Department: Computer Science
Program: Computer Science AS/BS
Semester Submitted: Spring 2012

Self-Study Team Chair:
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:
Dr. Kirk Love
Department Chair, Computer Science
Utah Valley University
800 West University Parkway
Orem, UT 84058
kirk.love@uvu.edu
801.863-8852

Dr. Brett Ellis
Information Technology Vice President
Weber State University
Ogden, UT 84408
bretellis@weber.edu
801.626.7660

Ms. JoEllen Jonsson
Assistant Professor
Weber State University
1503 University Circle
Ogden, UT 84408-1503
jjonsson@weber.edu
801.626.6910

CS Contact Information:
Dr. Delroy Brinkerhoff, Associate Professor
Phone: 801.626.7345
Email: dbrinkerhoff@weber.edu
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.
<table>
<thead>
<tr>
<th>WSU Student Learning Outcomes</th>
<th>Enabled ABET Outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td>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</td>
<td>(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</td>
</tr>
<tr>
<td>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</td>
<td>(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</td>
</tr>
<tr>
<td>3. Students will be able to solve new problems and to express their new solutions appropriately</td>
<td>(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</td>
</tr>
<tr>
<td>4. Students will be able to function as a team member and carry out assigned tasks</td>
<td>(d) An ability to function effectively on teams to accomplish a common goal</td>
</tr>
<tr>
<td>5. Students will have the knowledge and the skills needed to be employable, and to be immediately and continuously productive</td>
<td>(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</td>
</tr>
<tr>
<td>6. Students will have a basic understanding of computer theory, software design and operation, project management, databases, networking, and computer hardware</td>
<td>(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</td>
</tr>
<tr>
<td>7. Students will understand algorithm design and how to express and how to implement algorithms using a variety of notation, programming languages, and paradigms</td>
<td>(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</td>
</tr>
<tr>
<td>8. Students will be able to debug computer programs</td>
<td>(b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution</td>
</tr>
<tr>
<td>9. Students will be able to express themselves clearly both verbally and in writing</td>
<td>(f) An ability to communicate effectively with a range of audiences</td>
</tr>
<tr>
<td>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</td>
<td>(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</td>
</tr>
<tr>
<td>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</td>
<td>(k) An ability to apply design and development principles in the construction of software systems of varying complexity</td>
</tr>
</tbody>
</table>

Version Date: February 2012
<table>
<thead>
<tr>
<th>Required ABET Outcomes</th>
<th>Corresponding WSU Student Learning Outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) An ability to apply knowledge of computing and mathematics appropriate to the</td>
<td>3. Students will be able to solve new problems and to express their new solutions appropriately</td>
</tr>
<tr>
<td>discipline</td>
<td>6. Students will have a basic understanding of computer theory, software design and operation, databases,</td>
</tr>
<tr>
<td></td>
<td>networking, and computer hardware</td>
</tr>
<tr>
<td></td>
<td>7. Students will understand algorithm design and how to express and how to implement algorithms using a</td>
</tr>
<tr>
<td></td>
<td>variety of notation, programming languages, and paradigms</td>
</tr>
<tr>
<td>(b) An ability to analyze a problem, and identify and define the computing</td>
<td>3. Students will be able to solve new problems and to express their new solutions appropriately</td>
</tr>
<tr>
<td>requirements appropriate to its solution</td>
<td>6. Students will have a basic understanding of computer theory, software design and operation, databases,</td>
</tr>
<tr>
<td></td>
<td>networking, and computer hardware</td>
</tr>
<tr>
<td></td>
<td>7. Students will understand algorithm design and how to express and how to implement algorithms using a</td>
</tr>
<tr>
<td></td>
<td>variety of notation, programming languages, and paradigms</td>
</tr>
<tr>
<td></td>
<td>8. Students will be able to debug computer programs</td>
</tr>
<tr>
<td></td>
<td>10. Students will be able to critically evaluate the quality and the features of information from various</td>
</tr>
<tr>
<td></td>
<td>sources and to make informed decisions about the design of information systems</td>
</tr>
<tr>
<td>(c) An ability to design, implement, and evaluate a computer-based system, process,</td>
<td>5. Students will have the knowledge and the skills needed to be employable, and to be immediately and</td>
</tr>
<tr>
<td>component, or program to meet desired needs</td>
<td>continuously productive</td>
</tr>
<tr>
<td></td>
<td>7. Students will understand algorithm design and how to express and how to implement algorithms using a</td>
</tr>
<tr>
<td></td>
<td>variety of notation, programming languages, and paradigms</td>
</tr>
<tr>
<td></td>
<td>10. Students will be able to critically evaluate the quality and the features of information from various</td>
</tr>
<tr>
<td></td>
<td>sources and to make informed decisions about the design of information systems</td>
</tr>
<tr>
<td>(d) An ability to function effectively on teams to accomplish a common goal</td>
<td>4. Students will be able to function as a team member and carry out assigned tasks</td>
</tr>
<tr>
<td>(e) An understanding of professional, ethical, legal, security and social issues and</td>
<td>1. Students will understand the importance of and will practice professional and ethical behavior, and will</td>
</tr>
<tr>
<td>responsibilities</td>
<td>understand the professional, ethical, legal, security, and social responsibilities of computing</td>
</tr>
<tr>
<td>(f) An ability to communicate effectively with a range of audiences</td>
<td>9. Students will be able to express themselves clearly both verbally and in writing</td>
</tr>
<tr>
<td>(g) An ability to analyze the local and global impact of computing on individuals,</td>
<td>1. Students will understand the importance of and will practice professional and ethical behavior, and will</td>
</tr>
<tr>
<td>organizations, and society</td>
<td>understand the professional, ethical, legal, security, and social responsibilities of computing</td>
</tr>
<tr>
<td>(h) Recognition of the need for and an ability to engage in continuing professional</td>
<td>2. Students will be able to read and understand manuals, documentation, and technical literature, find and</td>
</tr>
<tr>
<td>development</td>
<td>understand sources of information, and learn on their own what they need to continue to perform professionally</td>
</tr>
<tr>
<td></td>
<td>after graduation</td>
</tr>
<tr>
<td>(i) An ability to use current techniques, skills, and tools necessary for computing</td>
<td>2. Students will be able to read and understand manuals, documentation, and technical literature, find and</td>
</tr>
<tr>
<td>practice</td>
<td>understand sources of information, and learn on their own what they need to continue to perform professionally</td>
</tr>
<tr>
<td></td>
<td>after graduation</td>
</tr>
<tr>
<td>(j) An ability to apply mathematical foundations, algorithmic principles, and computer</td>
<td>3. Students will be able to solve new problems and to express their new solutions appropriately</td>
</tr>
<tr>
<td>science theory in the modeling and design of computer-based systems in a way that</td>
<td>6. Students will have a basic understanding of computer theory, software design and operation, databases,</td>
</tr>
<tr>
<td>demonstrates comprehension of the tradeoffs involved in design choices</td>
<td>networking, and computer hardware</td>
</tr>
<tr>
<td>(k) An ability to apply design and development principles in the construction of</td>
<td>10. Students will be able to critically evaluate the quality and the features of information from various</td>
</tr>
<tr>
<td>software systems of varying complexity</td>
<td>sources and to make informed decisions about the design of information systems</td>
</tr>
</tbody>
</table>
C. Curriculum

