

Com S 362: Object-Oriented Analysis and Design

Fall 2006

Design and Implementation Homework 3

Due Date: Wednesday Oct 18, 2006, 11:59 PM

The objective of this homework assignment is to develop an understanding of interfaces, inheritance and polymorphism. You MAY use the WWW and other books for this homework. You may NOT consult with other students about this homework, post solution to this homework, or make the solution available to others in any format tangible or intangible. Failure to comply with these regulations will result in a grade of 'F' for this course without any question. The matter will also be pursued to the fullest extent possible under the university's regulation.

Problem Description: design and implement a sorting application for positive integers. This application must:

- Ask the user for an input list of numbers;
- Ask the user for the choice of storage technique for the numbers. Two storage techniques should be supported at the moment Array that inherits from java.util.ArrayList and Linked List that inherits from java.util.LinkedList,
- Ask the user for the choice of sorting algorithm. Two sorting algorithms merge sort and quick sort, should be supported now.
- Sort the numbers, and
- Output the sorted numbers.

There are, however, some constraints on the design and the implementation of this application.

- Implementation of each storage technique, i.e. linked list, array, etc should be name independent of each other, from the main program, and from the sorting algorithms.
- We should be able to add a new storage type (e.g. doubly linked list) and not have to change any other component in the program.
- Implementation of each sorting algorithm, i.e. merge sort, quick sort, etc should be name independent of each other, from the main program, and from the storage techniques (e.g. linked list, array).
- We should be able to add a new sorting algorithm (e.g. radix sort) and not have to change any components in the program and also be able to sort any existing storage type (e.g. array and linked list) as well as any new storage type (e.g. doubly linked list).
- Storage types (e.g. Array, Linked List) etc should be implemented as separate modules. Therefore, when adding a new storage type (e.g. doubly linked list), a new module for that storage type will be added without modifying any of the existing storage types.
- Similarly, sorting algorithms (e.g. merge sort) etc should be implemented as separate modules. Therefore, when adding a new sorting technique (e.g. radix sort), a new module for that algorithm will be added without modifying any existing modules for sorting algorithms.