PA2 – Hot / Cold Data Structures

Due Date

- See Piazza for due date and time
  - Grading the next day
- Submit program to perforce in your student directory
  - Sub directory called:
    - /PA2/...
  - Fill out your ***PA2 Submission Report.pdf***
    - Place it in the same directory as your solution
    - Enter the final Changelist number of your submission
    - Enter the number of test passed for Problem3
    - Save the output text files
      - // problem 1
      - // problem 2
      - // problem 3 - debug & release
    - Write up a quick discussion in the report
      - What you learned from this assignment

Goals

- Learn
  - Data cache / Alignment
  - Hot / Cold data structures
- Understand firsthand how alignment and data cache affects performance

Assignments

1. ***Identifying data layout and alignment for supplied data structures and C++ classes***
   - Write a program to cleanly demonstrate the data alignment for any give data structure.
     - Output should be *printable to show padding*
       - Number of padding bytes
       - Output should also return the size of the total alignment
     - Clean output, nice formatting
     - Save the Text file in that directory
       - Called: Output.txt
   - Use test classes and structures supplied in perforce:
     - /reference/Assignments/PA2/problem1/TestDataStruct.h

2. Rework several data structures to reduce memory size

- Verify the before and after data size for each of the data structures supplied
  - Clean output, nice formatting
  - Should show padding
  - Number of padding bytes
  - Size of structure before/after
  - Save the Text file in that directory
    - Called: Output.txt
- Use test classes and structures supplied in perforce:
  - /reference/Assignments/PA2/problem2/TestDataStruct.h

3. Rework the supplied linked list data structure to a hot / cold data structure

- Refactoring the necessary insert/delete/find functions to the linked list
- Converting the existing data structure data to this new format
- Profile the before and after performance numbers of the linked list for the given input.
  - Save the Text file in that directory
    1. Called: output_Debug.txt
    2. Called: output_Release.txt
- Use test system supplied in perforce:
  - /reference/Assignments/PA2/problem3/…

General:

- Write all programs in cross-platform C++.
  - Optimize for execution speed and robustness.
- Create a programming file for each problem, for example
  - Student directory
    - /PA2/problem1/…
    - /PA2/problem2/…
    - /PA2/problem3/…
  - Make sure that each problem can be compiled and run through the checked in solution
- Do all your work by yourself
  - Feel free to talk with others about ideas on Piazza
  - You are 100% responsible for all code
  - See syllabus about collaboration rules
- Check in the problems multiple times, at least 3 times per problem
  - Have reasonable check-in comments
  - Seriously, I'm checking
- Make sure that your program compiles and runs
  - Warning level 4, some times that is not possible due to MS headers…
  - Your code should be squeaky clean.
- More details for Problem 3: bloated problem
  - You need to implement 3 functions:

1) Convert the function from bloated to Hot/Cold data structures
2) Find a data node in the new Hot/Cold data structure
3) Verify that every original node is correctly represented in the Hot/Cold structure.

- I included my timings in output_Keenan_release.txt and output_Keenan_debug.txt
- You can see my timings, for reference. Please leave the output file scheme alone, I would like everyone to have similar format.
- Your timing will vary depending on your machine, but the ratios should indicate how much you improved the performance.
- Interesting results:
  - My original timing to find the last data structure in the code is:
    - 23.522184 ms
  - Hot Cold data structure:
    - 1.456168 ms
  - 16.153482 times faster!!!
    - Cache does yield performance improvements.
- You might think this is not much, but most games are running at 30Hz, so you have 33.33ms to do your whole game per tick.
- If you are at 60 Hz, you have 16.66ms. Reducing timing from 23.5 ms to 1.45 ms is quite significant.

Validation

Simple check list to make sure that everything is checked in correctly

- Did you do all 3 problems?
- Do they compile and run without any errors?
- Warning level 4 free?
- Submitted it into /PA2 directory?
- Filled out the submission report?
- Can you delete you local drive, regrab the /PA2 directory?
    - Is all the code there?
    - Does it compile?
- Did you submit your text files?

Hints

Most assignments will have hints in a section like this.

- Do many little check-ins
    - Iteration is easy and it helps.
    - Perforce is good at it.
- Look at the lecture notes!
    - A lot of good ideas in there.
    - The code in the examples work.