Curriculum Map: Core Courses Articulated with Student Learning Outcomes

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

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
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.

<table>
<thead>
<tr>
<th>Program Educational Objectives and Course Support</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Course</strong></td>
</tr>
<tr>
<td>CS1030(^1)</td>
</tr>
<tr>
<td>CS1400</td>
</tr>
<tr>
<td>CS1410</td>
</tr>
<tr>
<td>CS2420</td>
</tr>
<tr>
<td>CS2450</td>
</tr>
<tr>
<td>CS2550</td>
</tr>
<tr>
<td>CS2650</td>
</tr>
<tr>
<td>CS2705</td>
</tr>
<tr>
<td>MGMT 2400(^2)</td>
</tr>
<tr>
<td>CS3100</td>
</tr>
<tr>
<td>CS3130</td>
</tr>
<tr>
<td>CS3750</td>
</tr>
<tr>
<td>CS4110</td>
</tr>
<tr>
<td>CS4230(^3)</td>
</tr>
<tr>
<td>CS4750(^3)</td>
</tr>
<tr>
<td>CS4790(^3)</td>
</tr>
</tbody>
</table>

\(^1\) 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 fundamental 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 a 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.

Version Date: February 2012
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.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CS1410 Object-Oriented Programming</td>
<td>C</td>
<td>A</td>
<td>I</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>CS2420 Introduction to Data Structures &amp; Algorithms</td>
<td>C</td>
<td>A</td>
<td>I</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>CS2450 Software Engineering I</td>
<td>C</td>
<td>A</td>
<td>I</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>CS3130 Computational Structures</td>
<td>C</td>
<td>A</td>
<td>I</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>CS1400 Fundamentals of Programming</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS2550 Database Design &amp; Application Development</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS2705 Network Fundamentals and Design</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS3230 Internet Multimedia Services and Applications Using Java</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS2350 Web Development</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS2650 Computer Architecture/Organization</td>
<td>C</td>
<td>A</td>
<td>I</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS3100 Operating Systems</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS4110 Concepts of Formal Languages and Algorithms for Computing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 3750 Software Engineering II</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 4230 Java Application Development</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 4750 Advanced Software Engineering</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 4790 N-Tier Web Programming</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD – Collect Data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AD – Analyze Data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I – Implement Improvements</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

