Lab 9 Grading Guidelines last modified by Aaron Bloomfield on August 13, 2013 11:27:16 AM EDT

Lab 9 grading guidelines

Pre-lab

Out of 10 points. This lab part **is** expected to compile. The functions are expected to be recursive, or no points will be given.

Lab 9 grading guidelines

- 10 points: Collatz Conjecture
  - 3 points: Trial Runs
    - 1 point for each correct output.
  - 1 point: Timer
    - 1point for a timer with proper precision
  - 4 points: Code correctness
    - 4 points (even if it doesn't actually work): Code looks correct and should work
    - 3 points: There are a couple flaws here and there with the code such as syntax
    - 2 points: Code is there, but has some major pieces missing like correct calling convention.
    - 1 points: Something non-trivial was submitted
  - 2 points: Requirements
    - Function was optimized in some way or looks efficient (It must still follow proper calling conventions!)

They are supplying their own threexinput.cpp file, so there could be several differences in output (they could have even less information than what is in the sample output) and possibly in the way they are getting input. Below describes sample input and output for trial runs:

Pre-lab input

There were four test cases that were run. Each one was run 1,000,000 million times (this was the second input to the program). The numbers for the first input (the number to pass to the Collatz conjecture) are: 0, 1, 9, and 25. The output is, respectively, 0, 1, 9, and 23.

Pre-lab output

Their output format can vary. Note that many did average time, rather than total time - that's fine. And their times can vary, of course - but not the number of iterations

```
Iterations: 1000000
Steps per Iteration: 0
Total steps taken: 0
Time taken: 0.013000

Iterations: 1000000
Steps per Iteration: 1
Total steps taken: 1000000
Time taken: 0.025000

Iterations: 1000000
Steps per Iteration: 9
Total steps taken: 9000000
Time taken: 0.162000

Iterations: 1000000
Steps per Iteration: 23
Total steps taken: 23000000
Time taken: 0.244000
```

In-lab

Out of 10 points. This lab part is **not** expected to compile.

10 points: In-Lab Report

- 2 Points for overall quality of the report as in grammar, organization, etc.
- If no code was provided, 4 points were deducted--analyzing/comparing code was a large part of this assignment, and without it everything is just speculation/research. No points were taken off further down for this, though.
- Topic that was addressed (8 points): can be one of the following topics
    - Only one topic in List 2 had to be addressed for the in-lab (the other 2 must be addressed for the post-lab). We are mainly looking to see if they hit all of the main points in one of the topics. We are also not looking for a perfect report. That will be part of the Post-Lab. Feel free to interpolate between the point values as long as you are consistent.
    - Inheritance (8 points):
        - (2 points) They show where data members are laid out in an object that they created.
        - (2 points) They explain how construction and destruction occurs in the class hierarchy.
        - (2 points) They explain what happens when a user-defined object is instantiated and what happens when it goes out of scope.
        - (2 points) They show the "life cycle" of an object in assembly code and point out where destructors and constructors are getting called
    - Dynamic dispatch (8 points):
        - (4 points) They describe how dynamic dispatch is implemented.
        - (4 points) They show how dynamic dispatch is not the same thing as dynamic memory using a simple class hierarchy that includes virtual functions
    - Other architectures (8 points):
        - (6 points - 2 pts for overall explanations, 1 pt for each of the 4 differences found) They compare code generated by g++ for x86 to code generated by g++ for a different architecture and describe the differences found (at least 4 non-trivial differences must be described)
        - (2 points) Any similarities between code are identified
    - Optimized code (8 points):
        - (8 points - 2 pts for each of the 4 differences found and their explanations) They compare code generated normally to optimized code (They must provide the original sample code and optimized code) and at least four non-trivial differences are described.
    - Templates (8 points):
        - (2 points) They show what the code looks like for the instantiation of a simple templated class.
        - (4 points) They describe what happens when they instantiate the templated class for different data types, show the generated code, and describes the major differences.
        - (2 points) They compare code for a user-defined templated class or function to a templated class from STL.

Post-lab

Out of 10 points. This lab part is **not** expected to compile.

The lab report must cover all of the key points described in the lab assignment for the topics in List 2. The grading will be based on the quality and content of the report.

- C tutorial: 2 points
    - 1 point: properly prints out 2, 4, 6, 8, and 10 (in either order)
    - 1 point: properly allocates and/or deallocates the linked list (no credit for this point if the list is not deallocated properly)
- Report: 8 points
    - 1 point: At least 2 sources were cited
    - 2 points (1 point for each block of supporting information up to 2): There is strong evidence given supporting their report in the following ways:
        - Code snippets
        - Screenshots
        - Quotes from outside sources
    - 2 points: Analysis and Hypotheses
        - 2 points: Everything was well explained and educated hypotheses were made when there were things unknown.
        - 1 points: Not everything is explained nor even addressed in the analysis.
        - 0 point: Poor explanations, and little attempt was made at making hypotheses when they did not know what

was going on.
    - 3 points: Key points for their chosen topic were covered
        - 3 points: Everything seems to be covered
        - 2 point: There was something missing
        - 1 point: There were a lot of things missing

Key Points for Parameter Passing and Objects:

- Inheritance:
    - They show where data members are laid out in an object that they created.
    - They explain how construction and destruction occurs in the class hierarchy.
    - They explain what happens when a user-defined object is instantiated and what happens when it goes out of scope.
    - They show the "life cycle" of an object in assembly code and point out where destructors and constructors are getting called
- Dynamic dispatch:
    - They describe how dynamic dispatch is implemented.
    - They show how dynamic dispatch is not the same thing as dynamic memory using a simple class hierarchy that includes virtual functions
- Other architectures:
    - They compare code generated by g++ for x86 to code generated by g++ for a different architecture and describe the differences found (at least 4 non-trivial differences must be described)
    - Any similarities between code are identified
- Optimized code:
    - They compare code generated normally to optimized code (They must provide the original sample code and optimized code) and at least four non-trivial differences are described.
- Templates:
    - They show what the code looks like for the instantiation of a simple templated class.
    - They describe what happens when they instantiate the templated class for different data types, show the generated code, and describes the major differences.
    - They compare code for a user-defined templated class or function to a templated class from STL.

Untitled note

Find a notebook

Add tag

Add comments

Version 6.0.6: c128369/1.0.1.250

Web Clipper tutorial

Options

Saving clip...

To:

Share

Save

Selection

PDF