

# Project 1, CISC181 Fall 06

## Assignment

Write a program to compare the efficiency of insertion and selection sort.

Create an array of 10,000 integers and populate it with random numbers from 0-9999. Use the timing code from the class website to compare four sorting algorithms: iterative selection, iterative insertion, recursive selection, and recursive insertion.

The four algorithms will each be timed two ways: compiled the usual way, and compiled with optimization flags turned on. To find the optimization flags on Strauss, use “man CC” or “man g++” and look for how to optimize “tail recursion”. You will use a compiler flag to optimize, the same way we use the “-o” flag to name the output file.

A file cannot be run optimized and unoptimized at the same time (there will be two different executables created, one for each compilation), so the the code will be compiled and run unoptimized, then compiled and run optimized, to generate a total of eight data sets.

For each of the experiments to be timed, it is important to 1) time multiple trials (at least five per sort), and 2) use the same arrays for each algorithm. You can achieve the latter by either running each sort on one data array, and storing the results before making a new array, or by using srand() to ensure that the same sequence of arrays is used for each experiment. To help ensure this protocol, have the first and last ten elements of the data array be printed before each sort, so that you (and your TA) can see the data before it is processed.

Be sure that the timing of your sort is kept separate from the time it takes to initialize the arrays and print results - I/O calls can be time consuming.

All timing experiments must be scripted on Strauss.

When you have your multiple data points for each of the eight tests, graph them on a bar graph, showing all data points in one color and the mean in another.

## Grading

percent	
50	Code: modularity and decomposition, clarity of design and intent, productive comments, functional and efficient.
25	Data: generation and use protocol, following instructions
25	Results: reasonableness, presentation, following instructions