Optimized C++

Fall 2014

CSC 461

instructor:	Ed Keenan
email:	ekeenan2@cdm.depaul.edu
office hours:	Tues 3-5pm, 9-10pm after class or by email appointment
office:	CDM 830
phone:	(312)362-6747
website:	piazza.com/depaul/fall2014/csc461 (Preferred communication)
lecture:	CDM 224, Tuesdays, 5:45-9:00pm
Desired to Learn (D2L):	d2l.depaul.edu (Grades, Viewing lectures, Announcements)
Version Control:	perforce: *140.192.39.61:1666*

Description:

This game programming class will focus on developing software to efficiently use the fixed CPU power and resources that are found in today's console and mobile devices. This course will use real-world game examples that demonstrate performance and optimization issues that software architects face in software development. These problems include: performance enhancements through extended matrix instruction set, dynamic memory usages, performance related to increasing run-time systems to very large scale, C++ language enhancements and extensions, algorithms, streaming and profiling.

Prerequisites:
- Data Structures in Java or C++ (CSC 403)
- Computer Systems II (CSC 407)
- Discrete Math (CSC 400)
- or instructor consent

Learning Goals:
- Students will be able to analyze software systems, identifying performance related issues in its design and implementation.
- Students will be able to identify and remedy execution performance issues related to data layout, processor caching, unintended compiler interactions, algorithmic considerations, data containers and supplied subsystems.
- Student will be able to model the whole system to optimally configure runtime data layout on hard drives (external media) for speed of execution and to minimize memory spikes.
- Students will be able to configure and analyzed results from a commercial analyzer and profiler to optimized an existing software system.

Grading:

10 % - C++ Proficiency programs
70 % - Programming Assignments
- PA1: Assessment   – 5%
- PA2: Caching / Data Alignment – 10%
- PA3: Memory – 20%
- PA4: C++ Efficiency – 10%
- PA5: Math Optimizations – 10%
- PA7: Load in Place - 10%
- PA8: Commercial Profiler - 5%

10% - Final Exam
10% - Final Project

NOTE: You must pass the final exam (60% or higher) to receive a passing grade in the class.
(It's not an easy exam)

Textbooks and printed resources

Required Texts:

- ***The C++ Programming Language:*** Stroustrup
  - ***4th Edition 2013 (new edition) or 3rd Edition (either are acceptable)***
  - Stroustrup Addison-Wesley Longman/Pearson, 2014. ISBN: 978-0321563842
- ***STL Tutorial and Reference Guide: C++ Programming with the Standard Template Library*** (2nd Edition) 2001 - Musser, Derge, Saini, ISBN-13: 978-0201379235
- ***Additional material provide in class***
  - Websites and handouts

Optional:
- **Effective C++ (3rd Edition),** 2005, Scott Meyers, ISBN-13: 978-0321334879

Additional Material

- Will be provided by the instructor
- Lectures, links, SDKs and other corresponding material

Software

- ***Microsoft Visual Studio 2012 (ultimate edition - Recommended)***
  - DePaul MSDNAA link Microsoft_Visual_Studio_Ultimate_2012
  - C++ and C# install (future classes)
  - Microsoft Visual Studio 2013 is not used in this class.
- Perforce - Visual Client (p4v)
  - www.perforce.com
- Download and configuration instructions will be provided in class
  - Server address: **140.192.39.61:1666**

Programming assignments:

The intent assignments:
- Assignment 1 is a warm-up.
- Assignment 2-8 is to teach and demonstrate a new optimization concept.

Program Assignment #1: **Programming Assessment**
- Do a programming test (It's easy)
- Learn how to use version control and process
- Programming warm up
- Give material for our first code review

Program Assignment #2: **Caching / Data Alignment**
- Identifying data layout and alignment for supplied data structures and C++ classes
- Rework several data structures to reduce memory size
- Rework the supplied linked list data structure to a hot / cold data structure

Program Assignment #3: **Dynamic Memory in Real-time**
- Create a memory system within a heap
- Overloading new/delete C++ functions
- Placement New
- Create a different memory pools for the supplied C++ classes

Program Assignment #4: **C++ Efficiency**
- Implicit conversions
- Return value optimizations
- Proxy Objects
- Taking advantage of the compilers

Program Assignment #5: **Math optimizations**
- Algorithm optimization
- Refactor several matrices transformations to use SSE vector instruction set, Vector Unit or intrinsic math functions supplied by the compilers.

