WSU Five-Year Program Review

Self-Study

Cover Page

Department:	Computer Science
Program:	Computer Science AS/BS
Semester Submitted:	Spring 2012
Self-Study Team Chair: External to the University but within the discipline	*Dr. Nicole Anderson* Assistant Professor Winona State University 859 30th Ave SE Rochester, MN 55904 nanderson@winona.edu 507.285-7480
Self-Study Team Members: External to the University but within the discipline	*Dr. Kirk Love* Department Chair, Computer Science Utah Valley University 800 West University Parkway Orem, UT 84058 kirk.love@uvu.edu 801. 863-8852
Internal to the University but external to the College	*Dr. Brett Ellis* Information Technology Vice President Weber State University Ogden, UT 84408 bretellis@weber.edu 801.626.7660
Internal to the University and internal to the College	*Ms. JoEllen Jonsson* Assistant Professor Weber State University 1503 University Circle Ogden, UT 84408-1503 jjonsson@weber.edu 801.626.6910
CS Contact Information: **Phone:** **Email:**	*Dr. Delroy Brinkerhoff*, Associate Professor 801. 626.7345 dbrinkerhoff@weber.edu

Version Date: February 2012

A. Brief Introductory Statement

The Computer Science Department (CS) is a part of the College of Applied Science and Technology (COAST) at Weber State University (WSU). Students may pursue the following degree options in the Computer Science program:

- Bachelor of Science in Computer Science
- Associate of Applied Science in Computer Science
- Minor in Computer Science
- Component of a Bachelor's of Integrated Studies (BIS)
- Certificate in Game Development

In the Associate program, students learn the fundamentals of software design and implementation. The fundamentals include project management, web development, the behavior of common data structures, database design and development, computer architecture, designing and using networks, and programming experience in both the Java and C++ programming languages.

Students are further guided to select appropriate general education courses that complement their experience in the computer science department. These general education courses develop the student's verbal and writing communication skills, and their ability to solve problems using mathematics and physics.

Bachelor's-level courses expand the student's earlier experiences while also allowing them to tailor and focus their advanced training. Required courses include operating systems, computational structures (computer-centric mathematics and algorithm analysis), advanced software engineering, and formal computing languages (computability based on theoretical models of computers). Students also select and specialize in at least one of Java, C++, or C#.

Students must select a minimum of three addition elective courses, which are grouped into four focus areas: Master's degree preparation, web development, mobile development, and network security. Although elective courses are grouped into focus areas, students may choose to take electives from different groups.

B. Mission Statement

Weber State University's mission statement is:

> Weber State University provides associate, baccalaureate and master degree programs in liberal arts, sciences, technical and professional fields. Encouraging freedom of expression and valuing diversity, the university provides excellent educational experiences for students through extensive personal contact among faculty, staff and students in and out of the classroom. Through academic programs,

> research, artistic expression, public service and community-based learning, the university serves as an educational, cultural and economic leader for the region. (Approved by the Board of Regents July 2011)

In harmony with the University's mission, the Department of Computer Science has adopted the following vision statement:

> To become and be recognized as the outstanding undergraduate program in applied Computer Science in the Western United States. Specifically, to be recognized by employers as the best program to produce graduates who are quickly productive and produce software and computer systems of the highest quality.

To achieve this goal, the Department of Computer Science has initiated the process of becoming ABET accredited and so chooses to express as its mission the goal of graduating students who achieve the following program educational objectives. (The WSU CS department adopts the ABET definition of program educational objectives as "broad statements that describe what graduates are expected to attain within a few years of graduation.")

Students

1. Will conduct themselves professionally and ethically at all times, and will understand the professional, ethical, legal, security, social responsibilities of computing professionals
2. Have developed and practice the skills necessary for self-learning
3. Proficient at solving problems
4. Able to function effectively and to collaborate collegially as a part of a team
5. Proficient at analyzing, designing, and validating software with contemporary modeling languages and tools
6. Proficient at implementing software systems with at least one contemporary high-level programming language
7. Proficient at designing and documenting test cases and test plans
8. Proficient with at least one operating system
9. Proficient at designing and using databases

To guide and focus the activities of the department to achieve these program educational objectives, the department has adopted a set of student learning outcomes, which are presented in the table on the following page, and which conform to the ABET definition of student outcomes as describing "what students are expected to know and be able to do by the time of graduation." It is necessary that the department's student learning outcomes demonstrate an articulation with the ABET required student learning outcomes, and this articulation is also demonstrated in the following table. A second table demonstrates the same articulation but is organized by the ABET outcomes to ease the task of verifying that all ABET outcomes are appropriately and correctly enabled.

WSU Student Learning Outcomes	Enabled ABET Outcomes
1. Students will understand the importance of and will practice professional and ethical behavior, and will understand the professional, ethical, legal, security, and social responsibilities of computing professionals	(e) An understanding of professional, ethical, legal, security and social issues and responsibilities (g) An ability to analyze the local and global impact of computing on individuals, organizations, and society
2. Students will be able to read and understand manuals, documentation, and technical literature, find and understand sources of information, and learn on their own what they need to continue to perform professionally after graduation	(i) An ability to use current techniques, skills, and tools necessary for computing practice. (h) Recognition of the need for and an ability to engage in continuing professional development
3. Students will be able to solve new problems and to express their new solutions appropriately	(a) An ability to apply knowledge of computing and mathematics appropriate to the discipline (b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution (j) An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices
4. Students will be able to function as a team member and carry out assigned tasks	(d) An ability to function effectively on teams to accomplish a common goal
5. Students will have the knowledge and the skills needed to be employable, and to be immediately and continuously productive	(c) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs (i) An ability to use current techniques, skills, and tools necessary for computing practice
6. Students will have a basic understanding of computer theory, software design and operation, project management, databases, networking, and computer hardware	(a) An ability to apply knowledge of computing and mathematics appropriate to the discipline (b) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs (j) An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices
7. Students will understand algorithm design and how to express and how to implement algorithms using a variety of notation, programming languages, and paradigms	(a) An ability to apply knowledge of computing and mathematics appropriate to the discipline (b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution (c) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs
8. Students will be able to debug computer programs	(b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution
9. Students will be able to express themselves clearly both verbally and in writing	(f) An ability to communicate effectively with a range of audiences
10. Students will be able to critically evaluate the quality and the features of information from various sources and to make informed decisions about the design of information systems	(b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution (c) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs (k) An ability to apply design and development principles in the construction of software systems of varying complexity
11. Students will be prepared for graduate studies in Computer Science and will have the necessary knowledge and skills to be accepted into and succeed in relevant programs if they desire to continue their education in computer science	

Required ABET Outcomes	Corresponding WSU Student Learning Outcomes
(a) An ability to apply knowledge of computing and mathematics appropriate to the discipline	3. Students will be able to solve new problems and to express their new solutions appropriately 6. Students will have a basic understanding of computer theory, software design and operation, project management, databases, networking, and computer hardware 7. Students will understand algorithm design and how to express and how to implement algorithms using a variety of notation, programming languages, and paradigms
(b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution	3. Students will be able to solve new problems and to express their new solutions appropriately 6. Students will have a basic understanding of computer theory, software design and operation, project management, databases, networking, and computer hardware 7. Students will understand algorithm design and how to express and how to implement algorithms using a variety of notation, programming languages, and paradigms 8. Students will be able to debug computer programs 10. Students will be able to critically evaluate the quality and the features of information from various sources and to make informed decisions about the design of information systems
(c) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs	5. Students will have the knowledge and the skills needed to be employable, and to be immediately and continuously productive 7. Students will understand algorithm design and how to express and how to implement algorithms using a variety of notation, programming languages, and paradigms 10. Students will be able to critically evaluate the quality and the features of information from various sources and to make informed decisions about the design of information systems
(d) An ability to function effectively on teams to accomplish a common goal	4. Students will be able to function as a team member and carry out assigned tasks
(e) An understanding of professional, ethical, legal, security and social issues and responsibilities	1. Students will understand the importance of and will practice professional and ethical behavior, and will understand the professional, ethical, legal, security, and social responsibilities of computing professionals
(f) An ability to communicate effectively with a range of audiences	9. Students will be able to express themselves clearly both verbally and in writing
(g) An ability to analyze the local and global impact of computing on individuals, organizations, and society	1. Students will understand the importance of and will practice professional and ethical behavior, and will understand the professional, ethical, legal, security, and social responsibilities of computing professionals
(h) Recognition of the need for and an ability to engage in continuing professional development	2. Students will be able to read and understand manuals, documentation, and technical literature, find and understand sources of information, and learn on their own what they need to continue to perform professionally after graduation
(i) An ability to use current techniques, skills, and tools necessary for computing practice	2. Students will be able to read and understand manuals, documentation, and technical literature, find and understand sources of information, and learn on their own what they need to continue to perform professionally after graduation
(j) An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices	3. Students will be able to solve new problems and to express their new solutions appropriately 6. Students will have a basic understanding of computer theory, software design and operation, project management, databases, networking, and computer hardware
(k) An ability to apply design and development principles in the construction of software systems of varying complexity	10. Students will be able to critically evaluate the quality and the features of information from various sources and to make informed decisions about the design of information systems

