CS 5393 C++ FOR CS: LAB 2

C++ PROFILER

Overview: In this lab, you will design and implement a C++ Profiler system to measure and analyze the performance of code sections. You'll create a user-friendly interface, output profiling data, visualize results, and demonstrate the profiler's effectiveness in improving code performance.

Repo Link: https://classroom.github.com/a/UJGIrgIo

Due: October 23, 2024 @ 11:59pm.

Code Requirements (60%):

1. Implement a Profiler system that provides the following statistics for each profiled section:
    a. Section name (const char*)
    b. Call count (int)
    c. Total time (double)
    d. Minimum time (double)
    e. Maximum time (double)
    f. Average time (double)
    g. File name (const char*)
    h. Function name (const char*)
    i. Line number (int)
2. Implement functionality to output profiler statistics to a file. Support at least two of the following formats:
    a. CSV
    b. JSON
    c. XML
    d. Include a brief explanation of why you chose these formats.
3. Design a user-friendly interface for developers to easily integrate and use the profiler. Should include the following at a minimum.
    a. A method to start profiling a section
    b. A method to end profiling a section
    c. A method to calculate section statistics
4. Optimize the profiler design to minimize its impact on performance when measuring timing of specific sections. Consider concepts such as:
    a. Efficient data structures for storing profiling information
    b. Minimizing the use of system calls
    c. Thread-safety considerations

5. Support hierarchical measurements of sections, allowing profiling of nested code blocks:
    a. Enter A
        - Enter B
            1. Enter C
            2. Leave C
        - Leave B
    b. Leave A
6. Support interleaved profiling, where a previous section can be left before the last entered section completes:
    a. Enter A
        - Enter B
    b. Leave A
        - Leave B
7. Implement a visualization tool for the profiler outputs. This can be written in any language and use any 3rd party tools. Students are encouraged to use Generative AI tools to develop an advanced visualization solution. More points are given for the more advanced and interesting you make the visualization tool.
    a. Chose an appropriate visualization library or tool (e.g., matplotlib, D3.js, Excel)
    b. Create at least two types of visualizations (e.g., bar chart for average times, timeline for nested calls)
    c. Include options to filter and sort the data in the visualization
2. Create a demonstration of the profiler: a. Profile a non-trivial algorithm or data structure operation (e.g., sorting a large array, graph traversal) b. Make at least three different performance-related changes to the code c. Re-profile the modified code to show the performance impact of each change

Report Requirements (40%):

1. Provide a detailed writeup describing the design of the Profiler system:
    a. Explain the overall architecture and key classes/functions
    b. Discuss performance considerations and how they were addressed
    c. Provide a clear, example-driven guide to using the Profiler API
    d. Explain how hierarchical and interleaved profiling is handled, with diagrams if necessary
2. Document the visualization tool:
    a. Justify your choice of visualization technology
    b. Provide screenshots and explanations of each type of visualization
    c. Include a user guide for interacting with the visualizations
3. Present a comprehensive analysis of the demonstration:
    a. Describe the initial algorithm or operation being profiled
    b. Explain each performance modification in detail, including the rationale behind it
    c. Present before-and-after profiler results for each modification
    d. Analyze the impact of each change, explaining why it improved (or didn't improve) performance
4. Submit all code and reports in the repo

If your project fails the compile action upon final push to git repo, it will receive a 0. Be sure to resolve any issues with the compile action prior to deadline.