

CSC 342 - Computer Systems

From Metaverse

Contents

- 1 Semester Hours
- 2 Course Coordinator
- 3 Catalog Description
- 4 Textbooks
- 5 Course Goals
- 6 Prerequisites by Topic
- 7 Major Topics Covered in the Course
- 8 Learning Outcomes (Cross-Referenced with Program Outcomes)
- 9 Laboratory Projects
- 10 Estimate CSAB Category Content
- 11 Oral and Written Communication
- 12 Social and Ethical Issues
- 13 Theoretical Content
- 14 Problem Analysis
- 15 Solution Design
- 16 Course Assessment
 - 16.1 2008
 - 16.1.1 Proposed changes from last offering
 - 16.1.2 Items to be assessed, assesment mechanism, and success criteria
 - 16.2 2007
 - 16.2.1 Proposed changes from last offering
 - 16.2.2 Items to be assessed, assesment mechanism, and success criteria
 - 16.2.3 Assessment

Semester Hours

4sh

Course Coordinator

Joel Hollingsworth

Catalog Description

This course involves the study of the basic building blocks of modern computer systems. Topics include digital logic, machine-level representation of data, assembly-level organization, operating system

primitives and concurrency. Prerequisites: CSC 230. Corequisite: MTH 206. Offered Fall.

Textbooks

Computer Systems: A Programmer's Perspective
Bryant and O'Hallaron
Prentice Hall, 2003, ISBN 0-13-034074-X

The C Programming Language (Second Edition - ANSI C)
Kernighan and Ritchie
Prentice Hall, 1988, ISBN 0-13-110362-8

Course Goals

Examine the current interface between hardware and software. Establish an understanding of modern computer organization. Establish modern examples of computer architecture. Demystify how a computer works.

Prerequisites by Topic

DS2. Basic Logic
DS4. Basics of counting
DS6. Discrete Probability
PF1. Fundamental programming constructs
PF3. Fundamental Data Structures

Major Topics Covered in the Course

AR1. Digital logic and digital systems
AR2. Machine level representation of data
AR3. Assembly level machine organization
AR4. Memory system organization and architecture
AR5. Interfacing and communication
AR6. Functional Organization
AR7. Multiprocessing and alternative architectures
AR8. Performance enhancements

Learning Outcomes (Cross-Referenced with Program Outcomes)

The student will be able to:

1. Use the C programming language to directly use operating system primitives. (PO 7)
2. Manipulate data at bit-level. (PO 5,6)
3. Define the following numbering systems: unsigned integer, one's complement, two's complement, and IEEE floating-point. (PO 5,6)
4. Implement simple systems using digital logic. (PO 5,6)
5. Simplify Boolean algebraic statements using Karnaugh Maps. (PO 5,6)

6. Read and understand x86-style assembly language. (PO 5,6,9)
7. Define and discuss the implications of stack discipline. (PO 5,6,9)
8. Discuss the technological and ethical concerns of buffer overflow. (PO 4)
9. Use an HCL-style language to define a sequential and pipelined processor.
10. Convert data between binary, decimal, and hexadecimal formats. (PO 5,6)
11. Adequately work at a Linux command line for both the development of programs and the manipulation of file. (PO 7)
12. Work in teams as both a leader and a participant. (PO 3)

Laboratory Projects

- Lab 1: Manipulating Bits (2 weeks)
- Lab 2: Defusing a Binary Bomb (2 weeks)
- Lab 3: The Buffer Bomb (2 weeks)
- Lab 4: Code Optimization (2 weeks)
- Lab 5: Implementing a Shell (2 weeks)
- Lab 6: Implementing a Web Proxy (2 weeks)

Estimate CSAB Category Content

- Data Structures (2 core)
- Algorithms (2 core)
- Software Design
- Computer Architecture (30 core/2 advanced)
- Programming Languages (4 core)

Oral and Written Communication

During the discussion of stack discipline we discuss the ethical issues involved with buffer overflow.

Social and Ethical Issues

None.

Theoretical Content

Majority of algorithmic theoretical content was based on probability and digital logic.

Problem Analysis

The four lab assignments required students to analyze possible solutions to the given issues. These assignments had significant problem analysis build into the suggested method for project completion.

Solution Design

The four lab assignments required students to design and implement unique solutions to low-level problems.

Course Assessment

2008

Proposed changes from last offering

The addition of the Proxy Lab (implementing a web proxy) will provide a second systems-level project using the C programming language and multithreaded programming techniques. The Shell Lab and Code Optimization lab were used in the last offering of this course. By adding the Proxy Lab the course will tend greater towards a systems programming course than it has in the past.

Items to be assessed, assesment mechanism, and success criteria

The 6 labs will be completed in teams of size 2. After each of the labs, I will survey the students on the effectiveness of their partners and the team as a whole. Also, I will be assessing the Proxy Lab. I expect that the average grade for this lab should be greater than 75%.

2007

Proposed changes from last offering

This course has been renamed and reworked to include operating systems topics (the OS course is no more).

Items to be assessed, assesment mechanism, and success criteria

This course incorporates the study of three major topics: C programming language, computer architecture and organization, and operating systems. I plan to choose 1 project from each of these areas to assess in an attempt to determine how well each of the topics are being taught and received. I will use this information to determine which areas to improve in the next iteration.

Assessment

Lab 5 required the students to write a UNIX-style shell program using standard system calls in the C programming language. The average grade for this lab was 98. The lab scores tend to be very high, so I set a base score of 85 for passing this internal assessment. The students did very well. This was the last of their assignments requiring them to write C code. I was happy with their ability to write C code and understand the use of pointers.

Lab 3 required the students to defuse a binary "bomb" program by exploring disassembled code and their current understanding of how the hardware works. The average grade for this lab was 100. Again, I set the base score of 85 for passing this internal assessment. The students worked very hard and did well. I was please with their understanding of computer organization and assembly code by the end of the lab.

Lab 6 requires the students to implement a version of the malloc and free system calls. The students must have a good understanding of system calls and how the operating system is handling memory management. The average grade for this lab was 80.25. Using a base score of 85 for passing, the students did not achieve this goal. Since this was the last of the labs assigned I imagine a little bit of fatigue was present during this lab. I plan to set aside more time for this topic during the next iteration of this course.

Program Outcome 2:

This course made extensive use of groups for the 6 labs that occurred during the semester. The students performed well in these groups, though there was a student that decided to take advantage of being in groups. Whenever groups are used there needs to be some feedback mechanism to the faculty member about the amount of work provided by each member. The students are very adverse about "turning some in" who does not work at the same level.

Program Outcome 10:

The students explored and chose an algorithm for the maintaining of a heap. The students were required to select an efficient algorithm as the grade given for this lab was based partly on efficiency of the algorithm. The students did a very nice job on this portion of their lab.

Retrieved from "http://trumpy.cs.elon.edu/metaverse/wiki/CSC_342_-_Computer_Systems"

- This page was last modified 17:37, 22 August 2008.
- This page has been accessed 1,033 times.
- [Privacy policy](#)
- [About Metaverse](#)
- [Disclaimers](#)