University of Wollongong
School of Computer Science and Software Engineering

CSCI124/MCS9124 Applied Programming Autumn 2015

Lab 3 (1 mark)
Due at 5pm Thursday, 26th March.

Background:
In this lab you will gain some experience with sorting data.

Task:
For this lab assume you work at the NASDAQ (http://www.nasdaq.com) stock market. Each day your supervisor gives you a list of the top 100 or so traders in the IT sector. Such data is represented in a file named `stockdata.txt`. The stock data file consists of many lines, each representing trade information relating to a specific company. Consider the following line:

```
AAPL 155  747733
```

The first field represents the stock tag i.e. `AAPL` (Apple Inc). The second field represents the current value of each Apple share, in this case `$155.00` and the last field represents the number of Apple shares traded during the day.

The data in the file is not sorted. Your task is to write a program that sorts the data in `stockdata.txt` in various ways. Your program should produce two alternative reports. The reports are generated as text files, which can then be further processed.

We'll use a struct to store the input data, it should include a cstring to hold a 4 character tag, an integer cost, and a long integer volume.

And we'll store the data in an array of these records of length 100 (we don't know how many are needed but this will suffice – make it readily changeable).

Step 1:
Write a program in the file `lab3.cpp` that opens the file `stockdata.txt` and reads the data into the array. Our focus is the sort, so the filename can be hard-coded into your program. Input is terminated by the end-of-file, at which time you know how many entries were in the file.

Step 2:
Once all data is read in and stored, display a menu.

```
Main Menu:
c: sort data by Cost
v: sort data by trade Volume
q: Quit
Enter Choice:
```

If a user enters an invalid choice (upper and lower case should be treated the same), the main menu should be reprinted.

If the user enters the `c` option, your program should sort the data stored in the array using the **selection sort** algorithm provided to you in class. Once the data has been sorted, it should be written to the file `data.txt` (again hard-coded) in a format identical to the output below:

```
Corporate Stock Cost - NASDAQ
Stock Tag
Cost
GOOG 390
ADBE 192
INFY 167
AAPL 155
LOGI 130
ERIC 110
```

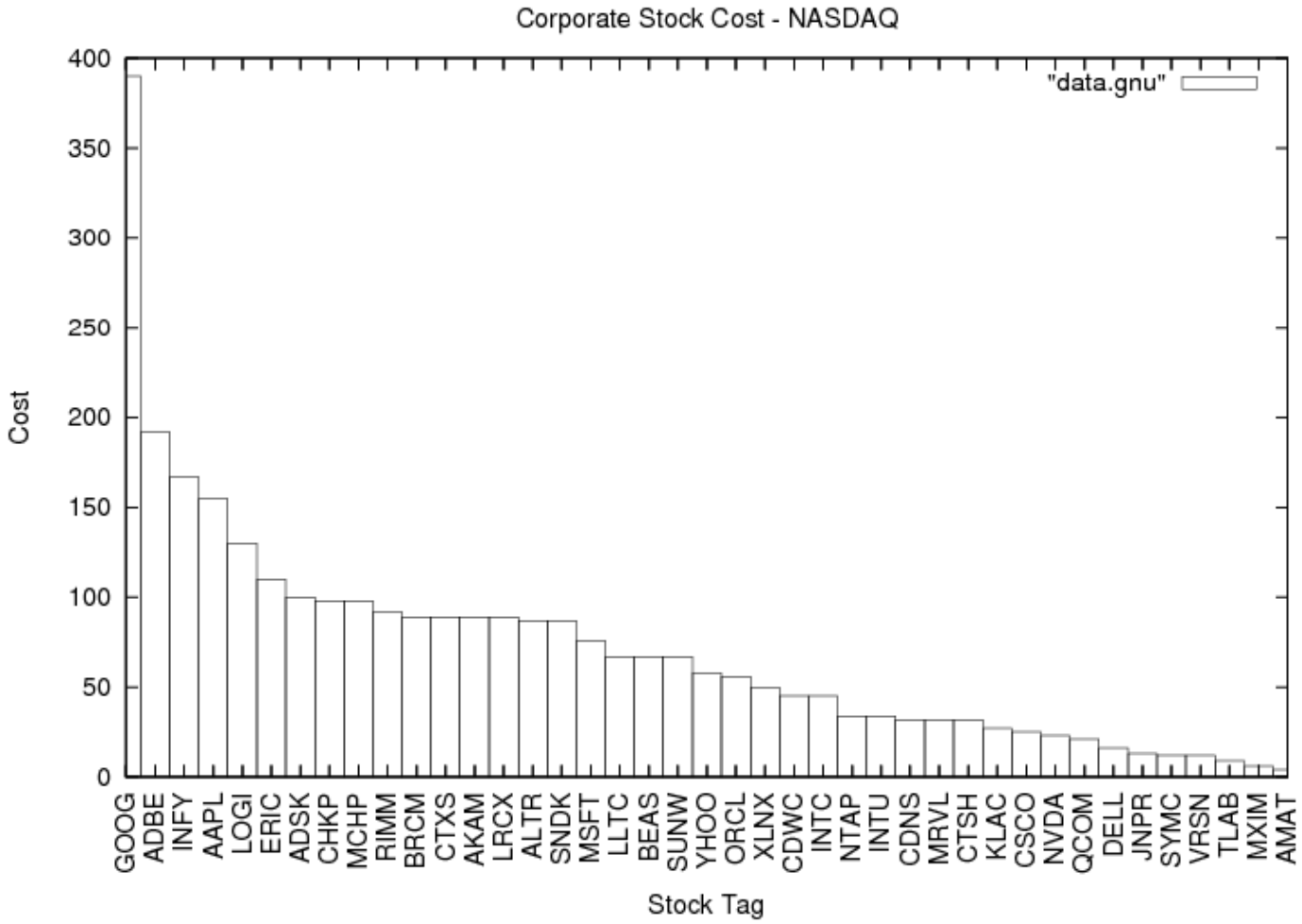
The first three lines represent the title of the report, the name of the first data item in the list and the name of the second. Each subsequent line represents a stock and its current cost in order of largest to smallest.
The program should then terminate.

If a user enters the `v` option, exactly the same process is repeated but this time instead of using selection sort you should use **insertion sort** and the output must look like the output below:

```
Corporate Stock Trade Volume - NASDAQ
Stock Tag
Volume
ERIC 928238
INFY 899921
ORCL 893487
BEAS 868694
AAPL 747733
CTSH 685684
NVDA 654743
```

In either case, options `c` and `v` produce the same named output file.

Step 3:
OK - now for the reason for the first three lines to the output file. The report you provide will be used by another program called `converter`. The converter program, converts your data into a format that can be plotted into a graph using `gnuplot`. Here's what you should get for the cost sort.



You will notice that each axis of the graph is labelled, and the graph has a heading. The first line in the output of your program represents the title of the graph, the second represents the x-axis title and the third line represents the y-axis title. Your program must write these three lines, and vary them depending on the report being produced.

Compile converter.cpp **as a separate program**, the executable should be named converter. The source for the converter can be found on moodle.

To produce these graphs do the following.
1. Run your program and produce a report in the file `data.txt`.
2. Run the `converter` program using the following command:

```
$ converter data.txt data.gnu | gnuplot
```

The converter takes `data.txt` and produces a file called `data.gnu` and some other input to gnuplot, which produces a file called `output.ps`. This file contains a graph in postscript format, which you can view in any Postscript viewer (such as `ghostscript` or `evince`).

Submit:

You are to submit `lab3.cpp` using

```
$ submit –u <username> –c CSCI124 –a lab3 lab3.cpp
```