*Program Assignment #6: **Reading / Writing files** (undergrad only - here for reference)*
- *Read / Write Buffers*
- *Data Layout*

Program Assignment #7: **Load in Place**
- Store data in contiguous memory footprint
- Load data in binary format into ONE memory block
- Perform pointer fix-up

Program Assignment #8: **Commercial profiler**
- Configure and setup Profiler/Analyzer (hard)
- Identify hot spots (screen grab)
- Refactor
- Analyze again

Final project

Given a particle system that dynamically updates several particles. Every particle and controlling object is dynamically allocated, often in very large blotted data structures. Each particle is controlled per frame by its own unique math transformations that are unoptimized.   The memory allocations in the system are slow and fragments memory as the number of particles increase. Some of the system uses STL in a very inefficient manor. Many C++ classes are inefficient and naïve in nature.

Students refactor this system to:
- Maximize the number of particles to be processed with the given memory and performance constraint
    - Keep the frame rate constant to specified
    - Keep the memory within a fixed specified size
- Real working system
    - Program needs to cleanly be created and destroyed with no memory or resource leaks
    - Error free
    - Warning level 3 free or higher
- Dynamic monitoring for development, such as:
    - Total memory consumption
    - Memory / particle ratio
    - Performance cycles
    - Particle stats
- It's a contest
    - See how you can improve to original system.
- A reflective paper describing the optimization issues presented in this project, 5 or more pages in length

Final Exam

- A comprehensive final exam, covering the concepts of this class.
- Closed book written exam.
    - 11 week of class
- You must pass the final exam (60% or higher) to receive a passing grade in the class.
    - *It's not an easy exam*

Piazza Discussion forum

- Statistics show: students who participate more and help other students do better!
    - The correlation is ridiculous!
        - Poor understanding / poor participation.
        - Great understanding / Great participation
    - As you master the material, help others learn!
        - You're in the master's program so master it!
- Everyone is ***expected*** and encouraged to participate on the Piazza discussion forum. All class-related discussion here this term.

- The quicker you begin asking questions on Piazza (rather than via emails), the quicker you'll benefit from the collective knowledge of your classmates and instructors. I encourage you to ask questions when you're struggling to understand a concept.

- All correspondence that is not personal in nature should be vectored through Piazza

- Sensitive material, use Piazza private note, not email.
- Keep the forum professional and positive, help each other out.
    - Karma really pays off here.
    - Help each other whenever you can.
        - There will be a section where you'll need help (trust me).

NOTE: Do **_NOT_** post until you have watched the entire lecture **_FIRST_** (in class or online)
        This will prevent frustration on all sides (members asking or answering questions)

Final Exam

- A comprehensive final exam, covering the concepts of this class.
- Closed book written exam.
    - 11 week of class
- You must pass the final exam (60% or higher) to receive a passing grade in the class.
    - _It's not an easy exam_

C++ Proficiency Programs

- There will be at least 8 weekly proficiency exams, validating your C++ fundamental knowledge
- The programs are easy to implement, but is a motivator for those who need encouragement to learn more thoroughly the basics C++ material.
- Topics range from:
    - Overloading, pointers, STL and Templates
    - vTable, inheritance, C-Strings and debugging
- If you already know the material, the assignments are a way to validate that knowledge

Perforce Submissions
- Every is expected to submit at least 10 submissions a week to perforce.
- The biggest reason students get into trouble with software design:
    - Not working on the material frequently enough
    - Taking too large of a bite of the design
- Both are fixed with this _Perforce RULE_
- Even my simplest programs take 10-20 submissions.
    - For these project assignments my average is 40-400 submissions, so 10 will be no problem.
- Detailed perforce changelist comments are expected

Piazza Discussion forum

- Statistics show: students who participate more and help other students <u>do better</u>!
    - The correlation is ridiculous!
        - Poor understanding / poor participation.
        - Great understanding / Great participation
    - As you master the material, help others learn!
        - You're in the master's program so master it!
- Everyone is expected and encouraged to participate on the Piazza discussion forum. All class-related discussion here this term.
- The quicker you begin asking questions on Piazza (rather than via emails), the quicker you'll benefit from the collective knowledge of your classmates and instructors. We encourage you to ask questions when you're struggling to understand a concept.
- All correspondence that is not personal in nature should be vectored through Piazza
- Sensitive material, use Piazza private channel.

