

The Register Container Itself

Purpose:

This exercise is intended to give you experience with storing instances in heterogeneous containers. This should give you experience with some of the power of the OO paradigm.

Your test program will give you an opportunity to see that, in fact, virtual behavior means the right function is called even when the client routine can't know what type it is using.

What to hand in:

Hand in a listing of the program (which must be understandable and formatted in a reasonable style), and a sample run of the program. Please ensure that your test program *thoroughly* tests the required classes.

The program will include a Register header file, one or more Register implementation files, and a test program that will store different types of entries in the register and ensure the functions work on them, demonstrating appropriate virtual behavior

You should use the provided object model to design and build the Register class.

Please capture your homework in a single file for submission, in an easily reviewable order.

Problems:

Register

Create the Register class. One of its components should be a container that can hold some number of register entries of any of the 7 requested types. Your container should be able to have elements inserted and "looked up" for reconciliation. This kind of wrapper is a standard use of *delegation*.

Build a test program that can insert and examine values in the container. Verify that dynamic polymorphic behavior is preserved once entries have been placed in the container.

Consider carefully what the common interfaces and structure are, and how your class can properly represent that. Ensure that all the functions required by your user interface in the final project can be implemented through your Register class.

The container that Register uses (delegates to) should be an STL container of an appropriate type.

User Interface

There is nothing to hand in for this thought exercise.

Think about how you will implement your user interface to the register after this week. What is the right way to do this? How can it be accomplished in a short period of time? How is your "main" related to an exercise with a user interface? We will also consider how it relates to an exercise providing a *service*.

Notes:

Don't forget proper constructors, destructors, copy constructors, assignment operators, and other support functions. Note that the container will also be very similar to that of the statement class, should you choose to implement it.

Evaluation criteria:

Container interface (.h file)	35%
Container implementation (.cpp file)	35%
Test program (.cpp file) and output	30%