C. Curriculum

Curriculum Map: Core Courses Articulated with Student Learning Outcomes

I = Introduced **R = Reinforced** **E = Emphasized** [1] **Core Courses in Department/Program**	**Department/Program Learning Outcomes**										
	1. Professional and ethical behavior	2. Read technical literature and learn on their own	3. Solve problems and express solutions	4.Function in teams and carry out assignments	5. Knowledge and skills need for employment	6. Theory, design, operation, project management, & DB [2]	7. Understand and use algorithms	8. Debug programs	9. Verbal and writing skills	10. Evaluate information systems	11. Preparation for graduate studies [2]
CS1400 Fundamentals of Programming	I		I		I			I			
CS1410 Object-Oriented Programming	R	I	R		R	I	I	R			I
CS2350 Web Development	R	R	R		R						
CS2420 Introduction to Data Structures & Algorithms	R	R	R		R	R	R	R			R
CS2450 Software Engineering I	R	R	R	I	R	R	R	R	I	I	
CS2550 Database Design & Application Development	R	R	R		R	R			R		
CS2650 Computer Architecture/Organization	R	R	R		R	R	R	R	R	R	R
CS2705 Network Fundamentals and Design	R	R	R		R			R		I	
MGMT2400 Project Management [3]	R		R	R	R	R			R		
CS3100 Operating Systems	R	R	R		R	R			R	R	R
CS3130 Computational Structures	R		R		R	R	R				R
CS3750 Software Engineering II	E	R	R	R	R			R	E	E	
CS4110 Concepts of Formal Languages and Algorithms			R	R	R	E	E				E
CS4230 Java Application Development [4]	E	E	E	E	E	E	E	E	E		
CS4750 Advanced Software Engineering [4]	E	E	E	E	E	E	E	E	E		
CS4790 N-Tier Web Programming [4]	E	E	E	E	E	E	E	E	E		

[1] Program improvement statistics are collected for these courses

[2] This outcome is more fully enabled through elective courses

[3] This course is taught by qualified CS faculty in support of departmental student learning outcomes

[4] Students must select one course

Version Date: February 2012

Curriculum Map: Core Courses Articulated with Program Educational Objectives

Most courses do not contribute to all objectives and no objective is fully enabled by a single course. The following table describes the course-level support for each of the department's program educational objectives.

Program Educational Objectives and Course Support									
Course	1. Will conduct themselves professionally and ethically at all times	2. Have developed and practice the skills necessary for self -learning	3. Proficient at solving problems	4. Able to function effectively and to collaborate collegially as a part of a team	5. Proficient at analyzing, designing, and validating software with contemporary modeling languages and tools	6. Proficient at implementing software systems with at least one contemporary high-level programming language	7. Proficient at designing and documenting test cases and test plans	8. Proficient with at least one operating system	9.Proficient at designing and using databases
CS1030[1]	x	x	x		x	x		x	x
CS1400	x	x	x	x				x	
CS1410	x	x	x		x	x	x	x	
CS2420	x	x	x		x	x			
CS2450	x	x	x	x	x				
CS2350									
CS2550	x	x	x		x				x
CS2650	x	x	x		x			x	
CS2705	x	x	x	x					
MGMT 2400[2]	x	x	x	x					
CS3100	x	x	x			x		x	
CS3130	x	x	x	x		x			
CS3750	x	x	x	x	x	x	x	x	x
CS4110	x	x	x	x		x			
CS4230[3]	x	x	x	x	x	x	x	x	x
CS4750[3]	x	x	x	x	x	x	x	x	x
CS4790[3]	x	x	x	x	x	x	x	x	x

[1] This is a preparatory not a required course; students may test-out or satisfy the requirement with high school classes

[2] This course is taught by qualified CS faculty in support of program educational objectives

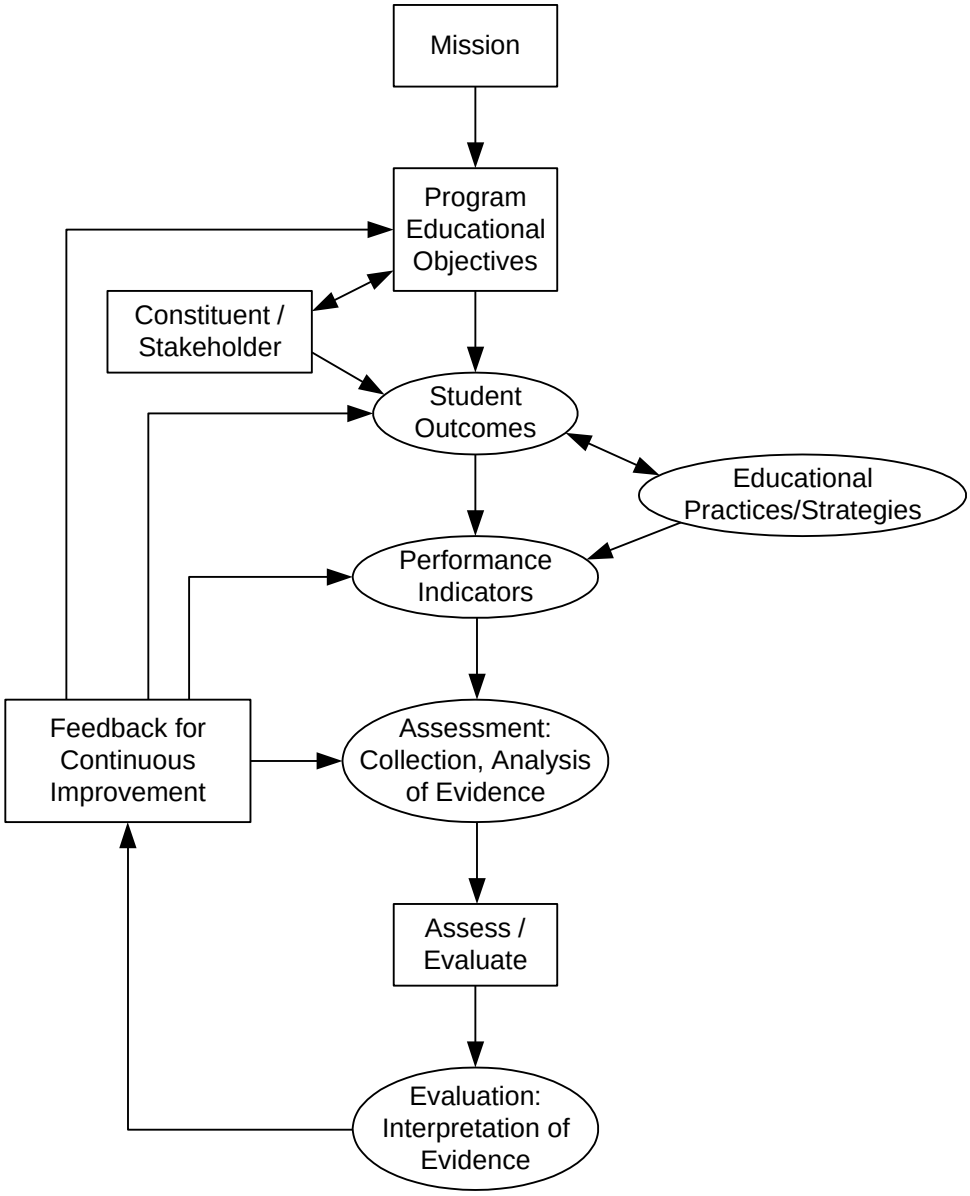
[3] Students must select one course

Version Date: February 2012

D. Student Learning Outcomes and Assessment

The WSU Dept. of Computer Science collects assessment data for each of its core courses as a part of its continuous program improvement process. The program improvement process is detailed in the diagram below. The process begins with the combined university and department mission statements presented previously. The department has established program educational objectives in harmony with the institution's mission and has derived a set of student outcomes that define a trajectory leading students to obtain the program educational objectives within a few years of graduation.