Collaborating together on programming assignments

- You are encourage to work together
    - Use the Piazza forums heavy
    - Even share your material with others in the common directory
- Everyone is 100% responsible for the work they do.
    - If you get help with a section of code,
    - Please refactor the code the ***snot out of it***
        - Comment and understand that material
        - Transform the code to ***make it yours***
    - Be able to answer ***any*** question regarding the code you commit
- System for Detecting Software Plagiarism
    - We will be using MOSS - Measure of Software Similarity (Standford University)
        - Indicates possible code infringements (plagiarism)
        - MOSS - will detect the similarity independent of naming convention, indentation style or formatting, it compares abstract syntax tree of your code.
- If you gain significant support / help from another student
    - Fully disclose the support / help you had in a Readme.txt file submitted with your assignments.
        - Disclosing the help, is ***not permission*** for copying the code.
        - Only there to clarify and acknowledge help you were given from a fellow student.
- Modifying any Unit Test to alter the outcome results is also an ***Academic Integrity Violation***
- If you are stuck and find yourself even tempted to plagiarize
    - Ask for help !!!!
        - Use on Piazza -> Visit during offices hours, make an appointment
    - ***Don't ever compromise your integrity!***

Tentative Class Schedule

Date	Lecture	Activity	Due
Week 1 16-Sep	Overview General Optimizations Perforce	PA1 - Assessment Basics1 - Debugging	
Week 2 23-Sep	Caching Data Alignment Hot/Cold Data Structures	PA2 - Hot/Cold data structures Basics2 - Overloading	PA1 Basics1
Week 3 30-Sep	Pointers Memory System	PA3 - Memory System Basics3 - Pointers	PA2 Basics2
Week 4 7-Oct	Memory System Detail Implementation Testing and Verification	Basics4 - C Strings	Basics3
Week 5 14-Oct	Implicit Conversions Proxy Objects Return Value Optimization	PA4 - Proxy, Implicit, RVO Basics5 - Inheritance	PA3 - complete Basics4
Week 6 21-Oct	Intrinsics - SSE, SIMD Matrix Math	PA5 - SSE Math Basics6 - vTable	PA4 Basics5
Week 7 28-Oct	File system Load in Place	PA7 - Load in Place Basics7 - STL Templates	PA5
Week 8 4-Nov	Memory Overloading Refactoring Particle System	Particle System PA8 - Profiling Basics8 - Templates	PA7 Basics7
Week 9 11-Nov	Profilers Strings PSE		Basics8
Week 10 18-Nov	C++ 11 / Boost Review		PA8
Week 11 25-Nov	Final Exam Performance Contest		Particle System

September 16, 2014   Last day to add classes to AQ2014 schedule
September 23, 2014   Last day to drop classes with no penalty, Last day to select pass/fail option
September 24, 2014   Grades of "W" assigned for AQ2014 classes dropped on or after this day
September 30, 2014   Last day to select auditor status
October 28, 2014   Last day to withdraw from AQ2014 classes

Course Policies

Changes to Syllabus

This syllabus is subject to change as necessary during the quarter. If a change occurs, it will be thoroughly addressed during class, posted under Announcements in D2L and sent via email.

Online Course Evaluations

Evaluations are a way for students to provide valuable feedback regarding their instructor and the course. Detailed feedback will enable the instructor to continuously tailor teaching methods and course content to meet the learning goals of the course and the academic needs of the students. They are a requirement of the course and are key to continue to provide you with the highest quality of teaching. The evaluations are anonymous; the instructor and administration do not track who entered what responses. A program is used to check if the student completed the evaluations, but the evaluation is completely separate from the student's identity. Since 100% participation is our goal, students are sent periodic reminders over three weeks. Students do not receive reminders once they complete the evaluation. Students complete the evaluation online in CampusConnect.

Academic Integrity and Plagiarism

This course will be subject to the university's academic integrity policy. More information can be found at http://academicintegrity.depaul.edu/. If you have any questions be sure to consult with your professor.

Academic Policies

All students are required to manage their class schedules each term in accordance with the deadlines for enrolling and withdrawing as indicated in the University Academic Calendar. Information on enrollment, withdrawal, grading and incompletes can be found at: cdm.depaul.edu/enrollment.

Students with Disabilities

Students who feel they may need an accommodation based on the impact of a disability should contact the instructor privately to discuss their specific needs. All discussions will remain confidential.
To ensure that you receive the most appropriate accommodation based on your needs, contact the instructor as early as possible in the quarter (preferably within the first week of class), and make sure that you have contacted the Center for Students with Disabilities (CSD) at: csd@depaul.edu.

Lewis Center 1420, 25 East Jackson Blvd.
Phone number: (312)362-8002
Fax: (312)362-6544
TTY: (773)325.7296

Retroactive withdrawal

This policy exists to assist students for whom extenuating circumstances prevented them from meeting the withdrawal deadline. During their college career students may be allowed one medical/personal administrative withdrawal and one college office administrative withdrawal, each for one or more courses in a single term. Repeated requests will not be considered. Submitting an appeal for retroactive withdrawal does not guarantee approval. Information on enrollment, withdrawal, grading and incompletes can be found at: http://www.cdm.depaul.edu/Enrollment-Policies.aspx