CS 246 Fall 2018 — Tutorial 8

November 7, 2018

Summary

1 Virtual destructors

When a class is expected to be inherited from, it should mark its destructor virtual. Otherwise, the correct destructor may not run if delete is called on a pointer to the superclass.

Recall that an abstract class needs at least one pure virtual method. Since abstract classes are expected to be inherited from, the destructor might as well be pure virtual.

2 Vectors

The C++ standard library has a class `vector` which stores homogeneous lists and resizes dynamically. Vectors have iterators and hence can be used in range-based loops.

3 Observer Pattern

The observer pattern allows us to achieve "message passing" between classes: a class `A` can notify anything that happens to be observing it when a particular event occurs.

Exercise: Write a class `Piazza` to which students can register. `Piazza` should have a `makePost` method that accepts a string (the text of the post). Make classes `StudiousStudent` and `LazyStudent`. A studious student echos the content of every post they see, while a lazy student does nothing with it.

4 Decorator Pattern

The decorator pattern is used to wrap additional functionality onto a component. It involves an abstract base class which typically splits into two kinds of children: those that implement basic behaviour, and those that own an instance of the abstract base class, and override the behaviour of some of the functions it provides.