Week 9 Exercises : Encapsulation

April 2nd 2012

1 Approach

As we move into the second term we want you to look at larger programs that you build up in steps. The first task is to be completed by everyone on the course. The second task is optional for students who want to progress further. Each of the steps inside a task corrosponds to the amount of work in an exercise last term.

Task 1 : Implement a Vector class

Arrays are fixed length containers. Vectors resize as necessary to hold their contents. This difference arises because arrays are provided as part of the language, but in many cases programmers find vectors more useful.

You will write a Vector class that stores arrays of integers internally, but provides an interface to the programmer to access the resizable container. You will write a simple test harness, in the form of a main function that creates Vector objects and performs some simple actions to check that they work.

The interface to your class should allow the programmer to find the current size of the vector, read values from indices within the vector, and write value to indices within the vector. When an interface call tries to write a value to an index that is too large the class should reallocate the storage transparently and continue.

Task 2* : Implement a Hashtable

The description in the lecture showed how a linked list works. A hash function takes a piece of data from a large domain (e.g. a string) and converts to a number in a smaller range. Because the range of the output is smaller than the input there will be collisions (two pieces of input data that hash to the same value). A hashtable is a method of storing data that uses a fixed number of storage slots, and a hash function to decide which slot (sometimes called a bucket) the data belongs. As two pieces of data that collide cannot be stored in the same slot a linked list is used to store all of the data in a single slot.

1. Design a hash-function for strings that outputs a number between 0-15.
2. Implement a class that holds a linked-list of strings.
3. Implement a class that holds a hash-table of strings.