Ohlone College
Program Review Report

● Program Description and Scope:

1. Program Review Title: Computer Science
3. Review Type: Instructional Disciplines
4. Program/Departments: Computer Science (07000)
5. Authority Code: 53-Dean, Business, Technology, and Learning Resources
6. External Regulations: Yes No

7. Provide a brief narrative that describes the instructional program/discipline.

The Computer Science (CS) program prepares students for employment in Information Technology such as computer programming and for transfer to 4-year schools in order to pursue an advanced degree. The CS Faculty recognizes that Ohlone serves two types of students: those who intend on getting an advanced degree (i.e. transfer) and those who are interested in enhancing employment skills, whether unemployed or on the job already. Fortunately there is no conflict in these two areas since the techniques needed are similar. The main difference is that transfer students need to focus on theory and the skills students on current tools and technologies. We teach the theory using current tools and technologies.

8. Describe how the program specifically serves students, faculty and staff.

Because of our proximity to Silicon Valley, Ohlone is an important resource for skill enhancement and transfer education. Staff and faculty all work together to support the needs of the students. Our CS instructors meet with faculty in local 4-year schools as well as local high schools and MVROP to discuss mutual areas of interest such as articulation agreements. We use these discussions to get valuable advice about how to improve the CS program and the skills in our students. This is taken into consideration as we fine tune our courses and create new courses.

9. Describe how the program addresses current needs and applies current technologies.

In the computer studies programs, "current" is a key driving force. Although our CS faculty keep the core concepts as our primary focus, we also must continually keep abreast of changes in programming languages, operating systems, and software tools. The CS faculty takes this very seriously and make sure that our students are aware of the changing nature of this field. We subscribe to academic alliances with major companies such as Oracle, IBM, and Microsoft to ensure access to the latest software development environment/tools and online training.

10. Discuss the impact of the program on the college and/or other programs.

We recognize that we have an opportunity to reach out to other disciplines in the college to demonstrate the relevance of CS and to find ways to collaborate. Past successful efforts include assisting in the testing and data acquisition for Currucinet, the addition of courses to apply programming to specific content areas such as biotech, and discussions of how to emphasize the connections among mathematics, web programming, multimedia and computer science. We believe we should improve this area as much as possible to help increase enrollments. We have continued the collaboration with the Math dept to use of the Latex typesetting language. Students now produce high-quality documentation for programs and even routine homework assignments. This is unique to our campus and something we are proud of! Some students have reported to us that their employers have noticed their skill in improved communication ability.

11. Discuss the impact of the program on the community and the impact of the community on the program.

Ohlone believes that the community has lots to offer us in improving our programs. One idea considered was to increase that contact through Service Learning projects. For example, it was suggested that we look for ways to have teams volunteer programming solutions to local nonprofit or small business to give students on-the-job experiences as part of a capstone project experience. It has been embedded into two sections of CSI70 class. Some progress has been made by adding a team project to 2 of our CS courses (CS102 and CS116). So far the projects are internal, not outreach, but the experience we gain working on these smaller projects can lead to confidence to reach out to the community in the future. One area that has been more fruitful recently has been participation in regular meetings at CSUEB in a joint program of local CC's and CSU's to improve transfer for Engineering students (not just Computer Engineering, but all Engineering fields of study). Another idea is to collaborate with Ohlone's WEX program that places students in work experience situations with local companies for credit.

● College Mission

1. Mission Statement

The mission of Ohlone College is to serve the community by offering instruction for basic skills, career entry, university transfer, economic development, and personal enrichment for all who can benefit from our instruction in an environment where student learning success is highly valued, supported and continually assessed.

2. Vision Statement

Ohlone College will be known throughout California for our inclusiveness, innovation and superior rates of student success.

3. Core Values, Goals & Objectives:

College Core Values

● We provide life-long learning opportunities for students, college personnel and the community.
● We open access to higher education and actively reach out to under-served populations.
● We promote diversity and inclusiveness.
● We maintain high standards in our constant pursuit of excellence.
● We value trust, respect and integrity.
● We promote team work and open communication.
● We practice innovation and actively encourage risk-taking and entrepreneurship.
● We demonstrate stewardship for our human, financial, physical and environmental resources.

College Goals/Objectives

1. Through innovative programs and services, improve student learning and achievement.
2. By 2013, have in place an ongoing system for identifying and assessing student learning outcomes at the program and course levels, which includes faculty dialogue and appropriate improvement plans.
3. By spring 2013, increase the college average course retention to a rate at or above the statewide average.
4. By spring 2013, increase to 600 the number of students transferring to UC and CSU.
5. By 2011, achieve 100% competition of professional development in online instructional methods and online course management for faculty who teach fully online or hybrid courses.
6. By 2012, establish mutual agreements with local school districts to redefine expectations of partnership in light of reduced staffing and budget support while maintaining clear pathways for students.
7. Support the economic vitality of the community through educational programs and services that respond to identified employment needs.
8. By 2013 provide opportunities across the curriculum for students to acquire key skill sets and concepts that will help them succeed in the workplace.
9. In the computer studies programs, "current" is a key driving force. Although our CS faculty keep the core concepts as our primary focus, we also must continually keep abreast of changes in programming languages, operating systems, and software tools. The CS faculty takes this very seriously and make sure that our students are aware of the changing nature of this field. We subscribe to academic alliances with major companies such as Oracle, IBM, and Microsoft to ensure access to the latest software development environment/tools and online training.

