Michael Camara

Honor code pledge: This work is mine unless otherwise cited

CMPSC 220

Due date: 11/11/15

Lab 7 Answers

Part 1:

Treating functions as first-class objects, specifically by allowing functions to be passed as parameters, seems to encourage reusability and generality of code in a program. The avg() function in fpar.c provides a perfect example for this. If one wanted to make functions to average two squared numbers and to average two cubed numbers, then two separate functions would normally be needed to handle each case. However, given the similarity of each of these cases, the avg() function uses function parameter passing to condense these cases into one. Further, the avg() facilitates countless other potential applications, such as averaging numbers raised to the power of 4, 5, 6, etc. Clearly this gives the programmer more options when designing the structure of code and the number of classes and functions needed.

Part 3:

After studying how lambda expressions work, I've realized that they could be very helpful when designing GUIs in Java. I've made a number of Java GUIs in the past to support user interaction with a program, and I'll often create various buttons, sliders, text fields, and other components in the process. However, I often end up creating unique event listeners for each of these components, many of which might have a very simple purposes, like incrementing a counter of retrieving a string from a text field. Lambda expressions could be used in place of these verbose listener classes to simplify and condense the amount of code needed. Although this might prohibit their reusability in some cases, they might still be worthwhile if one needs to support many unique but simple event listeners.