Version Date: February 2012
<table>
<thead>
<tr>
<th>Instructional Content</th>
<th>Student Outcomes</th>
<th>Assessment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CS 1410 Object-Oriented Programming in C++</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| 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 | Programs 6-10  
Exam 2: 16 - 18  
Exam 3: 26 - 29, 39 - 41, 43 - 44 |
| 8.1. Encapsulation, member data and functions, modifiers (public, private, & protected) | | Programs 6-10  
Exam 3: 1 - 2, 33 - 35 |
| 8.2. Constructors and destructors; the copy constructor; conversion constructors | | |
| 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 |
<p>| 11.2. friend functions | | Programs 7-9 |</p>
<table>
<thead>
<tr>
<th>Instructional Content</th>
<th>Student Outcomes</th>
<th>Assessment</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 1410 Object-Oriented Programming in C++</td>
<td></td>
<td>Exam 3: 36 - 37</td>
</tr>
<tr>
<td>12. Memory management</td>
<td>1, 3, 5, 6, 8, 11</td>
<td></td>
</tr>
<tr>
<td>12.1. Static versus dynamic instantiation</td>
<td></td>
<td>Program 10</td>
</tr>
<tr>
<td>12.2. Stack and heap</td>
<td></td>
<td>Program 10</td>
</tr>
<tr>
<td>12.3. New and delete operators</td>
<td></td>
<td>Program 10</td>
</tr>
<tr>
<td>13. I/O stream classes: ifstream, ofstream, fstream</td>
<td>1, 2, 3, 5, 6, 8, 11</td>
<td>Program 11</td>
</tr>
<tr>
<td>13.1. Stream functions</td>
<td></td>
<td>Program 11</td>
</tr>
<tr>
<td>13.2. Text versus binary files</td>
<td></td>
<td>Exam 4: 23</td>
</tr>
<tr>
<td>13.3. Manipulators and formatting functions</td>
<td></td>
<td>Exam 4: 7 - 11</td>
</tr>
<tr>
<td>13.4. Error detection: good, bad, fail</td>
<td></td>
<td>Program 11</td>
</tr>
<tr>
<td>14. Templates</td>
<td>1, 3, 5, 6, 8, 11</td>
<td>Program 10</td>
</tr>
<tr>
<td>15. Exceptions</td>
<td>1, 3, 5, 6, 8, 11</td>
<td>Exam 4: 3 - 4, 20</td>
</tr>
<tr>
<td>15.1. The purpose of exceptions</td>
<td></td>
<td>Exam 4: 1 - 2</td>
</tr>
<tr>
<td>15.2. try / catch blocks</td>
<td></td>
<td>Exam 4: 5 - 6</td>
</tr>
<tr>
<td>Instructional Content</td>
<td>Student Outcomes</td>
<td>Assessment</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CS 2420 Introduction to Data Structures and Algorithms</td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>1 Review of CS 1410 concepts</td>
<td>2, 3, 5, 6, 7, 8, 11</td>
<td></td>
</tr>
<tr>
<td>2.2 and 2.3 Singly linked lists and iterators</td>
<td>2, 3, 5, 6, 7, 8, 11</td>
<td>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 concepts.</td>
</tr>
<tr>
<td>Instructional Content</td>
<td>Student Outcomes</td>
<td>Assessment</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------</td>
<td>------------</td>
</tr>
<tr>
<td><strong>CS 2420 Introduction to Data Structures and Algorithms</strong></td>
<td>2, 3, 5, 6, 7, 8, 11</td>
<td>concepts. The homework assignment is also graded. These concepts are all assessed in a midterm.</td>
</tr>
<tr>
<td><strong>2.4 Doubly linked lists and 3 Stacks and Queues</strong></td>
<td></td>
<td>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 queues, 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.</td>
</tr>
<tr>
<td><strong>2.5 Circular linked lists</strong></td>
<td>7, 11</td>
<td>This is only lectured. Occasionally this is covered in a midterm.</td>
</tr>
<tr>
<td><strong>4. Hash tables</strong></td>
<td>2, 3, 5, 6, 7, 8, 11</td>
<td>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.</td>
</tr>
<tr>
<td>Instructional Content</td>
<td>Student Outcomes</td>
<td>Assessment</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------</td>
<td>------------</td>
</tr>
<tr>
<td>CS 2420 Introduction to Data Structures and Algorithms</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. Algorithmic efficiency</td>
<td>2, 6, 11</td>
<td>This topic covered in every subsequent lecture. As each new algorithm is described, its efficiency in time and space are analyzed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6. Sort and search algorithms</td>
<td>2, 3, 5, 6, 7, 8, 11</td>
<td>Each search and sort algorithm is heavily tested in both quizzes and the final exam.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.1 Sorted binary trees</td>
<td>2, 3, 5, 6, 7, 8, 11</td>
<td>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</td>
</tr>
<tr>
<td>Instructional Content</td>
<td>Student Outcomes</td>
<td>Assessment</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------</td>
<td>------------</td>
</tr>
<tr>
<td>CS 2420 Introduction to Data Structures and Algorithms</td>
<td>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.</td>
<td></td>
</tr>
<tr>
<td>7.2 AVL trees and B trees</td>
<td>2, 3, 5, 6, 7, 8, 11</td>
<td>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.</td>
</tr>
<tr>
<td>8 Graphs</td>
<td>2, 3, 5, 6, 7, 8, 11</td>
<td>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.</td>
</tr>
<tr>
<td>Instructional Content CS 2450 Software Engineering I</td>
<td>Student Outcomes</td>
<td>Assessment</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>------------------</td>
<td>------------</td>
</tr>
</tbody>
</table>
| **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](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 |
<table>
<thead>
<tr>
<th>Instructional Content</th>
<th>Student Outcomes</th>
<th>Assessment</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 2450 Software Engineering I</td>
<td></td>
<td>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).</td>
</tr>
<tr>
<td>2.2. Selecting a project</td>
<td>4,10</td>
<td>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.</td>
</tr>
<tr>
<td>3.0 Managing the project</td>
<td>2,3,9</td>
<td>Chapter 3, questions 2, 5, 7, 11</td>
</tr>
<tr>
<td>3.3.2 Project charter</td>
<td>2,4,9</td>
<td>Page 95. Do 3-4 the project charter</td>
</tr>
<tr>
<td>4.3. Requirements strategies</td>
<td>2,4,9</td>
<td>Chapter 4, questions 1-2, 5, 15</td>
</tr>
<tr>
<td>4.4. Gathering requirements</td>
<td>3,4,9,10</td>
<td>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.</td>
</tr>
<tr>
<td>5.1. Activity diagrams</td>
<td>3,4,5,9,10</td>
<td>Based upon the current project create an activity diagram and review the diagram as a team</td>
</tr>
<tr>
<td>5.2. Use case diagrams</td>
<td>3,4,5,9,10</td>
<td>Based upon the current project create a use case diagram and review the diagram as a team</td>
</tr>
<tr>
<td>6.2. CRC cards</td>
<td>3,4,5,9,10</td>
<td>Using the provided template, fill out the CRC cards for your project.</td>
</tr>
<tr>
<td>6.3. Class diagrams</td>
<td>3,4,5,9,10</td>
<td>As a team, create a class diagram for your project</td>
</tr>
<tr>
<td>7.1. Sequence diagrams</td>
<td>3,4,5,9,10</td>
<td>Based upon the current project create a sequence diagram and then review it with your team</td>
</tr>
<tr>
<td>7.2. CRUD analysis</td>
<td>3,4,5,9,10</td>
<td>As a team perform a CRUD analysis for your system</td>
</tr>
<tr>
<td>8.1. Validating the</td>
<td>3,4,5,9,10</td>
<td>Perform a walkthrough with your peers validating</td>
</tr>
<tr>
<td>Instructional Content CS 2450 Software Engineering I</td>
<td>Student Outcomes</td>
<td>Assessment</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>------------------</td>
<td>------------</td>
</tr>
<tr>
<td>analysis</td>
<td></td>
<td>the activity, use case, sequence, and class diagrams</td>
</tr>
<tr>
<td>9.2. Normalization</td>
<td>3,4,5,9,10</td>
<td>As a team, create an ERD</td>
</tr>
<tr>
<td>10. Human computer interface</td>
<td>3,4,5,9,10</td>
<td>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.</td>
</tr>
<tr>
<td>11.2. Deployment diagram</td>
<td>3,4,5,9,10</td>
<td>Create a deployment diagram for the current project and then review it with your team</td>
</tr>
<tr>
<td>11.3. Security requirements</td>
<td>3,4,5,9,10</td>
<td>Determine any security requirements for the current project</td>
</tr>
<tr>
<td>12.1 Testing plan</td>
<td>3,4,5,9,10</td>
<td>Create a plan to test the project to ensure that it meets all functional and non-functional requirements</td>
</tr>
<tr>
<td>12.2. Maintenance plan</td>
<td>3,4,5,9,10</td>
<td>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</td>
</tr>
<tr>
<td>Instructional Content</td>
<td>Student Outcomes</td>
<td>Assessment</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------</td>
<td>------------</td>
</tr>
<tr>
<td><strong>CS 3130 Computational Structures</strong></td>
<td>1 2 3 4 5 6 7 8 9 10 11</td>
<td></td>
</tr>
<tr>
<td><strong>Outcomes</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. Discrete Math Structure</td>
<td></td>
<td>Quiz #1,#2/Exam #1/Final Exam Team Assignment #1</td>
</tr>
<tr>
<td>1.1. Definition</td>
<td>x x x x x x</td>
<td>Quiz #1,#2/Exam #1/Final Exam Team Assignment #1</td>
</tr>
<tr>
<td>1.2. Operations</td>
<td>x x x x x x</td>
<td>Quiz #1,#2/Exam #1/Final Exam Team Assignment #1</td>
</tr>
<tr>
<td>1.3. Properties of Operations</td>
<td>x x x x x x</td>
<td>Quiz #1,#2/Exam #1/Final Exam Team Assignment #1</td>
</tr>
<tr>
<td>2. Application and Theory of Sets</td>
<td>x x x</td>
<td>Quiz #1,#2/Exam #1/Final Exam</td>
</tr>
<tr>
<td>2.1. Set notation and definition</td>
<td>x x x</td>
<td>Quiz #1,#2/Exam #1/Final Exam</td>
</tr>
<tr>
<td>2.2. Elements and member of a Set</td>
<td>x x x</td>
<td>Quiz #1,#2/Exam #1/Final Exam</td>
</tr>
<tr>
<td>2.3. Subsets</td>
<td>x x x</td>
<td>Quiz #1,#2/Exam #1/Final Exam</td>
</tr>
<tr>
<td>2.4. Operations on Sets, including Intersection, Union, Difference, Symmetric Difference</td>
<td>x x x</td>
<td>Quiz #1,#2/Exam #1/Final Exam</td>
</tr>
<tr>