The program educational objectives and student outcomes are also defined in cooperation with a community of primary stakeholders: faculty (contributing experience, observation, and research), an industrial advisory committee (whose members hire the department's graduates), students (data is gathered from graduates and their employers), and a



comparison of programs offered at other institutions. Stakeholders meet twice per year (Fall and Spring semester) and reevaluate the program educational objectives. The semiannual review process insures that (a) the program educational objectives remain germane, and (b) the department's educational practices and strategies are enabling graduates to achieve the objectives in a reasonable amount of time.

The following list of student outcomes define the fundament skills that a student should attain at the completion of their study in computer science at WSU:

1. Students will understand the importance of and will practice professional and ethical behavior, and will understand the professional, ethical, legal, security, and social responsibilities of computing professionals
2. Students will be able to read and understand manuals, documentation, and technical literature, find and understand sources of information, and learn on their own what they need to continue to perform professionally after graduation
3. Students will be able to solve new problems and to express their new solutions appropriately
4. Students will be able to function as a team member and carry out assigned tasks
5. Students will have the knowledge and the skills needed to be employable, and to be immediately and continuously productive
6. Students will have a basic understanding of computer theory, software design and operation, project management, databases, networking, and computer hardware
7. Students will understand algorithm design and how to express and how to implement algorithms using a variety of notation, programming languages, and paradigms
8. Students will be able to debug computer programs
9. Students will be able to express themselves clearly both verbally and in writing
10. Students will be able to critically evaluate the quality and the features of information from various sources and to make informed decisions about the design of information systems
11. Students will be prepared for graduate studies in Computer Science and will have the necessary knowledge and skills to be accepted into and succeed in relevant programs if they desire to continue their education in computer science

The student outcomes influence the selection and development of appropriate educational (i.e., pedagogical) practices and strategies. Together, the student outcomes and the educational practices guide the specification of a set of performance indicators, which clearly describe the various performance levels that students demonstrate. Assessment data is collected throughout and at the end of each course. This data is used for course-level improvement. Program-level data is collected at the end of emphasizing courses (see section C above) and drives the continuous program improvement. Program-level data is evaluated in the context of set of department-defined scoring rubrics that are articulated with the student outcomes. The Dept. of Computer Science is currently in the process of defining the performance indicators and the associated scoring rubrics.

When complete, the results of the analysis of the program-level assessment data provides feedback used to update the program educational objectives, the student outcomes, performance indicators and the full spectrum of data collection and analysis techniques.

The Department of Computer Science collects and analyzes program-level assessment data on the core courses on a five-year cycle as a part of its continuous improvement process. The following table summarizes the data collection schedule.

Course	2011-2012	2012-2013	2013-2014	2014-2015	2015-2016
CS1410 Object-Oriented Programming	C	A	I		C
CS2420 Introduction to Data Structures & Algorithms	C	A	I		C
CS2450 Software Engineering I	C	A	I		C
CS3130 Computational Structures	C	A	I		C
CS1400 Fundamentals of Programming		C	A	I	
CS2550 Database Design & Application Development		C	A	I	
CS2705 Network Fundamentals and Design		C	A	I	
CS3230 Internet Multimedia Services and Applications Using Java		C	A	I	
CS2350 Web Development			C	A	I
CS2650 Computer Architecture/Organization			C	A	I
CS3100 Operating Systems			C	A	I
CS4110 Concepts of Formal Languages and Algorithms for Computing			C	A	I
CS 3750 Software Engineering II				C	A
CS 4230 Java Application Development				C	A
CS 4750 Advanced Software Engineering				C	A
CS 4790 N-Tier Web Programming				C	A

CD – Collect Data
AD – Analyze Data
I – Implement Improvements

The definition of appropriate assessment procedures is currently underway for the first four courses (CS1410, CS 2420, CS 2450, and CS 3130). Currently the instructional content targeting student outcomes has been identified and the corresponding assessments have been specified, which now makes it possible to begin data collection and analysis. Articulating student outcomes with specific exam questions enables tracking student performance through automated testing tools (WSU has developed and uses Chi-Tester, a tool that supports this feature). The following tables summarize the course content, the associated outcomes, and the corresponding assessments.

Instructional Content CS 1410 Object-Oriented Programming in C++	**Student Outcomes**	**Assessment**
1. Basics		
1.1. Using Microsoft visual studio	5	Programs 1 - 11
1.2. The compilation process: the preprocessor, the compiler, the linker	5	Programs 1 - 11
1.3. Multi-file programs	5	Programs 4, 6 - 11
2. Simple Programs (variables, constants, operators, & casting)	1, 3, 5, 6, 8, 11	Programs 1 - 11 Exam 1: 1 - 23 Exam 2: 14 - 15
3. Program using flow-of-control statements (if, switch, for, while, do, break, and continue)	1, 3, 5, 6, 8, 11	Programs 2, 3, 5, 11 Exam 1: 24 - 58
4. Structures and enumerations	1, 3, 5, 6, 8, 11	Exam 2: question 1
4.1. Fields / members		Program 4
4.2. Pointers and references (content vs. address, address of and indirection operators)		Program 4 Exam 2: 2 & 3 Exam 4: 24
5. Functions	1, 3, 5, 6, 7, 8, 11	
5.1. Definition		Program 4, 6 - 11
5.2. Declaration / prototype		Program 4, 6 - 11
5.3. Calls (pass-by-value, reference, and pointer)		Program 4, 6 - 11 Exam 2: 4 - 6, 18 - 20 Exam 4: 25
5.4. Function overloading, recursion, and default arguments		Program 6 Exam 2: 22 - 23
6. Arrays, array function arguments	1, 3, 5, 6, 8, 11	Program 5 Exam 2: 9 - 12, 23 - 25, 27
7. C-strings and string objects	1, 2, 3, 5, 6, 8, 11	
7.1. c-string functions		Programs 5, 11 Exam 2: 7 - 8, 26
7.2. string class member functions		Programs 5, 11

Instructional Content CS 1410 Object-Oriented Programming in C++	**Student Outcomes**	**Assessment**
7.3. Command line arguments: argc & argv		Programs 5 Exam 2: 15 - 17
7.4. Ascii codes		
8. Classes and objects	1, 3, 5, 6, 8, 11	
8.1. Encapsulation, member data and functions, modifiers (public, private, & protected)		Programs 6-10 Exam 2: 16 - 18 Exam 3: 26 - 29, 39 - 41, 43 - 44
8.2. Constructors and destructors; the copy constructor; conversion constructors		Programs 6-10 Exam 3: 1 - 2, 33 - 35
8.3. The `this` pointer		
9. Class relations	1, 2, 3, 5, 6, 7, 8, 11	Exam 3: 42
9.1. UML diagrams		Programs 9 & 10 Exam 2: 11 - 15
9.2. implementing class relations in C++: inheritance, association, aggregation, composition, & dependency		Programs 9 & 10 Exam 2: 19 - 25
10. Polymorphism	1, 3, 5, 6, 8, 11	
10.1. virtual functions, casting, and function overriding		Program 10 Exam 4: 12 - 19, 21 - 22
10.2. pure virtual functions and abstract classes		Program 10
11. Overloaded operators	1, 3, 5, 6, 8, 11	Exam 3: 38
11.1. Overloading arithmetic operators and >> and <<		Programs 7-9 Exam 2: 3 - 10 Exam 3: 30 - 32
11.2. friend functions		Programs 7-9

Instructional Content CS 1410 Object-Oriented Programming in C++	**Student Outcomes**	**Assessment**
		Exam 3: 36 - 37
12. Memory management	1, 3, 5, 6, 8, 11	
12.1. Static versus dynamic instantiation		Program 10
12.2. Stack and heap		Program 10
12.3. New and delete operators		Program 10
13. I/O stream classes: ifstream, ofstream, fstream	1, 2, 3, 5, 6, 8, 11	Program 11
13.1. Stream functions		Program 11 Exam 4: 23
13.2. Text versus binary files		Exam 4: 7 - 11
13.3. Manipulators and formatting functions		Program 11
13.4. Error detection: good, bad, fail		Program 11
14. Templates	1, 3, 5, 6, 8, 11	Program 10 Exam 4: 3 - 4, 20
15. Exceptions	1, 3, 5, 6, 8, 11	
15.1. The purpose of exceptions		Exam 4: 1 - 2
15.2. try / catch blocks		Exam 4: 5 - 6

