Mandatory assignment ESW

Exercise 1 Struct/linked List Exercises

Design (<u>on paper</u>) the structure of your linked list. Describe how an element is added and removed. Describe for all cases: empty list, half-full list etc. – This is to understand the problem!!

In this exercise you must design a general *linked list* as an Abstract Data Type (ADT) (like a Class in Java).

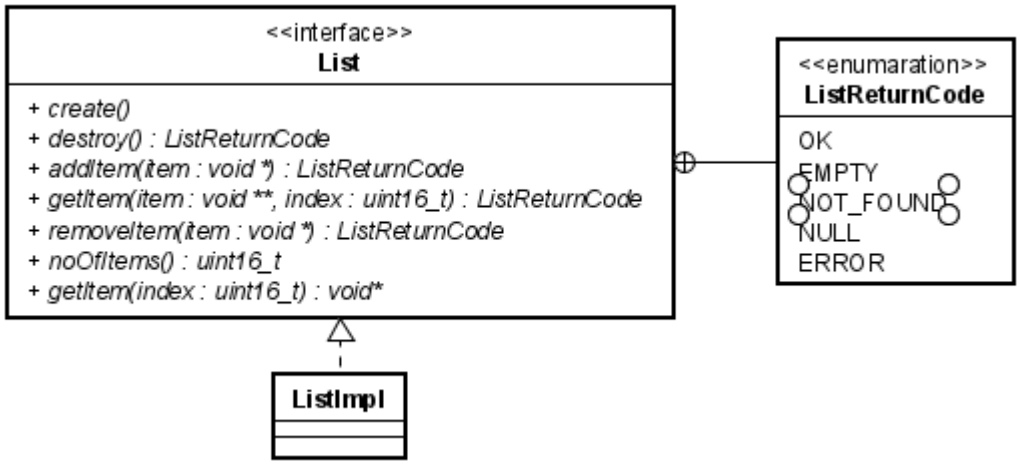

Figure 1 Preliminary Design for Exercise 1

Implement a linked list in two files list.h and list.c

The items to be stored in the list is given as a void pointer (`void *`) that points to the element. In this way, you can implement a generic linked list that will be able to hold any kind of elements.

The linked list must at least have the contents shown in the class diagram above.

The documentation for the wanted linked list can be found here:

https://github.com/ihavn/ESW1-LinkedList

Exercise 2 Use of your linked list

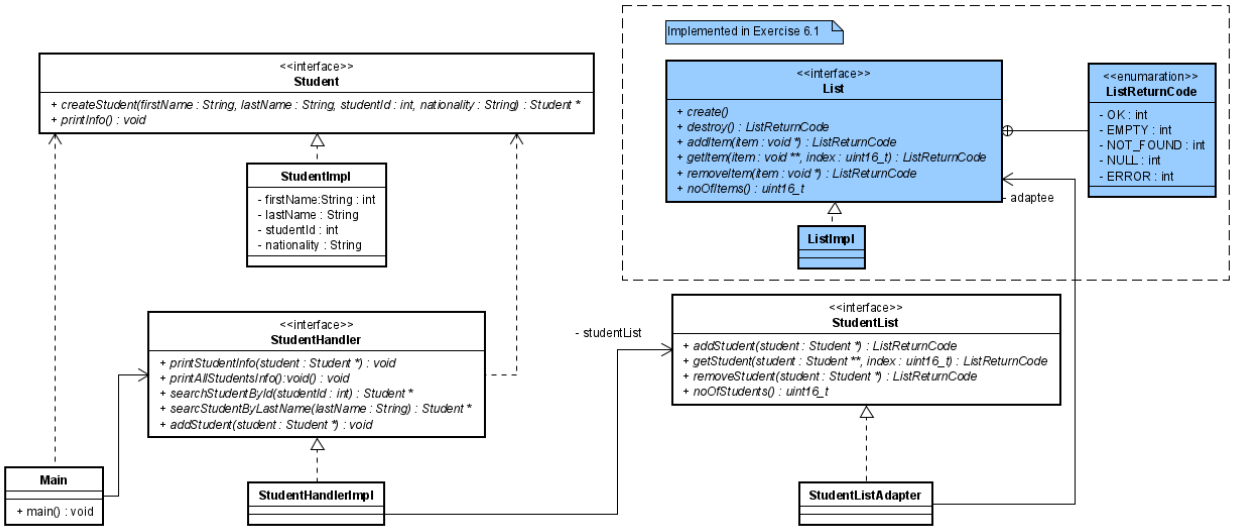

Figure 2 Preliminary Design for Exercise 2

To try out your linked list you must do the following:

1. Implement a structure type in a program that holds information about a student. Try to use typedefs to make your code more easily readable. The information about a student is:
   - First Name
   - Last Name
   - VIA Student ID
   - Nationality
2. Use your ADT linked list to hold the students (these will be the items in your linked list) As it can be seen in Figure 2, I suggest the use of the *Adapter Design Pattern* to make a *StudentListAdapter* for the *List* (Adaptee). The adapter will then take care of all the casting to and from *student* to *void**.

   The following is included in Figure 2 Preliminary Design for Exercise 2:

3. Make a function that can print the information about one student. The function must take a pointer to a student struct as argument
4. Build a main program that creates three students add them to the linked list
5. Get the students one by one from the linked list and print them out
6. [Optional] Implement a function that searches a linked list for a given VIA Student ID.
7. [Optional] Implement a function that searches a linked list for a given student by last name.

Exercise 3 From Design to Code

In this exercise you should implement an OOP solution for a project that is not OOP.