<td>2.5. Algebraic Properties of Set operations</td>
<td>x x x</td>
<td>Quiz #1,#2/Exam #1/Final Exam</td>
</tr>
<tr>
<td>2.6. The Addition Principle and its Application</td>
<td>x x x</td>
<td>Quiz #1,#2/Exam #1/Final Exam</td>
</tr>
<tr>
<td>2.7. Computer Implementation of Sets</td>
<td>x x x x x x</td>
<td>Programming Assignment #1</td>
</tr>
<tr>
<td>3. Functions</td>
<td>x x x x x x x</td>
<td>Quiz #1,#2/Exam #1/Final Exam Team Assignment #1 Programming Assignment #1</td>
</tr>
<tr>
<td>3.1. Specialized form of Relation</td>
<td>x x x x x x</td>
<td>Quiz #1,#2/Exam #1/Final Exam Team Assignment</td>
</tr>
<tr>
<td>3.2. Functions as a mapping between sets</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.3. Domain, Co-Domain, and Range</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.4. Composition of three or more functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.5. Properties of Functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.5.1. One-to-one correspondence (bijection)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.5.2. Everywhere defined</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.5.3. Onto</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.5.4. Invertible</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.6. Functions for Computer Science</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.6.1. Characteristic Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.6.2. Floor function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.6.3. Ceiling function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.6.4. Hashing function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Propositions and Logical</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operations</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>4.1. Types of Statements – Declarative, Interrogative, etc.</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>4.2. Propositional Variables</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>4.3. Truth Tables</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>4.4. Negation, Conjunction, Disjunction, Biconditional</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>4.5. Implications (hypothesis and conclusion)</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>4.6. Predicates and Quantifiers</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>4.6.1. Universal Quantifier</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>4.6.2. Existential Quantifier</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>4.7. Properties of Operations on Propositions</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>5. Logic Programming</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>5.1. Prolog syntax and relations</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>5.2. Application of Prolog Facts and Rules</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>5.3. Modeling Real-world relationships using Prolog</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>5.4. Recursion</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>6. Boolean Algebras and Circuit Design</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>6.1. Boolean Polynomials</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>6.2. Lattices and Partially Ordered Sets</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>6.3. Digital Logic Gates</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Section</td>
<td>Example 1</td>
<td>Example 2</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>6.3.1. AND gate</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>6.3.2. OR gate</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>6.3.3. NOT gate</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>6.4. Circuit Design</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>6.4.1. Relationship with Boolean Expressions and Truth Tables</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>6.5. Sum of Products Expression</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>6.5.1. Minimization of Sum of Products Expression</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>6.5.2. Karnaugh Maps for minimizing number of circuit components</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>7. Algorithms and the Growth of Functions</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>7.1. Computational Complexity</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>7.2. Definition of big-O</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>7.3. Definition of big-Θ</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Chapter</td>
<td>Section</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>7.4</td>
<td>Interpreting algorithms expressed as pseudocode</td>
<td>x x x x x x x x x x x x</td>
</tr>
<tr>
<td>7.5</td>
<td>Recursion</td>
<td>x x x x x x x x x x x x</td>
</tr>
<tr>
<td>7.6</td>
<td>Rules for determining the ( \Theta )-class of a Function</td>
<td>x x x x x x x x x</td>
</tr>
<tr>
<td>8.</td>
<td>Integers and Counting</td>
<td>x x x x x x x x x x x</td>
</tr>
<tr>
<td>8.1</td>
<td>Properties of Integers</td>
<td>x x x x x x x x x x x</td>
</tr>
<tr>
<td>8.1.1</td>
<td>Prime, LCM, GCD</td>
<td>x x x x x x x x x x x</td>
</tr>
<tr>
<td>8.2</td>
<td>Integer Representations (Base representations)</td>
<td>x x x x x x x x x x x</td>
</tr>
<tr>
<td>8.3</td>
<td>Permutations</td>
<td>x x x x x x x x x x x</td>
</tr>
<tr>
<td>8.4</td>
<td>Combinations</td>
<td>x x x x x x x x x x x</td>
</tr>
<tr>
<td>8.5</td>
<td>The Pigeonhole Principle</td>
<td>x x x x x x x</td>
</tr>
<tr>
<td>9.</td>
<td>Discrete Probability</td>
<td>x x x x x x x x x x x x</td>
</tr>
<tr>
<td>9.1</td>
<td>Sample Spaces</td>
<td>x x x x x x x x x x x</td>
</tr>
<tr>
<td>9.2. Events</td>
<td>Individual Assignment #4</td>
<td>Team Assignment #2</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td></td>
<td>Quiz #4/Exam #2/Final Exam</td>
<td>Individual Assignment #4</td>
</tr>
<tr>
<td>9.3. Assigning Probabilities to Events</td>
<td>Quiz #4/Exam #2/Final Exam</td>
<td>Individual Assignment #4</td>
</tr>
<tr>
<td>10. Boolean Matrices</td>
<td>Quiz #5/Final Exam Individual Assignment #5</td>
<td>Team Assignment #2</td>
</tr>
<tr>
<td>10.1. Elements</td>
<td>Quiz #5/Final Exam Individual Assignment #5</td>
<td></td>
</tr>
<tr>
<td>10.1.1. Zero Matrix</td>
<td>Quiz #5/Final Exam Individual Assignment #5</td>
<td></td>
</tr>
<tr>
<td>10.1.2. Identity (Diagonal) Matrix</td>
<td>Quiz #5/Final Exam Individual Assignment #5</td>
<td></td>
</tr>
<tr>
<td>10.2. Operations</td>
<td>Quiz #5/Final Exam Individual Assignment #5</td>
<td></td>
</tr>
<tr>
<td>10.2.1. Meet</td>
<td>Quiz #5/Final Exam Individual Assignment #5</td>
<td></td>
</tr>
<tr>
<td>10.2.2. Join</td>
<td>Quiz #5/Final Exam Individual Assignment #5</td>
<td></td>
</tr>
<tr>
<td>10.2.3. Boolean product</td>
<td>Quiz #5/Final Exam Individual Assignment #5</td>
<td></td>
</tr>
<tr>
<td>10.3. Properties</td>
<td>Quiz #5/Final Exam Individual Assignment #5</td>
<td></td>
</tr>
<tr>
<td>11. Relations and Digraphs</td>
<td>x x x x x x x x x</td>
<td>Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td>11.1. Partitions and Coverings</td>
<td>x x x x x x x x</td>
<td>Final Exam Team Assignment #3</td>
</tr>
<tr>
<td>11.2. Relations and Sets</td>
<td>x x x x x x x x</td>
<td>Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3</td>
</tr>
<tr>
<td>11.3. Relations and Functions</td>
<td>x x x x x x x x</td>
<td>Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3</td>
</tr>
<tr>
<td>11.4. Relations and Boolean Matrices</td>
<td>x x x x x x x x</td>
<td>Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3</td>
</tr>
<tr>
<td>11.5. Representing relations as Digraphs</td>
<td>x x x x x x x x</td>
<td>Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3</td>
</tr>
<tr>
<td>11.5.1. In-degree of nodes</td>
<td>x x x x x x x x</td>
<td>Quiz #5 Team Assignment #3</td>
</tr>
<tr>
<td>11.5.2. Out-degree of nodes</td>
<td>x x x x x x x x</td>
<td>Quiz #5 Team Assignment #3</td>
</tr>
<tr>
<td>11.5.3. Paths and Cycles</td>
<td>x x x x x x x x</td>
<td>Quiz #5/Final Exam Team Assignment #3</td>
</tr>
<tr>
<td>11.6. Connectivity Relation</td>
<td>x x x x x x x x x</td>
<td>Quiz #5/Final Exam Team Assignment #3</td>
</tr>
<tr>
<td>11.7. Properties of Relations</td>
<td>x x x x x x x x x</td>
<td>Quiz #5/Final Exam Individual Assignment #5 Team Assignment #3</td>
</tr>
<tr>
<td>11.7.1. Reflexive and Irreflexive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.7.2. Symmetric, Antisymmetric, and Asymmetric</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.7.3. Transitive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.8.Closures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.8.1. Reflexive, Symmetric, and Transitive Closures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.8.2. Warshall’s Algorithm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12. Trees</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.1. Definition of Trees</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.2. Tree levels, parents, siblings, leaves, vertex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.3. N-trees</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.4. Binary Trees and Complete Binary Trees</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-----------------------------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Quiz #6/Final Exam Individual Assignment #6</td>
<td>Quiz #6/Final Exam Individual Assignment #6</td>
<td>Quiz #6/Final Exam Individual Assignment #6</td>
</tr>
</tbody>
</table>