Instructional Content **CS 2420 Introduction to Data Structures and Algorithms**	**Student Outcomes**	**Assessment**
1 Review of CS 1410 concepts	2, 3, 5, 6, 7, 8, 11	Two or three challenging homework assignments are given as review. A common assignment used is a Big Int calculator class which performs addition, subtraction, multiplication, and division, for both negative and positive numbers. Another is a fully functional roman numeral class, with similar mathematical operators. For item 1 given in the Contest List, each assignment attempts to review five to seven of the nine listed review items. It takes roughly three to four weeks to review all concepts through homework and lecture. Assessment is done with weekly quizzes on these concepts. Homework assignments are also graded. These concepts are all assessed in a midterm.
2.2 and 2.3 Singly linked lists and iterators	2, 3, 5, 6, 7, 8, 11	An initial homework assignment has students implementing additional methods for a linked list class. These include deleting nodes by value, deleting all nodes by value (in one pass), deleting the smallest item, finding the kth element and returning its info. Iterators are added into this assignment. Students must make iterators act similar to STL list iterators, with a few modifications. The iterators should be able to suppose operator overloads for +, -, ++, --, overloaded * for dereferencing, and overloaded [] for array like access. Sample code is given in main() which provides test cases to ensure the student code meets the expected output. Assessment is again done with weekly quizzes on these

Instructional Content **CS 2420 Introduction to Data Structures and Algorithms**	**Student Outcomes**	**Assessment**
		concepts. The homework assignment is also graded. These concepts are all assessed in a midterm.
2.4 Doubly linked lists and 3 Stacks and Queues	2, 3, 5, 6, 7, 8, 11	A homework assignment covering stacks and queues are given. A lecture is given on stacks, queues, and priority queues. The expected implementation of the homework is to effectively write a class which handles all functionality of stacks, queues, and priority queus, but does so internally using a doubly linked list. Students are required to modify their prior singly linked list into a doubly linked list. Then the student must implement all necessary stack, queue, and priority queue methods. Sample code is given in main() which provides test cases to ensure the student code meets the expected output. Assessment is again done with weekly quizzes on these concepts. The homework assignment is also graded. These concepts are all covered in a midterm.
2.5 Circular linked lists	7, 11	This is only lectured. Occasionally this is covered in a midterm.
4. Hash tables	2, 3, 5, 6, 7, 8, 11	A homework assignment for hash tables are given. The student must write his or her own hash algorithm. The resulting object must be stored in the hash table, which internally is implemented as an array of linked lists. The homework covers closed hashing. The assignment also ties together multiple review concepts from content list item #1 in ways that students typically had not yet encountered. Specifically, the students must learn to work with multiple classes simultaneously. The student must also understand how to properly work with pointers as arrays, and how to create many linked lists in an array.

Instructional Content **CS 2420 Introduction to Data Structures and Algorithms**	**Student Outcomes**	**Assessment**
		Sample code is given in main() which provides test cases to ensure the student code meets the expected output. Open hashing, array based concept, and probing techniques are lectured but not assessed. Assessment is again done with weekly quizzes on these concepts. The homework assignment is also graded. These concepts are all covered in a final exam.
5. Algorithmic efficiency	2, 6, 11	This topic covered in every subsequent lecture. As each new algorithm is described, its efficiency in time and space are analyzed. This is heavily tested in both quizzes and the final exam. One variation of an upcoming sort assignment does have students identify which possible sort algorithms are used by measuring how long it takes to complete.
6. Sort and search algorithms	2, 3, 5, 6, 7, 8, 11	Each search and sort algorithm is heavily tested in both quizzes and the final exam. Because textbooks supply these algorithms freely, the assignment does not require students to solve a problem by implementing code. Rather, the student needs to provide a visual display to how sorting actually processes. One variation of an has have students identify which possible sort algorithms are used by measuring how long it takes to complete.
7.1 Sorted binary trees	2, 3, 5, 6, 7, 8, 11	A homework assignment is given which requires the student to generate a parse tree to take a normal mathematical expression given as a C string, place it into a parse tree, then compute the solution to that expression. The student also needs to print out the

Instructional Content CS 2420 Introduction to Data Structures and Algorithms	Student Outcomes	Assessment
		expression again from the tree in pre-order, in-order, and post-order (Reverse Polish notation) fashion. Occasionally functors are included as part of the implementation for this assignment. Traversal methods are frequently tested in both quizzes and in the final exam.
7.2 AVL trees and B trees	2, 3, 5, 6, 7, 8, 11	Due to the lack of time typically found at the end of each semester, only one of these two are assessed in a homework assignment. The assignment is fairly straightforward. Each tree needs a handful of commonly used methods. The textbook provides code for some, concepts for others. The assignment is to complete the methods in which the book did not provide the code. Insertion and deletion algorithms are assessed in both quizzes and the final exam.
8 Graphs	2, 3, 5, 6, 7, 8, 11	A homework assignment is given in which students are given a PDF containing a graph of roughly 20-30 nodes and 50-70 edges. The student then needs to provide a program which asks the user for a starting node, and then lists the shortest path and path sequence needed to each other node. The student also needs to print out the graph using breadth first and depth first traversal to ensure the graph was implemented in code correctly. Breadth first, depth first, and Dijkstra's algorithm are covered on the final exam. They are not covered in a quiz, as the semester is drawing to a close.

Instructional Content CS 2450 Software Engineering I	Student Outcomes	Assessment
1.1. Steps to problem solving	3, 7, 9	Problem solving consists of six steps: 1. Identify the problem (What is the problem?) 2. Understand the problem (What is involved with the problem? What does the client want? Maybe the client does not know what they want. Make sure you know the client.) 3. Identify alternative ways to solve the problem (Create a list. Maybe talk with others. Make sure they could be acceptable solutions.) 4. Select the best way to solve the problem from the list of alternative solutions (What are the pros and cons of each solution?) 5. List the instructions that enable you to solve the problem using the selected solution (Create a numbered list of instructions) 6. Evaluate the solution (Did it satisfy the needs of the client with the problem?) Use these steps to solve the problem such as: - What to do this evening? - Where to eat dinner?
1.2. Why projects fail	1,2,3,9,10	Find a failed Software Project. Create a PowerPoint with graphics and sources as to why it failed (you can use http://www.codinghorror.com/blog/2006/05/the-long-dismal-history-of-software-project-failure.html as resources to find a project) There should be one slide describing the project, one slide describing why it failed and one slide with your source(s)
1.7. Working as a team	4	Fill out group survey and discuss different personalities. Apply throughout the semester as Professor meets with teams in verbal environment discussing and re-emphasizing personalities
2.1. System request	2-6, 9, 10	Create a system request similar to the one on page 61 using Professor Anderson as the Project Sponsor. The Business need will be to improve the program. Then look at page 58 and create a feasibility analysis including the technical, economic, and

Instructional Content CS 2450 Software Engineering I	Student Outcomes	Assessment
		organizational aspects similar to the one on page 63. The economic might be difficult depending on your system request but try your best. You can also use the project sponsor as a resource for information. There is no page requirement. Just make sure you do a thorough job and think about the opportunity costs (if you do this you can't do something else) and the ROI (return on your investment - is this project better to do than another).
2.2. Selecting a project	4,10	As a team, think about your Computer Science Department and choose an idea that could improve student satisfaction within your educational experience. Create a system request similar to the one on page 61 using Professor Anderson as the Project Sponsor. The Business need will be to improve the program.
3.0 Managing the project	2,3,9	Chapter 3, questions 2, 5, 7, 11
3.3.2 Project charter	2,4,9	Page 95. Do 3-4 the project charter
4.3. Requirements strategies	2,4,9	Chapter 4, questions 1-2, 5, 15
4.4. Gathering requirements	3,4,9,10	Create a list of questions for the client (the professor) regarding your system request. Email the list to the client by Jan 31st at midnight. When the client responds, use that information plus all other information you have gathered to create a list of the functional and nonfunctional business requirements for your system request.
5.1. Activity diagrams	3,4,5,9,10	Based upon the current project create an activity diagram and review the diagram as a team
5.2. Use case diagrams	3,4,5,9,10	Based upon the current project create a use case diagram and review the diagram as a team
6.2. CRC cards	3,4,5,9,10	Using the provided template, fill out the CRC cards for your project.
6.3. Class diagrams	3,4,5,9,10	As a team, create a class diagram for your project
7.1. Sequence diagrams	3,4,5,9,10	Based upon the current project create a sequence diagram and then review it with your team
7.2. CRUD analysis	3,4,5,9,10	As a team perform a CRUD analysis for your system
8.1. Validating the	3,4,5,9,10	Perform a walkthrough with your peers validating

