CS 456 : Advanced Algorithms

Programming Assignment #02

Total Points: 150

Assigned Date	: Thursday, October 17, 2019
Due Date	: Thursday, October 31, 2019 @ 09:29:59 a.m.

Overview

For your second programmming assignment, you will implement three versions of the **Convex Hull Problem** – Graham Scan (see p. 1029 of the CLRS book), Jarvis March (see p. 1037 of the CLRS book), and QuickHull (see p.195 of the Levitin book) – and empirically validate their asymptotic runtime behavior using **computer generated results**. The convex hull algorithms run at different complexities with one of the algorithms even exhibiting a different complexity based on average case and one on worse case. Specifically, you are expected to think about and address the following questions:

a. At what size n_0 does your implementation start to exhibit asymptotic complexity?
b. What is the characteristic of your input required to generate *worst case* complexity of the quickhull algorithm?
c. How does the measured run time correspond to the theoretical asymptotic bounds?

Instructions

- This is an individual assignment. **Do your own work**.
- **Start early!!**
- **Take backups of your code often!!**.
- You may use any programming language of your choice. However, you **must** make sure that your code compiles and runs on a typical Linux machine.
- Absolutely **DO NOT** include executables with your submissions.
- The report part of your solution must be produced using a word processor. I highly recommend **Latex**.
- Any figures, graphs, plots, etc., should also be produced using appropriate computer applications.
- Graphs/plots should be properly labeled.
- Your final report should be in **PDF** format. No exceptions.
- Follow a good coding standard. Use the Google C++ coding standard found here `http://goo.gl/1rC1o`, if you don't already follow one.
- Make sure to use test cases of sufficient size so that problem size is the dominating factor in run times.
- Use enough input sets so that a clear pattern of run time is established.

Input Specification

- Your code must be capable of handling command line argument which will be a file path to a list of points in 2D space. A small test case is provided at the end to help you design your input files.
- You are required to generate a file named **worstcase.txt** that contains a set used to demonstrate worst case time complexity of the quickhull algorithm.
- All input files should follow a format that has each line containing only one point with the x-coordinate listed first followed by a space and the y-coordinate.

Measurements

- It is important that the same set is ran by all three algorithms. This is important because the runtime of the jarvis march is dependent on the location of the points as well as the number of points. It will be important to track the number of input points as well as the number of output points.

Output Specification

- For output, generate a file named **[numberofpoints].txt**. The top three lines should include the times of the three algorithms followed by the output set in the same format as the input set. e.g. 500 points found in file 500.txt.
- Be sure to test enough different size sets to accurately graph behavior.

Deliverables

The due date of this assignment is **Thursday, October 31, 2019 @ 09:29:59 a.m.** A dropbox will be opened for submission on Moodle before the due date. A complete solution comprises of:

- A report **in PDF format** that includes the followings:
  - Motivation and background of the experiment
  - Testing plan
  - Have a section for each algorithm
    - Psuedocode
    - Correctness Proof
    - Test Results
    - Observations and justifications about results. If things do not appear as expected, explain why the results are the way they are
    - Problems encountered
  - Conclusion and performance comparison. Be sure to discuss the worst case quickhull file you created.
- A compressed tarball of the directory containing your source codes. Your Makefile should be in the base directory created by decompressing the tarball.
- Absolutely **do not** include any executables with your submission.
- To create a compressed tarball of the directory source, use the following command: `tar -zcvf lastname.tar.gz source/`. Change lastname to your last name. Your tarball should decompress such that it will create a directory named your last name
- **worstcase.txt** your file used to demonstrate worst case behavior in quickhull
- Your executable should be named project2

Tentative Grading Rubric

- Styling: Compiles, easy to read, properly indented, etc.., [**10 points**]
- Handles Input Correctly [**10 points**]
- Output Correctness [**10 points**]
- Correct Submission Structure [**10 points**]
- For each algorithm: [**35x3 points**]
  - Implementation [**5 points**]
  - Test Plan [**5 points**]
  - Psuedocode [**5 points**]
  - Proof [**5 points**]
  - Result Presentation [**5 points**]
  - Observations [**5 points**]
  - Justifications [**5 points**]
- Conclusion [**5 points**]

Sample input file

small.txt

```
6 6
7 9
5 0
0 1
4 9
7 4
8 5
2 6
8 3
8 1
```

Sample output file

set10.txt

```
Graham: .0912 sec
Jarvis: .124 sec
Quick: .065 sec
5 0
8 1
8 5
7 9
4 9
2 6
0 1
```