CSCI 2720
Project 0: C++ and Algorithm Analysis
Assigned: January 20, 2016
Project Due: January 29, 2016 by 9:00PM

This project is to help you gain familiarity with the basics of C++, submitting files on nike, and analyzing algorithms. There are two distinct parts that must be turned in. Part I is a programming portion where you will submit a .cpp file and Part II is a writeup. You may work in teams for both parts, only one submission is required per team. Detailed instructions for successfully completing Project 0 are below.

Part I – Programming in C++

1. Setup your nike account – See the "Setting Up Your Nike Account" PDF to ensure you have the correct versions of gcc, gdb, and g++ required for this course.

2. Your program must include a makefile, called makefile (do not add a file extension), that compiles your source files into an executable object called "p0" using a compile directive. The makefile should also have a run directive and clean directive. (Note: you may specify a default textfile for the run directive).

3. Input:
   Your program must read input from a file. The filename (.txt file) will be specified at runtime. This file will contain a list of integers, each on a separate line. You can assume that there will be valid input with a single integer on each line. See sample input files for examples.

4. Processing;
   Your program should read in the numbers from the input file and insert them (in the order they are given) into an array. Additionally, your program should not allow duplicates. Ex: If the current number you read is 5, you should only insert it into your array if 5 does not already exist in the array.

5. Output: Your program should output the following things:
   1. The size of your array - Printed to the terminal
   2. The reversed order of the array elements - In a file named "output.txt"

Command	Required Program Output
`./p0 input1.txt`	`Array size: 5`
`./p0 input2.txt`	`Array size: 8`
`./p0 input3.txt`	`Array size: 2`

6. Include a README.txt file specifiying both partners' names and the division of work. (Just your name if you are working individually). Include a short description of your program and instructions for

compilation and execution. You must also include the following academic honesty statement in your readme file:

Statement of Academic Honesty:

The code in this project represents my own work. I have neither received nor given inappropriate assistance. I have not copied or modified code from any source other than the course webpage or the course textbook. I recognize that any unauthorized assistance or plagiarism will be handled in accordance with the University of Georgia's Academic Honesty Policy and the policies of this course. I recognize that my work is based on a programming project created by the Department of Computer Science at the University of Georgia. Any publishing of source code for this project is strictly prohibited without written consent from the Department of Computer Science.

Part II – Algorithm Analysis

This portion of the project should be typed and submitted as a PDF file. The goal is to analyze the code that you have written.

1. You should provide pseudocode for the code you have written in Part I. Each separate operation should be written on separate lines, numbered, and proper indentation should be used to indicate nesting. You will not be graded on your "correctness" of following pseudocode convetions, but please ensure that your pseudocode is legible and written at a high-level (ie: Someone who does not know C++ could understand). For example, if I read a line from a file in line 8, I might write it as "get next number in file". For more detailed examples, see the textbook and lecture slides.

2. Analyze your pseudcode - Follow the convention we learned for Insertion Sort in-class to determine the cost of each instruction and the number of times that instruction occurs. You should provide a chart similar to the table below:

Pseudocode	Cost	# Times
`1. Pseudocode here`		
`2. More code here…`		
`3. And more..`		

3. Using your above analysis, calculate the best Big-O notation for the worst-case of your program. Though you do not need to exhaustively show your steps, you should show your formula for the total running time T(n) and describe the steps you take to come to your conclusion.

Testing

You must test that your software compiles and runs correctly on nike using /usr/bin/g++. You are responsible for testing your software. Thoroughly and rigorously test, test, and re-test your software before submitting it to ensure that it works correctly on any test cases that we may use for grading.

Submission Procedure

In your home directory on nike, make a subdirectory named "proj0"and put copies of your source files in it and your .pdf report, as well as the header files, makefile, and readme.

Once you've LOGGED ONTO nike and you are in your home directory, execute the command line "submit proj0 cs2720". After submitting, verify that a file beginning with "rec" was created in your submitted directory. This "rec" file is your receipt of submission. If a "rec" file was not created, then the submission was unsuccessful, and you'll need to try resubmitting. If resubmitting does not work, zip or tar your directory and email it to the instructor and TA before the deadline.

Late Policy

Projects submitted even a minute after the deadline are considered late. Late projects will receive 20 points off per day up to 48 hours. Projects submitted after the 48 hour late period will receive a grade of zero. Projects not submitted will also receive a grade of zero

Grading

Part I	40 points
Part II.	60 points
Total	100 points