Instructional Content CS 2450 Software Engineering I	Student Outcomes	Assessment
analysis		the activity, use case, sequence, and class diagrams
9.2. Normalization	3,4,5,9,10	As a team, create an ERD
10. Human computer interface	3,4,5,9,10	For the assigned project, design the graphical user interface to meet the client's needs within the scope of the project. As a team, review the documentation and confirm that the GUI does indeed meet functional requirements.
11.2. Deployment diagram	3,4,5,9,10	Create a deployment diagram for the current project and then review it with your team
11.3. Security requirements	3,4,5,9,10	Determine any security requirements for the current project
12.1 Testing plan	3,4,5,9,10	Create a plan to test the project to ensure that it meets all functional and non-functional requirements
12.2. Maintenance plan	3,4,5,9,10	Create a maintenance plan for the project to ensure that it future changes will be handled according the strategy defined within the scope of the project

Instructional Content **CS 3130 Computational Structures**	**Student Outcomes**											**Assessment**
Outcomes	1	2	3	4	5	6	7	8	9	10	11	
1. Discrete Math Structure	x	x	x	x	x	x			x	x	x	Quiz #1,#2/Exam #1/Final Exam Team Assignment #1
1.1. Definition	x	x	x	x	x	x			x	x	x	Quiz #1,#2/Exam #1/Final Exam Team Assignment #1
1.2. Operations	x	x	x	x	x	x			x	x	x	Quiz #1,#2/Exam #1/Final Exam Team Assignment #1
1.3. Properties of Operations	x	x	x	x	x	x			x	x	x	Quiz #1,#2/Exam #1/Final Exam Team Assignment #1
2. Application and Theory of Sets		x	x		x	x					x	Quiz #1,#2/Exam #1/Final Exam
2.1. Set notation and definition		x	x		x	x					x	Quiz #1,#2/Exam #1/Final Exam
2.2. Elements and member of a Set		x	x		x	x					x	Quiz #1,#2/Exam #1/Final Exam
2.3. Subsets		x	x		x	x					x	Quiz #1,#2/Exam #1/Final Exam
2.4. Operations on Sets, including Intersection, Union, Difference, Symmetric Difference		x	x		x	x					x	Quiz #1,#2/Exam #1/Final Exam
2.5. Algebraic Properties of Set operations		x	x		x	x					x	Quiz #1,#2/Exam #1/Final Exam
2.6. The Addition Principle and its Application		x	x		x	x					x	Quiz #1,#2/Exam #1/Final Exam
2.7. Computer Implementation of Sets		x	x		x	x	x	x			x	Programming Assignment #1
3. Functions	x	x	x		x	x		x	x		x	Quiz #1,#2/Exam #1/Final Exam Team Assignment #1 Programming Assignment #1
3.1. Specialized form of Relation	x	x	x		x	x			x		x	Quiz #1,#2/Exam #1/Final Exam Team Assignment

												#1 Programming Assignment #1
3.2. Functions as a mapping between sets	x	x	x		x	x			x		x	Quiz #1,#2/Exam #1/Final Exam Team Assignment #1 Programming Assignment #1
3.3. Domain, Co-Domain, and Range	x	x	x		x	x			x		x	Quiz #1,#2/Exam #1/Final Exam Team Assignment #1 Programming Assignment #1
3.4. Composition of three or more functions		x	x		x	x					x	Quiz #2/Exam #1/Final Exam
3.5. Properties of Functions		x	x		x	x					x	Quiz #2/Exam #1/Final Exam Programming Assignment #1
3.5.1. One-to-one correspondence (bijection)		x	x		x	x					x	Quiz #2/Exam #1/Final Exam Programming Assignment #1
3.5.2. Everywhere defined		x	x		x	x					x	Quiz #2/Exam #1/Final Exam Programming Assignment #1
3.5.3. Onto		x	x		x	x					x	Quiz #2/Exam #1/Final Exam Programming Assignment #1
3.5.4. Invertible		x	x		x	x					x	Quiz #2/Exam #1/Final Exam Programming Assignment #1
3.6. Functions for Computer Science		x	x		x	x	x	x			x	Quiz #1/Exam #1/Final Exam
3.6.1. Characteristic Function		x	x		x	x	x	x			x	Programming Assignment #1
3.6.2. Floor function		x	x		x	x	x	x			x	Quiz #2
3.6.3. Ceiling function		x	x		x	x	x	x			x	Quiz #2
3.6.4. Hashing function		x	x		x	x	x	x			x	Quiz #2
4. Propositions and Logical		x	x		x	x	x				x	Quiz #3,#4/Exam

Operations												#2/Final Exam
4.1. Types of Statements – Declarative, Interrogative, etc.		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam
4.2. Propositional Variables		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam
4.3. Truth Tables		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam
4.4. Negation, Conjunction, Disjunction, Biconditional		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam
4.5. Implications (hypothesis and conclusion)		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam
4.6. Predicates and Quantifiers		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam
4.6.1. Universal Quantifier		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam
4.6.2. Existential Quantifier		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam
4.7. Properties of Operations on Propositions		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam
5. Logic Programming		x	x		x	x	x	x			x	Individual Assignment #2, #3/Exam #2
5.1. Prolog syntax and relations		x	x		x	x	x	x			x	Individual Assignment #2, #3/Exam #2
5.2. Application of Prolog Facts and Rules		x	x		x	x	x	x			x	Individual Assignment #2, #3/Exam #2
5.3. Modeling Real-world relationships using Prolog		x	x		x	x	x	x			x	Individual Assignment #2, #3/Exam #2
5.4. Recursion		x	x		x	x	x	x			x	Individual Assignment #2, #3/Exam #2
6. Boolean Algebras and Circuit Design		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam Individual Assignment #3
6.1. Boolean Polynomials		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam Individual Assignment #3
6.2. Lattices and Partially Ordered Sets		x	x		x	x					x	Quiz #3
6.3. Digital Logic Gates		x	x		x	x	x	x			x	Quiz #3,#4/Exam

												#2/Final Exam Individual Assignment #3
6.3.1. AND gate		x	x		x	x	x	x			x	Quiz #3,#4/Exam #2/Final Exam Individual Assignment #3
6.3.2. OR gate		x	x		x	x	x	x			x	Quiz #3,#4/Exam #2/Final Exam Individual Assignment #3
6.3.3. NOT gate		x	x		x	x	x	x			x	Quiz #3,#4/Exam #2/Final Exam Individual Assignment #3
6.4. Circuit Design		x	x		x	x	x	x			x	Quiz #3,#4/Exam #2/Final Exam Individual Assignment #3
6.4.1. Relationship with Boolean Expressions and Truth Tables		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam Individual Assignment #3
6.5. Sum of Products Expression		x	x		x	x	x	x			x	Quiz #3,#4/Exam #2/Final Exam Individual Assignment #3
6.5.1. Minimization of Sum of Products Expression		x	x		x	x	x				x	Quiz #3,#4/Exam #2/Final Exam
6.5.2. Karnaugh Maps for minimizing number of circuit components		x	x		x	x					x	Quiz #3,#4/Exam #2/Final Exam
7. Algorithms and the Growth of Functions	x	x	x		x	x	x		x		x	Quiz #3,#4/Exam #2/Final Exam Team Assignment #1
7.1. Computational Complexity	x	x	x		x	x	x		x		x	Quiz #3,#4/Exam #2/Final Exam Team Assignment #1
7.2. Definition of big-O	x	x	x		x	x	x		x		x	Quiz #3,#4/Exam #2/Final Exam Team Assignment #1
7.3. Definition of big-Θ	x	x	x		x	x	x		x		x	Quiz #3,#4/Exam #2/Final Exam

												Team Assignment #1
7.4. Interpreting algorithms expressed as pseudocode	x	x	x	x	x	x	x	x	x	x	x	Team Assignment #1
7.5. Recursion	x	x	x		x	x	x	x	x		x	Exam #1 Team Assignment #1 Individual Assignment #2
7.6. Rules for determining the Θ-class of a Function	x	x	x		x	x	x		x		x	Quiz #3,#4/Exam #2/Final Exam Team Assignment #1
8. Integers and Counting		x	x		x	x	x				x	Quiz #4/Exam #2/Final Exam
8.1. Properties of Integers		x	x		x	x	x				x	Quiz #4/Exam #2/Final Exam
8.1.1. Prime, LCM, GCD		x	x		x	x	x	x			x	Quiz #4/Exam #2/Final Exam Individual Assignment #4
8.2. Integer Representations (Base n expansions)		x	x		x	x	x	x			x	Quiz #4/Exam #2/Final Exam
8.3. Permutations		x	x		x	x	x	x			x	Quiz #4/Exam #2/Final Exam Individual Assignment #4 Team Assignment #2
8.4. Combinations		x	x		x	x	x	x			x	Quiz #4/Exam #2/Final Exam Individual Assignment #4 Team Assignment #2
8.5. The Pigeonhole Principle		x	x		x	x					x	Exam #2
9. Discrete Probability	x	x	x	x	x	x	x	x	x	x	x	Quiz #4/Exam #2/Final Exam Individual Assignment #4 Team Assignment #2
9.1. Sample Spaces	x	x	x		x	x	x	x	x		x	Quiz #4/Exam #2/Final Exam

