

The point of this assignment is to practice the first steps in producing an object-oriented design.

Consider the operation of an order-processing center for a mail-order catalog company. The center will receive orders from customers, specifying one or more items to be purchased. It is possible that some ordered items may be unavailable, either temporarily or permanently. The center must provide a proper response to each order.

Ultimately, a software system must be implemented to handle these tasks. However, that is not your assignment. You are to identify a collection of "candidate" objects. That is, analyze the scenario described above and determine a collection of objects that you believe would be adequate for modeling the situation. You should strive to identify and describe a set of objects that is adequate to the task, without redundancy or logically unnecessary elements. Each object should be coherent and reasonably simple; don't overlook the opportunity to practice composition.

Each object should be given a descriptive name, and also a short but precise description of the data and operations it will encapsulate. This should not be expressed in C++ language syntax. One acceptable format would be a tabular arrangement such as:

Object Name: FileObject	
Data Content:	Name of file File contents Name of file owner File status (hidden, archive, read-only, . . .)
Operations:	Change file name Change file owner name Change file status Report file name Report file owner name Report file status Write file to a DiskObject Read file from a DiskObject

Note that the goal here is to identify and describe objects, not to design an implementation. The data types of members, parameter lists, return types, etc., are not of interest. On the other hand, relationships among objects should be indicated. For instance, in the example above, a FileObject may interact with (but does not contain) a DiskObject. It is certainly possible that some of the data members of a FileObject could be objects in their own right; if so, that should be indicated also. Good use of object names will make that trivial.

Your solution must be typed either as a plain text file (e.g., Notepad) or in another format that can be opened, viewed, and edited in MS Word. Your file should begin with a paragraph of the form:

Name: Joe Bob Hokie
ID Number: 123-45-6789
Section Index: 1234

The name you give the file does not matter, since it will be renamed when you submit it.

Submission Instructions: You will submit your design electronically, using the EAGS. Read the *Student Guide to the EAGS* for instructions. The *Student Guide* is available at:

<http://ei.cs.vt.edu/~eags/>

You must submit your design by the due date/time given below. The EAGS Server will be prepared to accept your submissions by September 8.