Version Date: February 2012
<table>
<thead>
<tr>
<th>Section</th>
<th>Table</th>
<th>Quiz #6/Final Exam</th>
<th>Assignment #6</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.1.1. States and Alphabet</td>
<td>x x x x x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>15.1.2. State transition table</td>
<td>x x x x x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>15.1.3. Acceptance States</td>
<td>x x x x x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>15.2. Language of a Machine</td>
<td>x x x x x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>15.3. Moore machine</td>
<td>x x x x x</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>
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.
<table>
<thead>
<tr>
<th></th>
<th>Year 1</th>
<th>Year 2</th>
<th>Year 3</th>
<th>Year 4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Spring</td>
<td>Summer</td>
<td>Fall</td>
<td>Spring</td>
</tr>
<tr>
<td>CS 1010</td>
<td>M, D, O</td>
<td>D, O</td>
<td>M, O</td>
<td>M, D, O</td>
</tr>
<tr>
<td>CS 1030</td>
<td>M, D, O</td>
<td>D, O</td>
<td>M, D</td>
<td>M, D, O</td>
</tr>
<tr>
<td>CS 1400</td>
<td>M, D, O</td>
<td>M, D, O</td>
<td>M, D, O</td>
<td>M, D, O</td>
</tr>
<tr>
<td>CS 1410</td>
<td>M, D</td>
<td>O, M</td>
<td>M, D</td>
<td>M, D</td>
</tr>
<tr>
<td>CS 2350</td>
<td>M, D</td>
<td>O, M</td>
<td>M, D</td>
<td>M, D</td>
</tr>
<tr>
<td>CS 2420</td>
<td>M, D</td>
<td>O, M</td>
<td>M, D</td>
<td>M, D</td>
</tr>
<tr>
<td>CS 2450</td>
<td>M, D</td>
<td>O, M</td>
<td>M, D</td>
<td>M, D</td>
</tr>
<tr>
<td>CS 2550</td>
<td>M, D</td>
<td>O, M</td>
<td>M, D</td>
<td>M, D</td>
</tr>
<tr>
<td>CS 3730</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 3805</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 3830</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 3850</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 4230</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 4280</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 4350</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 4500</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 4730</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 4740</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 4750</td>
<td>D</td>
<td>M</td>
<td>D</td>
<td>M</td>
</tr>
<tr>
<td>CS 4780</td>
<td>D</td>
<td>M</td>
<td>D</td>
<td>M</td>
</tr>
<tr>
<td>CS 4920</td>
<td>S</td>
<td>S</td>
<td>M</td>
<td>S</td>
</tr>
<tr>
<td>CS 4930</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>MATH 1630</td>
<td>O</td>
<td>M, D</td>
<td>M, D</td>
<td>M, D</td>
</tr>
<tr>
<td>MGMT 2400</td>
<td>M, D</td>
<td>S</td>
<td>S</td>
<td>M, D</td>
</tr>
</tbody>
</table>

