There are a few points worth noting out that can be used for later improvement:

--- 1 ---
Navigating through the menu is a bit clumsy. Displaying a compact list should not cluster the entire navigation window. In order to navigate (for example assign a boat), I have to go into a list (compact or detailed) and then go into another list for more commands which leads to an even deeper subsystem. My recommendation for you is to achieve as many as functions or actions with as less as many required menus. Try to avoid menu's within menu's.

--- 2 ---
Every input command is starting on a new row. Try to set input on the same row as the instructions for that input. For example: “Please enter a name: “, where I can give input after the semicolon. It seems like a small thing. But your program is filling up with more rows of text with each action. Editing a member takes up to 10 rows.

--- 3 ---
Your program is quite slow in responding when showing the lists of members. Takes up to 2 seconds. On slower computers this could take up too about 10 seconds or more. Try to optimize list reading. This only seems to happen the first time it load a list.
This is most likely due importing the member data from a text file. In that case, try to import the data on start-up of the program, or as soon as possible, so that when the user is giving a input command it will work directly.

--- 4 ---
Some grammar mistakes throughout the program. While there are English grammar mistakes, they are not really a issue. The issues is the wrong and forgotten use of spaces, dots, comma's, punctuation, semicolons and so on. In short, sentence structure. The issues with it is that it makes it harder for me to keep track of reading the output and input. This problem also relates to issue 2).
Also what is missing if giving instructions on the screen for some input commands.

--- 5 ---
Documentations is missing, as well diagrams and source code. So I am not able to properly inform myself on how to use your program and can not comment on that. For example, I am unable to find where member data is stored (which is most likely in one of your .xml files).
Of course, it is your decision if other users are allowed to check member data from outside the program.

--- 6 ---
Your program is behaving weirdly if users temper with files. For example, if I remove all your .xml files, your program still loads and shows a list of all members. Yet, when interacting with your list, your members do not exist anymore. You should prevent the program in that case from working or recreate the contents of all .xml files. Try to add a check if all required files exist, otherwise send out a exception with giving re-installing instructions and abort the program.

--- 7 ---
Regarding “wrong input”, for example, using two strings for a person number and name. If last names are allowed, then it should be explained. “wrong input” should not be allowed and should return an error (or exception) and the member should not be created at all. In fact, I have not seen an exception at all from the program, or at least an error from “wrong input”. Of course, I understand not everything can be checked with a short time of develop. But at-least some input commands should have been prevented, like a negative value for boat length.

--- 8 ---
A huge issue. Your program uses hidden files. Most likely the database for members. If I would re-install your program, the database still remains on my computer even after replacing all of your files included the installation. This should not be acceptable at all. At least give the user the possibility to complete reset / delete the database (or program), or place all files from your program within the installation folder. Honestly, I have no idea where your database is located on my computer.

On to the good points worth mentioning about your program:

--- 1 ---
The given folder containing your program looks nice and compact, including with an easy to use executable file. Your program requires no configuration in order to get it working.

--- 2 ---
Your program contains all of the required objectives for grade 3.

--- 3 ---
No sudden crashes. Even with not-allowed input, your program still works.

--- 4 ---
I may seem hard with all my critics, but in fact your program works completely apart from the issues. These issues seem easy to fix from my perspective with a little extra time.

Finally some small suggestions:

--- 1 ---
Try to reset or cleanup the window.

--- 2 ---
A small suggestion, but nevertheless important to prevent confussion. Two different names are given for the Member ID attribute: “MemberID” and “Member number”. Same should be done for each attribute that currently does not have the same name. A program benefits from constant terms.

--- 3 ---
Try to make a simple ID system for members, long integer values are hard to interact with.

--- 4 ---
Try to split the name attribute into a first name and last name.

--- 5 ---
If there a difference between the member ID and the number assigned to a member? I do understand member ID (read suggestion 3), but I do not understand the “number”. It is not explained what it's function is. Seems a bit double for me.

--- 6 ---
Try to make sure that when the user should select a member, it can see all members at once within the same point of view. Even when I enlarge the console window, it is still to far behind to read.

--- 7 ---
Perhaps use centimeters instead of meters for boat length. Your demo includes a member with a boat that has a length of 2424,5 meters (which is a canoe).

In short:

Your program all works and have all required things needed for grade 3. However, I would recommend you to clean-up your programs presentation, along with some required encapsulation and error control. I understand that implementing an error check on person numbers takes way to long, but negative values for length should never be allowed and it quite easy to implement fast.

I do not know how well your source code works, and can not comment on that as I have not seen it. But given the impression with your presentation of input and output of the program, it seems that your program lacks some proper organization. Maintaining clear and nice looking code makes it easier to maintain a better presentation.

The presentation is oblivious the most important part of your program, and should be easy to use, navigate and keep track of what happens during execution. I do get the impression that your program does not make use of constant and global source code instructions because of the presentation as things seem to be presented randomly without structure, for example different names for one presented attribute.

I think your program is a few inches away from grade 3. But if you improved upon a few of my suggestions and issues, I am almost sure you would pass grade 3 with ease.

There is only one issues that stands above all other issues, issue number 8, and should be fixed, explained, documented or prevented in one way or another as soon as possible. Users would like to keep track of all files that are placed on their computers. You should never install or even place text documents from outside the installation folder without letting the user know of it. Your program does not even require a installation, so it was already assumed your installation folder had all the required files.

Summary:

Your program has two major issues: Presentation and Error Control. Other than those major issues, your program seems good enough for grade 3. Issue number 8 should be fixed as soon as possible as it should not be acceptable.

On a side note not regarding the review, I would like to know from you where the hidden database would be installed on a computer.