1. The polymorphism techniques used to implement Sorter and CountingSorter, discussed in the class, can be extended to allow Sorters to support arrays of different types. Modify the OOSort2.cpp program (the source codes can be obtained from my web page), so that it supports arrays of integer, double, and string types. Note: (a) you should modify main program to demonstrate your results, (b) you should make your program as simple as possible, and (c) you are encouraged to use operator overload to implement your program, if you are familiar with it (however, using operator overload is not required; you may define member functions to accomplish the same results, if you are not familiar with operator overload).
   (a) Use the following class diagram to implement your program (called OOSort4A.cpp). Draw the class diagram of your implementation, showing all member variables and functions. [4 points]

   ![Class Diagram A](image1.png)

   (b) Use the following class diagram to implement your program (called OOSort4B.cpp). Draw the class diagram of your implementation, showing all member variables and functions. [4 points]

   ![Class Diagram B](image2.png)

2. This problem is an enhancement to your homework#1.
   (a) Rewrite your program written for homework#1 so that it is organized as a set of classes (you do not have to do anything if you have already done so). [4 points]
   (b) Draw a class diagram for your new program. Note: both the member variables and functions of your classes should be shown clearly in your diagram. [3 points]