Version Date: February 2012
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).

<table>
<thead>
<tr>
<th>Main Categories</th>
<th>Subcategory</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender</td>
<td>Male</td>
<td>100.0%</td>
</tr>
<tr>
<td>Ethnicity</td>
<td>Euro-American</td>
<td>92.3%</td>
</tr>
<tr>
<td></td>
<td>Afro-American</td>
<td>7.7%</td>
</tr>
<tr>
<td></td>
<td>Disabled</td>
<td>7.7%</td>
</tr>
<tr>
<td></td>
<td>Veteran</td>
<td>15.4%</td>
</tr>
<tr>
<td>Degree</td>
<td>Doctorate</td>
<td>46.1%</td>
</tr>
<tr>
<td></td>
<td>Master’s</td>
<td>38.5%</td>
</tr>
<tr>
<td></td>
<td>Bachelor’s</td>
<td>15.4%</td>
</tr>
<tr>
<td>Rank/Tenure</td>
<td>Tenured</td>
<td>28.0%</td>
</tr>
<tr>
<td></td>
<td>Tenure Track</td>
<td>16.0%</td>
</tr>
<tr>
<td></td>
<td>Instructor</td>
<td>8.0%</td>
</tr>
<tr>
<td></td>
<td>Adjunct</td>
<td>48.0%</td>
</tr>
<tr>
<td>Years Teaching</td>
<td>&lt;5</td>
<td>46.2%</td>
</tr>
<tr>
<td></td>
<td>5-20</td>
<td>46.2%</td>
</tr>
<tr>
<td></td>
<td>&gt;20</td>
<td>7.7%</td>
</tr>
</tbody>
</table>