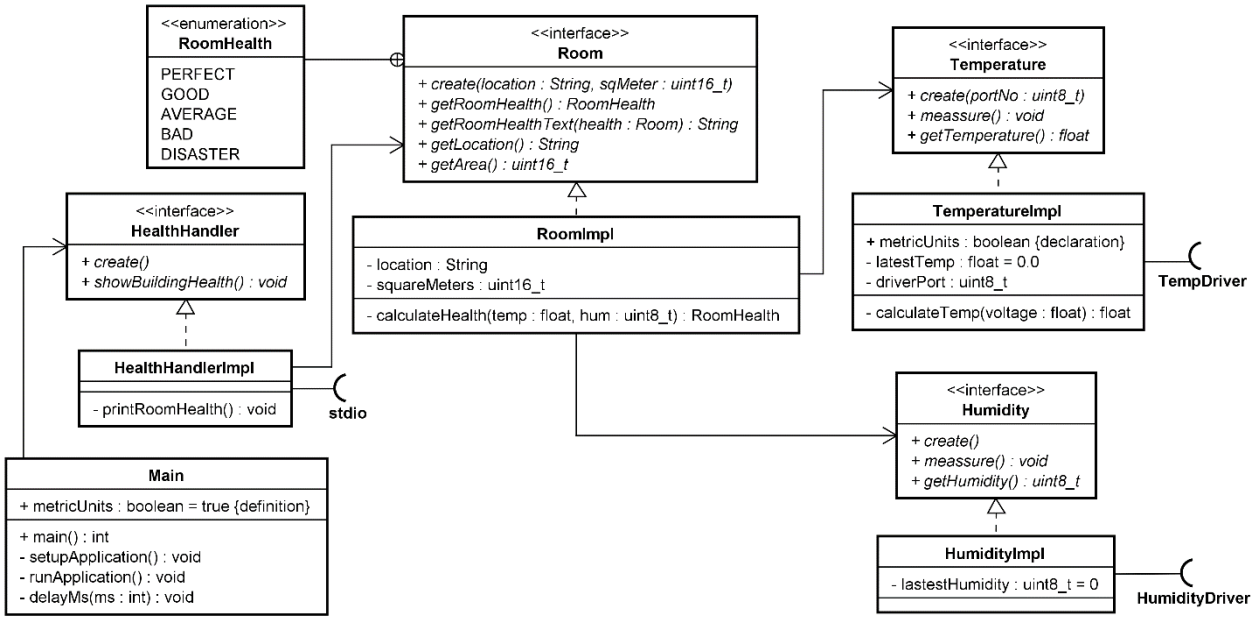

Figure 3 The original Projects design

The original Projects code can be found in *Example-Without-ADT.7z*.

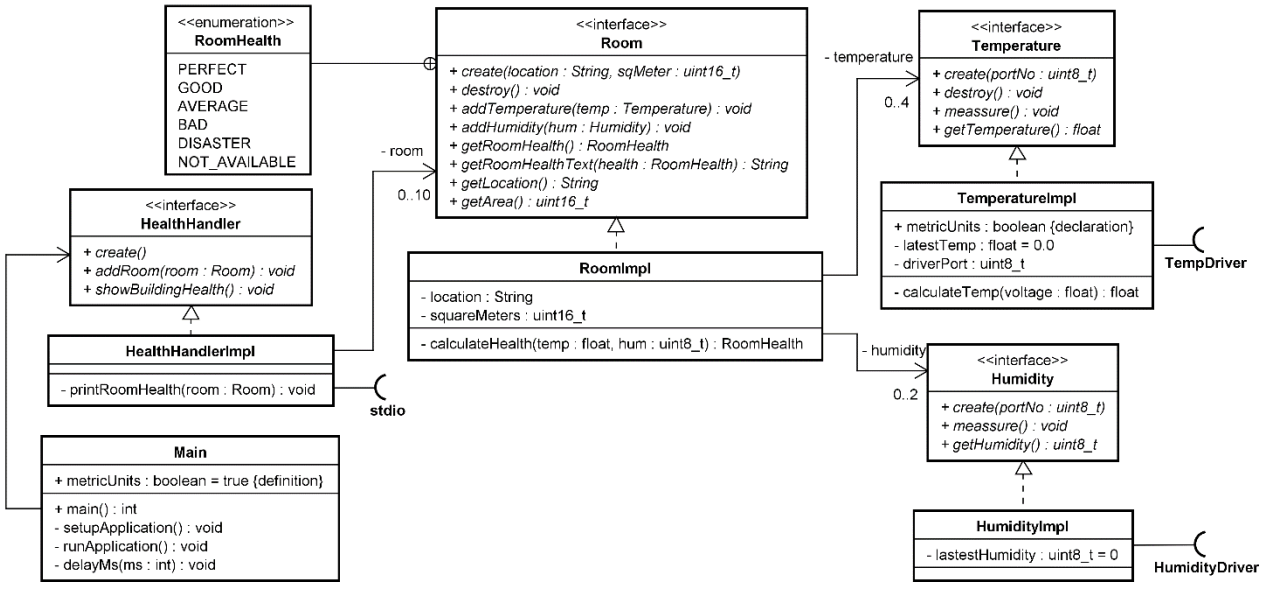

Figure 4 The OOP Projects design

The rest of the design (Sequence Diagrams and Class Diagram) can be found in *Design With Classes.7z*.

A complete description of the exercise and explanation of the design can be seen in the video: *ESW1 From Design to C Part III.mp4.*