**Name: Kenyon Geetings**

# CMSC 115 Reading Guide

Please enter your responses in red.

## Chapter 3

## Section 3.4 Strategies for program design Read

### Section 3.4.1 Re-read

1. What is their first advice for problem solving? Their first advice for problem solving is that different strategies work for different people.
2. What is the “danger of multitasking” that they talk about? Why would they think that this is particularly relevant for a book about programming? The danger of multitasking is that if often takes 64 seconds for you to regain your train of thought. This means that if you check your email every couple of minutes, you waste a significant amount of time just trying to get back into what you were doing. This is relevant for programming as it can be quite easy to get distracted when programming which will cost a lot of time for you to remember what you were working on.
3. Why do they suggest that waiting to the last minute is particularly foolish for a programming problem? They suggest that waiting to the last minute is foolish for a programming problem as you lose the valuable options to engage and fully focus on the problem.

### Section 3.4.2 Read

1. Why do they suggest that understanding what is being asked is more important in programming than in some other areas of life? The suggest that understanding what is asked if more important than other areas of life because you can not start to solve the problem until you understand it. Once you understand it, you could also relate other similar problems to help you.
2. What strategies have you used to “make the problem real” in your own work so far? Some strategies that I have used to make the problem real in my work is by testing my work with values that I know, either through a calculator, or by using a conversion chart online.

### Section 3.4.3 Re-read

1. What is dung-ball programming? Dung-ball programming is when you do not plan out your program before you start, and then you end up adding pieces on later without truly knowing where your program is heading.

### Section 3.4.4 Read

### Section 3.4.5 Re-read

Notice that this is an approach I have modeled in class. For example, when finding the longest Collatz sequence, we first started by finding the Collatz sequence for one starting number, then for the first 10,000 starting numbers, and finally found the longest sequence from the first 10,000 starting numbers. We built the program from the inside out, which is what they’re talking about here.

1. What is “skeleton code”? Skeleton code is basically a step up from pseudocode, where it allows you to test your program before actually writing the entire code for the problem. This allows you to focus on small parts of the program before putting it all together.

### Section 3.4.6 Read

### Section 3.4.7 Re-read

## Section 3.5 A Simple Example Skim

1. What strategy do they use to first build, and then “flesh out” their code skeleton? First, they start with building the skeleton program by using comments broken into parts, then they solve each part starting with the output, then the input. Then they test the input routine and do the calculations in order to flesh out their program.