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:

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

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>
<thead>
<tr>
<th>Problem Identified</th>
<th>Action Taken:</th>
<th>Progress:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Develop a system of communication between University IT Division, lab managers and</td>
<td>Change in IT staff. The CS lab manager has developed a professional relationship with the current</td>
<td>Previous issues with the IT department resulted in the CS department computers being sporadically</td>
</tr>
<tr>
<td>the faculty</td>
<td>IT personnel and has access to the IT department’s secure facilities.</td>
<td>disconnected from the network. These issues have been resolved and the IT department now hosts the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CS department’s networked servers.</td>
</tr>
<tr>
<td><strong>Action to Be Taken:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complete</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Action Taken:</strong></td>
<td><strong>Progress:</strong></td>
</tr>
<tr>
<td>Continue to use the Advisory Council for developing ways to improve programs</td>
<td>The department currently meets with the advisory committee two times each academic year: Fall and</td>
<td>The industry advisory committee made numerous recommendations regarding the curriculum. The</td>
</tr>
<tr>
<td></td>
<td>Spring semester.</td>
<td>department has implemented all recommendations, leading to significant improvements to the curriculum.</td>
</tr>
<tr>
<td></td>
<td><strong>Action to Be Taken:</strong></td>
<td></td>
</tr>
<tr>
<td>Task is ongoing</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| Hire more qualified faculty | **Action Taken:**  
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. | **Progress:**  
Budgetary constraints preclude hiring additional faculty members at this time. |
|----------------------------|-------------------------------------------------|-------------------------------------------------|
| **Action to Be Taken:**  
There is still a need to hire more qualified faculty members. | **Action to Be Taken:**  
Overhauled the curriculum increasing the amount of math and science required.  
Dropped obsolete courses.  
Developed new courses to support emerging technologies. | **Progress:**  
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. |
| Keep course work up-to-date | **Action to Be Taken:**  
Overhauled the curriculum increasing the amount of math and science required.  
Dropped obsolete courses.  
Developed new courses to support emerging technologies. | **Action to Be Taken:**  
Task is ongoing |
<table>
<thead>
<tr>
<th>Expand the opportunities for faculty to obtain additional educational credentials</th>
<th><strong>Action Taken:</strong></th>
<th><strong>Progress:</strong></th>
</tr>
</thead>
<tbody>
<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>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>
<td></td>
</tr>
</tbody>
</table>

| **Action to Be Taken:** | **Task is ongoing** |

<table>
<thead>
<tr>
<th>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</th>
<th><strong>Action Taken:</strong></th>
<th><strong>Progress:</strong></th>
</tr>
</thead>
<tbody>
<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>The department continues to monitor national and worldwide 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>
<td></td>
</tr>
</tbody>
</table>

| **Action to Be Taken:** | **Complete** |
J. Action Plan for Ongoing Assessment Based on Current Self-Study Findings

Action Plan for Evidence of Learning Related Findings

<table>
<thead>
<tr>
<th>Problem Identified</th>
<th>Action to Be Taken</th>
</tr>
</thead>
<tbody>
<tr>
<td>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.</td>
<td>Establish performance indicators for all student outcomes: Summer 2012</td>
</tr>
<tr>
<td></td>
<td>Establish scoring rubrics for all performance indicators: Summer 2012</td>
</tr>
<tr>
<td></td>
<td>Implement the 5-year data collection schedule outlined in section D: 2012 - 2017</td>
</tr>
<tr>
<td>The first ABET accreditation visit is currently anticipated Fall semester, 2013.</td>
<td>Collect and analyze first year data in anticipation of first accreditation visit: 2012 - 2013</td>
</tr>
<tr>
<td></td>
<td>Gather and organize assessment artifacts: 2013</td>
</tr>
</tbody>
</table>

Action Plan for Staff, Administration, or Budgetary Findings

<table>
<thead>
<tr>
<th>Problem Identified</th>
<th>Action to Be Taken</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hire more qualified faculty</td>
<td>Current 5 Year Program Review: 2017</td>
</tr>
<tr>
<td></td>
<td>Request one or more additional tenure-track faculty positions for the Department of Computer Science.</td>
</tr>
</tbody>
</table>
APPENDICES

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

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

*Note: Data provided by Institutional Research*
# Appendix B: Contract/Adjunct Faculty Profile

