Exercise – Bubble Sort

Introduction:

In this exercise we will write the *bubble sort* algorithm and profile its performance.

In the next several exercises we will be adding various sorting algorithms to this application and comparing their performance.

Application Setup:

As with the tutorial for the session on *search*, we’ll be using the same *chrono* library to calculate the execution time of the sort algorithm.

Create a new console application. We’ll use the same application for the next several tutorials, adding each sorting algorithm as a new function.

The main() function for our profiling program is as follows:

#include <iostream>

#include <chrono>

using namespace std;

using namespace std::chrono;

void bubbleSort(int\* const array, int size);

int main()

{

const int size = 100;

// create a large array

int\* values = new int[size];

srand(time(nullptr));

// and fill it with some ordered data

for (int i = 0; i < size; i++) {

values[i] = rand() % size;

}

high\_resolution\_clock::time\_point t1, t2;

t1 = high\_resolution\_clock::now();

bubbleSort(values, size);

t2 = high\_resolution\_clock::now();

(continued on the next page)

for (int i = 0; i < size; i++) {

std::cout << values[i] << ", ";

}

std::cout << std::endl << std::endl;

std::cout << "bubble sort took " << (t2-t1).count() <<

" nanoseconds" << std::endl;

delete[] values;

std::cin.ignore(std::cin.rdbuf()->in\_avail());

std::cin.get();

return 0;

}

The first thing we do in this program is create an array filled with random data. Although it’s not a concern to us in this program, when we add more sorting algorithms we’ll want to use the same data set for all algorithms to ensure our profiling is accurate.

With our data set created we then get the current system time, call the bubble sort function, and once again get the system time when the sorting algorithm has finished.

The last part of this program will output the values in our (now ordered) array, before displaying the execution time of the bubble sort function. We output all the values in the array so that we can visually verify that the bubble sort function has correctly sorted the data.

Exercise:

Complete the bubble sort function.

You may want to refer to the pseudo-code and notes included in the lecture slides for this session.