Registration system for sports club

The assignment is to program a system for registering data about sports, groups and members in a sports club. There is no particular API or class description to fill in and no part of the tests will be automated. The assignments will be evaluated based on functionality that works through the user interface. Code will be evaluated based on organization of classes and operations and what data structures are used. Operations that do not work in the user interface will not be evaluated for partial points based on code. User interfaces are evaluated based on efficiency and clarity of information, not on beauty. Text-based UI is sufficient.

Base functionality - 80%

- User shall be able to register new sport
    - Name of sport
- User shall be able to register new member
    - Name
    - Phone
    - Email
    - Year of birth
- User shall be able to sign member up for a particular sport
- User shall be able to remove member from a sport
- User shall be able to remove a member from the system
- User shall be able to remove a sport from the system
- User shall be able to see a list of all members and order it on different data
    - Ordered by name
    - Ordered by age
    - Ordered by sport
- User shall be able to retrieve members by different data
    - Get by name
    - Get by phone
    - Get by email
    - Get by age or year of birth
- User shall be able to select a member and see detailed information about them
    - Name, phone, email, year of birth
    - All sports this user is registered in
- User shall be able to see a list of sports
- User shall be able to select a sport and see detailed information
    - List of users in the sport

Operation memory and undo - 5%

- The system shall keep track of every operation done in the system
    - At least operations that change data
- The user shall be able to undo the last operation
    - Then the next-to-last, etc… like undo usually works

Groups in sports - 5%

- User shall be able to register groups in each sport
    - Group name
    - Age from
    - Age to
- When signing a user up for a sport a group must be selected
    - Only groups for correct age available
    - Possible to sign into more than one group, if age allows
- When viewing details for a sport the user should see a list of groups
- User should be able to select a group and see details
    - List of users in a group
- When viewing details on a user the sport and group should be displayed in list

Waiting lists - 5%

- The user shall be able to register a maximum number of members in a group (*or sport*)
- If a group (*or sport*) is full a member is added to waiting list instead
- If member is removed from a full group (*or sport*) a member from the waiting list is added
    - Notify the user when this happens

Persistent data - 5%

- The system shall store all data in files
    - Store when system is closed
        - prompt?
    - Store when user chooses to save
- The system shall read everything from the files when the system starts
    - Read data into more efficient data structures in memory

Efficient data structures - 5%

- Efficient data structures should be used for each function of the system
    - Not everything in lists that are then sorted over and over
        - Use sorted structures when needed
        - Use multi-key structures where applicable
            - But not when not needed
    - Use simple structures like stack and queue when sufficient

Efficient flow of operations - 5%

- Make your system pleasant to use :)
    - Quick selections
        - Have numbered lines rather than having to enter full strings
    - Clear user interface
        - Windowed UI not required
            - Operation can still be made clear and obvious