<table>
<thead>
<tr>
<th>Name</th>
<th>Gender</th>
<th>Ethnicity</th>
<th>Rank</th>
<th>Tenure Status</th>
<th>Highest Degree</th>
<th>Years of Teaching</th>
<th>Areas of Expertise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Greg Anderson</td>
<td>M</td>
<td>Euro-American</td>
<td>Associate</td>
<td>Tenured</td>
<td>Doctorate</td>
<td>12</td>
<td>Software Engineering, Gaming, Database Development/Management, Project Management</td>
</tr>
<tr>
<td>Delroy Brinkerhoff</td>
<td>M</td>
<td>Euro-American</td>
<td>Associate</td>
<td>Tenured</td>
<td>Doctorate</td>
<td>16</td>
<td>Programming, operating systems, knowledge and learning</td>
</tr>
<tr>
<td>Ted Cowan</td>
<td>M</td>
<td>Afro-American</td>
<td>Assistant</td>
<td>Tenure Track</td>
<td>Master’s</td>
<td>3</td>
<td>Unix Programming, Scripting, Software Engineering, Project Management, Online instruction and Operating Systems</td>
</tr>
<tr>
<td>David Ferro</td>
<td>M</td>
<td>Euro-American</td>
<td>Associate</td>
<td>Tenured</td>
<td>Doctorate</td>
<td>10</td>
<td>Early instruction in computer science, computing history and culture, usability and user-centric design, web development, service learning, online instruction</td>
</tr>
<tr>
<td>Richard Fry</td>
<td>M</td>
<td>Euro-American</td>
<td>Associate</td>
<td>Tenured</td>
<td>Doctorate</td>
<td>11</td>
<td>Relational DB Design, SQL Programming, N-Tier Web Development</td>
</tr>
<tr>
<td>Spencer Hilton</td>
<td>M</td>
<td>Euro-American</td>
<td>Instructor</td>
<td>Tenure Track</td>
<td>Master’s</td>
<td>5</td>
<td>Software Engineering, Project Management, Mobile Development, Database</td>
</tr>
<tr>
<td>Joshua Jensen</td>
<td>M</td>
<td>Euro-American</td>
<td>Instructor</td>
<td>Non Tenure Track</td>
<td>Bachelor’s</td>
<td>1</td>
<td>Software Engineering, Mobile Development, Web Development, UX Design, Database</td>
</tr>
<tr>
<td>Ronald Peterson</td>
<td>M</td>
<td>Euro-American</td>
<td>Associate</td>
<td>Tenured</td>
<td>Doctorate</td>
<td>34</td>
<td>Artificial intelligence, especially human language processing and cryptography.</td>
</tr>
<tr>
<td>Name</td>
<td>Gender</td>
<td>Ethnicity</td>
<td>Rank</td>
<td>Tenure Status</td>
<td>Highest Degree</td>
<td>Years of Teaching</td>
<td>Areas of Expertise</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td>--------------</td>
<td>-----------</td>
<td>-------------------</td>
<td>----------------</td>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Bradley Peterson</td>
<td>M</td>
<td>Euro-American</td>
<td>Instructor</td>
<td>Non Tenure Track</td>
<td>Bachelor’s</td>
<td>4</td>
<td>Parallel programming, GPU programming, data structures and algorithms</td>
</tr>
<tr>
<td>Brian Rague</td>
<td>M</td>
<td>Euro-American</td>
<td>Associate</td>
<td>Tenured</td>
<td>Doctorate</td>
<td>10</td>
<td>Software Engineering, parallel computing and programming languages</td>
</tr>
<tr>
<td>Garth Tuck</td>
<td>M</td>
<td>Euro-American</td>
<td>Assistant</td>
<td>Tenure Track</td>
<td>Master’s</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Drew Weidman</td>
<td>M</td>
<td>Euro-American</td>
<td>Assistant</td>
<td>Tenure Track</td>
<td>Master’s</td>
<td>6</td>
<td>Information Assurance and Network Security</td>
</tr>
</tbody>
</table>
## Appendix C: Staff Profile

<table>
<thead>
<tr>
<th>Name</th>
<th>Gender</th>
<th>Ethnicity</th>
<th>Job Title</th>
<th>Years of Employment</th>
<th>Areas of Expertise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anita Proul</td>
<td>F</td>
<td>Euro-American</td>
<td>Secretary II</td>
<td>5</td>
<td>Office Support, Student Advising</td>
</tr>
<tr>
<td>Patrick Beck</td>
<td>M</td>
<td>Euro-American</td>
<td>Technical Support Specialist</td>
<td>7</td>
<td>Computer Hardware and Software, Networking, Database Administration</td>
</tr>
</tbody>
</table>
Appendix D: Financial Analysis Summary

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

*Note: Data provided by Provost's Office*
**Appendix E: External Community Involvement Names and Organizations**

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

**Appendix F: External Community Involvement Financial Contributions**

<table>
<thead>
<tr>
<th>Organization</th>
<th>Amount</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>L3</td>
<td>$55K</td>
<td>Donation</td>
</tr>
<tr>
<td>IBM</td>
<td>$20K</td>
<td>Donation</td>
</tr>
<tr>
<td>SIMS</td>
<td>$4.5K</td>
<td>Donation</td>
</tr>
<tr>
<td>Stewart Trust</td>
<td>$50K</td>
<td>Donation</td>
</tr>
<tr>
<td>Imagicom</td>
<td>$500</td>
<td>Donation</td>
</tr>
<tr>
<td>MarketStar</td>
<td>In Progress</td>
<td>Donation</td>
</tr>
</tbody>
</table>