												Individual Assignment #4 Team Assignment #2
9.2. Events	x	x	x		x	x	x	x	x		x	Quiz #4/Exam #2/Final Exam Individual Assignment #4 Team Assignment #2
9.3. Assigning Probabilities to Events	x	x	x		x	x	x	x	x		x	Quiz #4/Exam #2/Final Exam Individual Assignment #4 Team Assignment #2
10. Boolean Matrices	x	x	x	x	x	x	x	x	x	x	x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #2
10.1.Elements		x	x		x	x	x	x			x	Quiz #5/Final Exam Individual Assignment #5
10.1.1. Zero Matrix		x	x		x	x	x	x			x	Quiz #5/Final Exam Individual Assignment #5
10.1.2. Identity (Diagonal) Matrix		x	x		x	x	x	x			x	Quiz #5/Final Exam Individual Assignment #5
10.2.Operations		x	x		x	x	x	x			x	Quiz #5/Final Exam Individual Assignment #5
10.2.1. Meet		x	x		x	x	x	x			x	Quiz #5/Final Exam Individual Assignment #5
10.2.2. Join		x	x		x	x	x	x			x	Quiz #5/Final Exam Individual Assignment #5
10.2.3. Boolean product		x	x		x	x	x	x			x	Quiz #5/Final Exam Individual Assignment #5
10.3.Properties		x	x		x	x	x				x	Quiz #5/Final Exam Individual Assignment #5

11. Relations and Digraphs	x	x	x	x	x	x	x		x	x	x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3
11.1.Partitions and Coverings	x	x	x	x	x	x			x	x	x	Final Exam Team Assignment #3
11.2.Relations and Sets	x	x	x		x	x	x		x		x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3
11.3.Relations and Functions	x	x	x		x	x	x		x		x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3
11.4.Relations and Boolean Matrices	x	x	x		x	x	x		x		x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3
11.5.Representing relations as Digraphs	x	x	x		x	x	x		x		x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3
11.5.1. In-degree of nodes	x	x	x		x	x	x		x		x	Quiz #5 Team Assignment #3
11.5.2. Out-degree of nodes	x	x	x		x	x	x		x		x	Quiz #5 Team Assignment #3
11.5.3. Paths and Cycles	x	x	x		x	x	x		x		x	Quiz #5/Final Exam Team Assignment #3
11.6.Connectivity Relation	x	x	x	x	x	x	x		x	x	x	Quiz #5/Final Exam Team Assignment #3
11.7.Properties of Relations	x	x	x		x	x	x		x		x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3

11.7.1. Reflexive and Irreflexive	x	x	x		x	x	x		x		x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3
11.7.2. Symmetric, Antisymmetric, and Asymmetric	x	x	x		x	x	x		x		x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3
11.7.3. Transitive	x	x	x		x	x	x		x		x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3
11.8.Closures	x	x	x		x	x	x	x	x		x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3
11.8.1. Reflexive, Symmetric, and Transitive Closures	x	x	x		x	x	x	x	x		x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3
11.8.2. Warshall's Algorithm	x	x	x		x	x	x	x	x		x	Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3
12. Trees	x	x	x		x	x	x		x		x	Quiz #6/Final Exam Team Assignment #3
12.1.Definition of Trees	x	x	x		x	x	x		x		x	Quiz #6/Final Exam Team Assignment #3
12.2.Tree levels, parents, siblings, leaves, vertex	x	x	x		x	x	x		x		x	Quiz #6/Final Exam Team Assignment #3
12.3.N-trees	x	x	x		x	x	x		x		x	Quiz #6/Final Exam Team Assignment #3
12.4.Binary Trees and Complete Binary Trees	x	x	x		x	x	x		x		x	Quiz #6/Final Exam Team Assignment #3

13. Sequences, Strings, and Regular Expressions		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
13.1.Infinite and finite sequences		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
13.2.Recurrence relations		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
13.3.Sets corresponding to a sequence		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
13.4.Regular Expression Alphabet		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
13.5.Regular Expression over a Set		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
14. Languages and Grammars		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
14.1.Natural Language vs. Computer Language		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
14.2.Phrase Structure Grammar		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
14.3.Terminals and Nonterminals		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
14.4.Production Rules		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
14.5.Derivation Trees		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
14.6.Regular Grammars and Regular Expressions		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
15. Machines and Languages		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
15.1.Finite State Machines		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6

15.1.1. States and Alphabet		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
15.1.2. State transition table		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
15.1.3. Acceptance States		x	x		x	x	x				x	Quiz #6/Final Exam Individual Assignment #6
15.2.Language of a Machine		x	x		x	x	x	x			x	Quiz #6/Final Exam Individual Assignment #6
15.3.Moore machine		x	x		x	x	x	x			x	Quiz #6/Final Exam Individual Assignment #6

E. Academic Advising

Advising Strategy and Process

The Department of Computer Science operates on three separate campuses, and each campus has designated advising personnel. At the main campus in Ogden, Ms. Anita Proul provides simple, routine advising and major declaration. Drs. Greg Anderson and Richard Fry provide advanced advising, including transfer credit, graduation pass-off, and detailed program planning. Mr. Bradley Peterson provides all advising at the Davis campus. Mr. Ted Cowan provides all advising at the Salt Lake Community College campus.

Effectiveness of Advising

Students are encouraged to have appointments with an advisor at least once a year. During the interview, plans are created for the sequence of courses needed to complete the requirements in the amount of time designated by the students.

The effectiveness of the advising is shown through students taking the courses in correct sequence; thus, eliminating extra semesters. Those who do not meet with their advisors find their courses out of sequence and cannot take the next course due to prerequisites not being fulfilled.

Past Changes and Future Recommendations

The Department of Computer Science has created and follows an extended course schedule that rotates on a four-year cycle. The extended schedule projects the number of specific courses needed over time and the semesters when the courses are offered. Although the department follows the schedule closely, it is altered occasionally based on enrollment, demand, and resources. Working from the extended schedule allows students to better plan their individual programs. Specifically, students can better tailor their program to their work and family schedules while minimizing the number of semesters taken to complete their degree.

The following table defines the extended course schedule.
Year 1 = 2010, 2014, . . .
Year 2 = 2011, 2015, . . .
Year 3 = 2012, 2016, . . .
Year 4 = 2013, 2017, . . .
M = Main Ogden campus
D = Davis campus
S = SLCC campus
O = Online

Note that MATH 1630 will be replaced by CS 3130 in the future.

	Year 1			Year 2			Year 3			Year 4		
	Spring	**Summer**	**Fall**	**Spring**	**Summer**	**Fall**	**Spring**	**Summer**	**Fall**	**Spring**	**Summer**	**Fall**
CS 1010	M, D, O	D, O	M, O	M, D, O	D, O	M, O	M, D, O	D, O	M, O	M, D, O	, D, O	M, O
CS 1030	M, D, O	D, O	M, D, O	M, D, O	D, O	M, D, O	M, D, O	D, O	M, D, O	M, D, O	, D, O	M, D, O
CS 1400	M, D, O	O	M, D, O	M, D, O	O	M, D, O	M, D, O	O	M, D, O	M, D, O	O	M, D, O
CS 1410	M, D	O	M, O	M, D	O	M, O	M, D	O	M, O	M, D	O	M, O
CS 2350	M, D	O	M, O	M, D	O	M, O	M, D	O	M, O	M, D	O	M, O
CS 2420	M	O	M, D	M	O	M, D	M	O	M, D	M	O	M, D
CS 2450		O	M	D		M		O	M	D		M
CS 2550	M	O	M, D	M	O	M, D	M	O	M, D	M	O	M, D
CS 2650	M, D	O	O	M, D	O	O	M, D	O	O	M, D	O	O
CS 2705	M	O	M, D	M	O	M, D	M	O	M, D	M	O	M, D
CS 3030	S					M	S					M
CS 3040		S	M	S		M		S	M	S		M
CS 3100	M, D	S	M, S	M		M, S	M	S	M, S	M		M, S
CS 3210	M		S	M		S	M		S	M		S
CS 3230	M, S		M	S	D	M	M	S	M			M
CS 3540	S		M		S	M	S		M		S	M
CS 3550	M		S	M		S	M	D	S	M		S
CS 3705	M		S	M		S	M		S	M		S
CS 3730			M						M			
CS 3750		D	M, S			M, S			M, S			M, S
CS 3805	S					M	S					M
CS 3830			M				M		D			
CS 3840			M						M			S
CS 4110	M, S		M	M, S		M	M, S	D	M	M, S		M
CS 4230						M						M
CS 4280			M	S					M	S		
CS 4350	M		M	M		M	M		M, S	M	S	M
CS 4500				M		S				M		S
CS 4730	M			M			M			M		
CS 4740	S		M	S		M	S		M	S		M
CS 4750	M, S			M, S			M, S		D	M, S		
CS 4780		D	M	S		M	S		M	S		M, D
CS 4790	M		S	M	D		M			M	D	
CS 4820			S			M		D	S			M
CS 4830		S			S	M		S			S	M
MATH 1630	M	O	M, D	M	O	M, D	M	O	M, D	M	O	M, D
MGMT 2400	M, D	S	M, D	M, D		M, D, S	M, D	S	M, D	M, D, S	S	M, D, S

