COMP220 - Lab 5 & Homework 5

Fall 2015

Abstract

This lab requires zero programming and will be done in psuedo-randomly assigned groups. Your goal is to analyze problems, determine the merits of different ADTs in the contexts of those problems, and decide on the best choice of ADT. One person in your group should record your notes in a text file that is submitted as assignment *lab5* at the end of the lab. **Be certain to put everyone's name on the document!**

1 The Problems

You'll be working with problems 6, 12, 13, 14, 19, 22, and 25 from Chapter 5 of the text. Take a moment to familiarize yourself with them. Do not feel as though you have to proceed through them in any kind of ordered fashion. In fact, try working them in least to most obvious order.

2 The Analysis

Begin by looking at the problem and envisioning one or more concrete test case. Go ahead and document the test cases in a unit-test like fashion. See if you can identify different cases within the problem. As you think about how the problems are solved pay careful attention to the following details:

- What type of data are you collecting? What goes in the ADT?
- What types of *abstract* operations do you need to perform most? Add? Remove? Access?
- Is there a pattern to the common operations? If so, does it imply an ADT?

The exercise is about working from problem to program. *Don't fit the problem to any one ADT. Fit the ADT to the problem.*

Once you think you've identified the core ADT(s) needed to solve the problem, take a moment and identify the *critical operation*: the ADT operation called most in the process of solving the problem. If we can determine how often it is called, we can often get a pretty good feel for the *run-time efficiency* of the solution (algorithm) with respect to the operation's performance.

3 Homework 5

Due Monday 9/28. **Submit as hwk5 with handin.**

For homework 5 take a crack at a core procedure for one or more of the problems. You do not need to implement the problem as stated, but should instead capture the core functionality of the problem as a testable procedure. We're focusing on the central issue of data structures and algorithms right now. Problems are the genesis of algorithms, so I'll be looking as much as what you identify as "core" to the problem as the code itself.You're free to do the complete problem if you wish. In the even that your code involves I/O, be sure to adjust I/O from console based to the generalized, testable stream design we use.