.
'15. Read the members data from an external file or database.
'16. Include a separate golf Members class for the project.
'17. Check for valid phone and email--consider using JFormattedTextField for them.
'18. Add new non-existing member only if all four required fields are valid and
included.
'19. Enable menu choices that synchronize with buttons' functionality.
'20. Provide a sound data structure for the Members.
'21. Sort the data with two different sorting algorithms by last name and by level.
'22. Disable maximization of the form.
'23. Add an About form which describes the project.
'24. Add a Splash screen that starts the project, displayed in the middle of the
screen.
'25. Provide ToolTip to help user navigate through the form.
'26. Consider separate forms/classes for adding/editing a member.
'27. Highlighting a golf member in the JListbox should display that member's
information in the corresponding text boxes.
'28. Make changes to the database dynamic. For example, if a member is deleted, that
member should not show up in the JListbox after being deleted.
'29. Add ability to search for a member in the database.
'30. Add icons to all forms.
'31. The Member class needs in addition to the instance variables at least two
constructors, set and get methods, equals and toString methods.
'32. Verify that required fields are entered before a member is added to the team.
'33. Avoid duplicating a member.
,
'The ones that apply to your project are:
,
'4, 5, 12, 25
'26+1=27/30
'~~~~~*/

```