F. Faculty

Faculty Demographic and Diversity Information

The Computer Science program currently employs thirteen full-time faculty members and approximately twelve part-time adjunct instructors. (The number and composition of adjuncts varies over time; therefore, their information is included only in the rank/tenure data).

Main Categories	Subcategory	%
Gender	Male	100.0%
Ethnicity	Euro-American	92.3%
	Afro-American	7.7%
	Disabled	7.7%
	Veteran	15.4%
Degree	Doctorate	46.1%
	Master's	38.5%
	Bachelor's	15.4%
Rank/Tenure	Tenured	28.0%
	Tenure Track	16.0%
	Instructor	8.0%
	Adjunct	48.0%
Years Teaching	<5	46.2%
	5-20	46.2%
	>20	7.7%

Programmatic/Departmental Teaching Standards and Faculty Qualifications

Tenured faculty must meet one of the following two requirements:

1. Attainment of the earned doctorate in Computer Science or a related field plus two years of full-time industry experience, or
2. A master's degree in computer science or a related field plus five years of full-time industry experience and appropriate industry certification.

Adjuncts must have a degree in computer science or a related field and be currently active in the content area in which they are instructing. Adjuncts must submit:

- A current resume
- Copies of teaching licensure or certification
- Documentation of degree and years of related experience

Evidence of Effective Instruction

i. Regular Faculty
All faculty (both tenured and tenure track) are evaluated each semester for every class they teach. Any concerns are discussed with the department chair.

ii. Adjunct Faculty
All adjunct faculty members are evaluated each semester for every class they teach. Any concerns are discussed with the department chair.

Mentoring Activities

Faculty mentors work with adjunct faculty to improve teaching and to assist with classroom issues such as testing, syllabi, online, cheating, and classroom discipline.

Mr. Bradley Peterson and Mr. Ted Cowan manage adjunct faculty at the Davis and the SLCC Campuses respectively. They provide direct support and advice regarding syllabi, student performance, classroom ambience, instructional materials, and performance. A record of each adjunct faculty is maintained and used in assessing future employment. Any concerns are immediately discussed with the department chair.

Ongoing Review and Professional Development

Faculty members are provided opportunities in many avenues for professional development in areas of instruction, scholarship, and service. This includes taking professional courses, attending and/or presenting a professional conferences, and participating in research and scholarly discussion groups on campus.

All contract, salaried faculty are encouraged to submit proposals to the Research Scholarship and Professional Growth Committee and the Academic Resources and Computing Committee.

G. Support Staff, Administration, Facilities, Equipment, and Library

Adequacy of Staff

See Appendix C.

Adequacy of Administrative Support

The Department of Computer Science enjoys excellent support from university, which supplies essential infrastructure. The infrastructure includes campus-wide network support and a global network connection, hosting faculty web pages, Chi-Tester (an automated online exam delivery tool), and WSU Online (a web-based instructional tool that supports online and hybrid - online and in-class - instruction). The WSU Online staff routinely provides training and ongoing support and emerging technologies. Many of the CS faculty members have received Master Online Teacher certification by completing a series of workshops coordinated by the WSU Online office related to teaching techniques and current technology.

Adequacy of Facilities and Equipment

The Department of Computer Science participates in the Microsoft Developer Network Academic Alliance (MSDNAA) through which it provides to its students the essential software used in many its courses. The latest versions of the software are provided to students at no cost and include operating systems, integrated development environments (editors, compilers, debuggers, etc.), diagramming and scheduling tools, etc.

The Department of Computer Science maintains the following facilities:

Building	Room Number	Room Type/Usage
Technical Education Building Ogden Campus	103C	Computer Lab / Classroom (30 N-Computing Workstations, 1 Server)
	103D	Computer Lab / Classroom (30 N-Computing Stations, 1 Server)
	108	Computer Lab / Classroom (18 Mac Workstations, 1 Server)
	109F	Computer Lab / Classroom (22 PC Computers, 1 Server)
	109D	Computer Lab / Classroom (13 Computers)
	109C	Computer Lab / Classroom (24 N-Computing Workstations, 1 Server)
	202S	Computer Lab / Classroom (31 Computers)
	105-105	Classroom (62 seats)
	109B, 110I	Conference / Meeting Rooms
	109	Open Study Lab (20 N-Computing Workstations, 1 Server)
	110(B-G, J-K), 111(A-C)	Faculty Offices
	109A	Lab Manager Office
	110	Administrative Assistant Office
	103E	Storage (potential small computer lab)
	109E, 103A&B	Parts and Equipment Storage
	110H	Break Room
D2 Davis Campus	312	Computer Lab / Classroom (30 PC Computers)
	314	Computer Lab / Classroom (28 PC Computers)
Salt Lake Community College Meadowbrook Campus (Bldg B)	B126	Computer Lab / Classroom (23 seats & 23 PC Computers)
	B130	Computer Lab / Classroom (23 seats & 16 PC Computers)

All computer labs also include an instructor's workstation.

Adequacy of Library Resources

The Stewart Library maintains extensive subscriptions to numerous academic databases, on which faculty and students rely for research. In addition to maintaining a wide selection of printed material on location, the library is also able to retrieve books and scholarly articles from a national network of lending libraries.

H. Relationships with External Communities

Description of Role in External Communities

The Department of Computer Science enjoys the support and guidance of an Industry Advisory Committee composed of computing businesses located along the northern Wasatch Front. The advisory committee has been essential to the development of CS department's current curriculum. The committee's recommendations help keep courses current and relevant and also provide input regarding quality of student work.

Local businesses and organizations provide support to the department in several ways. They provide internships, which provide an opportunity for students to gain relevant work experience. Businesses also contact the department seeking part-time employees still in school and full-time employees following graduation.

Many local businesses also contribute financially to the department. Contributions are often in the form of student scholarships and occasionally provide for hardware or facility upgrades. See Appendix E for further details.

I. Results of Previous Program Reviews

<table>
<tr><th colspan="3">Previous Program Review: 2003-2004</th></tr>
<tr><th>Problem Identified</th><th>Action Taken:</th><th>Progress:</th></tr>
<tr><td rowspan="3">Develop a system of communication between University IT Division, lab managers and the faculty</td><td>Change in IT staff.
The CS lab manager has developed a professional relationship with the current IT personnel and has access to the IT department's secure facilities.</td><td rowspan="3">Previous issues with the IT department resulted in the CS department computers being sporadically disconnected from the network. These issues have been resolved and the IT department now hosts the CS department's networked servers.</td></tr>
<tr><td>Action to Be Taken:</td></tr>
<tr><td>Complete</td></tr>
<tr><td rowspan="4">Continue to use the Advisory Council for developing ways to improve programs</td><td>Action Taken:</td><td>Progress:</td></tr>
<tr><td>The department currently meets with the advisory committee two times each academic year: Fall and Spring semester.</td><td rowspan="3">The industry advisory committee made numerous recommendations regarding the curriculum. The department has implemented all recommendations, leading to significant improvements to the curriculum.</td></tr>
<tr><td>Action to Be Taken:</td></tr>
<tr><td>Task is ongoing</td></tr>
</table>

<table>
<tr><td rowspan="3">Hire more qualified faculty</td><td>Action Taken:</td><td>Progress:</td></tr>
<tr><td>Faculty that have retired or have been promoted are being replaced. However, staffing levels have not increased since the last review even though enrollment levels have increased.</td><td rowspan="3">Budgetary constraints preclude hiring additional faculty members at this time.</td></tr>
<tr><td>Action to Be Taken:</td></tr>
<tr><td rowspan="3">Keep course work up-to-date</td><td>There is still a need to hire more qualified faculty members.</td></tr>
<tr><td>Action Taken:</td><td>Progress:</td></tr>
<tr><td>Overhauled the curriculum increasing the amount of math and science required.
Dropped obsolete courses.
Developed new courses to support emerging technologies.</td><td rowspan="3">The curriculum overall was carried out under the direction of the industrial advisory committee. The committee has indicated that the current curriculum is appropriate for current and projected industry needs. New courses in mobile computing have been added since the last review.</td></tr>
<tr><td>Action to Be Taken:</td></tr>
<tr><td>Task is ongoing</td></tr>
</table>

<table>
<tr><td rowspan="3">Expand the opportunities for faculty to obtain additional educational credentials</td><td>Action Taken:</td><td>Progress:</td></tr>
<tr><td>All faculty are encouraged to submit grant proposals for professional growth.
Faculty are encouraged to attend courses technical and certification courses and the department and college share the expense.
The college continues to offer financial support for faculty seeking appropriate graduate degrees.</td><td rowspan="3">Since the last review:
One faculty member (with a previous master's degree) completed a bachelor's degree in CS;
Two faculty members have completed master's degrees; and
Four faculty members have completed doctorates.</td></tr>
<tr><td>Action to Be Taken:</td></tr>
<tr><td></td><td>Task is ongoing</td></tr>
<tr><td rowspan="3">Require more mathematics for the Systems Integration emphasis or more mathematics and science for the Software Engineering emphasis that will better meet ABET accreditation standards</td><td>Action Taken:</td><td>Progress:</td></tr>
<tr><td>Renamed and moved the systems integration out of the CS department.
Increased the math and science requirement for the engineering emphasis to match ABET requirements.</td><td rowspan="3">The department continues to monitor national and world-wide trends in computer science and the evolving ABET standards to ensure that its curriculum conforms to accepted standards. Current math and science standards do conform to ABET and align well with the curricula of similar institutions.</td></tr>
<tr><td>Action to Be Taken:</td></tr>
<tr><td></td><td>Complete</td></tr>
</table>

J. Action Plan for Ongoing Assessment Based on Current Self-Study Findings

Action Plan for Evidence of Learning Related Findings

Problem Identified	Action to Be Taken
The CS department must fully implement its data collection and assessment plans to assure that all student outcomes are being met and to support ABET accreditation. The first ABET accreditation visit is currently anticipated Fall semester, 2013.	Establish performance indicators for all student outcomes: Summer 2012
	Establish scoring rubrics for all performance indicators: Summer 2
	Implement the 5-year data collection schedule outlined in section D: 2012 - 2017
	Collect and analyze first year data in anticipation of first accreditation visit: 2012 - 2013
	Gather and organize assessment artifacts: 2013

Action Plan for Staff, Administration, or Budgetary Findings

Problem Identified	Action to Be Taken
Hire more qualified faculty	Current 5 Year Program Review: 2017
	Request one or more additional tenure-track faculty positions for the Department of Computer Science.

APPENDICES

Appendix A: Student and Faculty Statistical Summary for Computer Science Department

	2006-07	2007-08	2008-09	2009-10	2010-11
Student Credit Hours Total	**7,147**	**6961.5**	**7,612**	**8,839**	**10,590**
Student FTE Total	**238.23**	**232.05**	**253.73**	**294.63**	**353.00**
Student Majors					
Computer Science	**493**	**483**	**540**	**590**	**667**
Program Graduates					
Certificate					2
Associate Degree	35	28	32	34	38
Bachelor Degree	89	75	71	60	63
Student Demographic Profile	**493**	**483**	**540**	**590**	**667**
Female	38	37	39	45	66
Male	455	446	501	545	601
Faculty FTE Total	**23.43**	**22.15**	**20.36**	**18.82**	**NA**
Adjunct FTE	11.13	9.84	12.64	9.51	NA
Contract FTE	12.31	12.31	7.72	9.31	NA
Student/Faculty Ratio	**10.17**	**10.48**	**12.46**	**15.66**	**NA**

Note: Data provided by Institutional Research

Appendix B: Contract/Adjunct Faculty Profile

Name	Gender	Ethnicity	Rank	Tenure Status	Highest Degree	Years of Teaching	Areas of Expertise
Greg Anderson	M	Euro-American	Associate	Tenured	Doctorate	12	Software Engineering, Gaming, Database Development/Management, Project Management
Delroy Brinkerhoff	M	Euro-American	Associate	Tenured	Doctorate	16	Programming, operating systems, knowledge and learning
Ted Cowan	M	Afro-American	Assistant	Tenure Track	Master's	3	Unix Programming, Scripting, Software Engineering, Project Management, Online instruction and Operating Systems
David Ferro	M	Euro-American	Associate	Tenured	Doctorate	10	Early instruction in computer science, computing history and culture, usability and user-centric design, web development, service learning, online instruction
Richard Fry	M	Euro-American	Associate	Tenured	Doctorate	11	Relational DB Design, SQL Programming, N-Tier Web Development
Robert Hilton	M	Euro-American	Associate	Tenured	Master's	13	Database Theory and Design, SQL Programming, Enterprise System Development, Web Development, Mobile Development, Computer Architecture
Spencer Hilton	M	Euro-American	Instructor	Tenure Track	Master's	5	Software Engineering, Project Management, Mobile Development, Database
Joshua Jensen	M	Euro-American	Instructor	Non Tenure Track	Bachelor's	1	Software Engineering, Mobile Development, Web Development, UX Design, Database
Ronald Peterson	M	Euro-American	Associate	Tenured	Doctorate	34	Artificial intelligence, especially human language processing and cryptography.

Name	Gender	Ethnicity	Rank	Tenure Status	Highest Degree	Years of Teaching	Areas of Expertise
Bradley Peterson	M	Euro-American	Instructor	Non Tenure Track	Bachelor's	4	Parallel programming, GPU programming, data structures and algorithms
Brian Rague	M	Euro-American	Associate	Tenured	Doctorate	10	Software Engineering, parallel computing and programming languages
Garth Tuck	M	Euro-American	Assistant	Tenure Track	Master's	2	
Drew Weidman	M	Euro-American	Assistant	Tenure Track	Master's	6	Information Assurance and Network Security

Appendix C: Staff Profile

Name	Gender	Ethnicity	Job Title	Years of Employment	Areas of Expertise
Anita Proul	F	Euro-American	Secretary II	5	Office Support Student Advising
Patrick Beck	M	Euro-American	Technical Support Specialist	7	Computer Hardware and Software Networking Database Administration

Appendix D: Financial Analysis Summary

Department of Computer Science					
Cost	**06-07**	**07-08**	**08-09**	**09-10**	**10-11**
Direct Instructional Expenditures	1,322,954	1,399,151	1,092,461	1,144,647	1,140,885
Cost Per Student FTE	5,553	6,030	4,306	3,885	3,232
Funding	**06-07**	**07-08**	**08-09**	**09-10**	**10-11**
Appropriated Fund	1,281,052	1,213,518	1,080,300	1,109,031	1,100,059
Other:					
Special Legislative Appropriation					
Grants of Contracts		118,869			
Special Fees/Differential Tuition	41,902	66,764	12,161	35,616	40,825
Total	1,322,954	1,399,151	1,092,461	1,144,647	1,140,885

Note: Data provided by Provost's Office

Appendix E: External Community Involvement Names and Organizations (Industrial Advisory Committee)

Name	Organization
Sean Stromberg	Imagicom
Jim Hood	LDS Church
Michael Halverson	IRS
John Blackburn	Disney
Shawn Cowder	Boeing
Russ Reed	MarketStar
Steve Hilton	America First Credit Union
Matt Baxter	Bank of Utah
Randall J Hughes	L3
Christine Barton	Sorenson Communications
Donald Brenner	Sorenson Communications
Mattock Smith	SelectHealth
George R New	HAFB
Norm LeClair	HAFB
Chuck Crandall	WebChuck Web Design
John Minor	HAFB
David Young	Autoliv
Matt Firth	ATK
Robyn Hunter	Flying J
Kyle Andersen	IHC
Lin Richardsen	IHC
Mike Taylor	Guru Technologies

Appendix F: External Community Involvement Financial Contributions

Organization	Amount	Type
L3	$55K	Donation
IBM	$20K	Donation
SIMS	$$4.5K	Donation
Stewart Trust	$50K	Donation
Imagicom	$500	Donation
MarketStar	In Progress	Donation