We practice innovation and actively encourage risk-taking and entrepreneurship. We promote diversity and inclusiveness by being sensitive to the different backgrounds that our students bring to the classroom. It is typical to have multiple cultures mixed together and communication is a challenge because the references are not shared history for everyone. We listen very carefully and consistently ask students if they understand and how they interpret what information has been given to them so we can find out how to better include everyone in the discussions and the learning process. We want our CS program to be known for its excellence. Our students should be recognized by CSU, UC, and the private sector as being very well prepared. We have listened to advice from the community by focusing more instruction toward communication skills. Technical skills are certainly the main focus, but we are adding requirements for our students to write about the technologies too. This develops higher skills on the Bloom's Taxonomy of Learning Domains. Not only can they do it, but they can discuss it, evaluate it, analyze it, etc.?"
5. Briefly describe how the program supports selected college goals.

Focus is on the individual; it is a college for and about the individual learner. Each CS faculty member works in the Hyman Hall student lab area directly tutoring students with their unique problems in understanding computer technology an average of 5 or more hours per week.

More options for students are better; experimentaion and innovation are encouraged and supported. We let students know that they can go beyond the minimum requirements for programming problems assigned and we help them to reach their goals. This is important because we get a broad range of students with a variety of backgrounds and preparation in the same classes. For example it is not uncommon to have a student fresh out of high school in the same class with a professional in the computer industry and another with a master’s degree in a field such as math or engineering.

Engagement in collaborative learning; learners are active, not passive vessels being filled. Our classes are taught in a facility where each student has their own computer used during class to try out the new techniques immediately after or interspersed with the lecture. This is beneficial because many people do not realize what questions to ask until they actually try to apply the techniques directly. Many classes already include group projects and effort is underway to add this component to several others (see the SLO matrix).

Everyone is a facilitator of student success. The CS faculty is lucky to have excellent support from lab assistants in Hyman Hall that devote themselves to helping students succeed. We all know that the students are our only reason for being here! We interact with IT staff on an almost daily basis, getting feedback, and furthering college goals as well as student goals by improving our knowledge about applications, infrastructure, and lab issues as we help them to prioritize work for maximum effectiveness. More collaboration can and should be done in this area.

As part of the Engineering Transfer partnership with CSU, we are finding ways to provide instruction that will consistently allow transfer students to perform in their junior year at the university at a level at or above that of students who started at the university as freshmen. Statements by instructors in those collaborative meetings indicate they (and we) are meeting that goal!

6. Briefly describe how the program supports selected college objectives.

This review process formalizes our work in studying ways to make our CS program better each year by observing students and the patterns of learning that show the greatest student success rates. In the area of sustainability and energy conservation we have discussed participating in studies to engage students in an understanding of using computers in a socially responsible way. We are using human, fiscal, technological, and physical resources responsibly, effectively, and efficiently to maximize student learning and achievement. On example of this is our willingness to schedule our office hours in the open lab in order to save expenses for the division in the need for staffing. We use computers beyond their normal lifespan and use freely available software rather then paying license fees for commercial software. We encourage students to choose topics for the team projects that explore socially relevant issues. For example, students have designed and written programs that were used by NASA in a space exploration project on a satellite.

Program SLOs & Assessment

1. Program SLO -

Given a specification, design an algorithm and implement the pseudocode to solve the problem.

a. Indicate program assessment strategies used.
   i. Rubrics
      ii. Culminating project
      iii. Performance Assessment
      iv. Skills Assessment
      v. Department Testing

b. Describe the criteria and standards used to appraise student work.

   We chose one test question out of three, which we hope will give meaningful data as to how SLO #1 is attained. The criteria for the questions follows: 1. The student will have to have understood the problem and pseudocode solution in order to complete one function out of three (the other two being already solved)
   2. It will be efficient for the instructor (full time or adjunct) to check the answer, because it is only 5/6 lines of code
   3. This question should work well in an online test of the type 'fill in the blank'
   4. There are 4 rubrics, with examples to guide the instructor with evaluation.

   c. Enter assessment results and analyze student success in achieving this program SLO.

   Data collected for scholastic years 09/10:

<table>
<thead>
<tr>
<th>Courses</th>
<th>No Clue</th>
<th>Some Proficiency</th>
<th>Close But No Cigar</th>
<th>Got It</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS102</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>CS102</td>
<td>7</td>
<td>6</td>
<td>14</td>
<td>11</td>
</tr>
<tr>
<td>CS102</td>
<td>14</td>
<td>1</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>CS175</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>CS102</td>
<td>6</td>
<td>5</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>CS102</td>
<td>12</td>
<td>5</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>CS170</td>
<td>5</td>
<td>1</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>Totals</td>
<td>51</td>
<td>24</td>
<td>34</td>
<td>60</td>
</tr>
</tbody>
</table>

   Based on the data, we have students on both sides of the spectrum, showing that either students get it or they don’t, with little in the middle. This is an expected trend in programming.

d. Describe revisions in curriculum or teaching strategies implemented to promote student success.

   The cs instructors collecting the data have become aware of the problem (data above) and are talking about ways to incorporate the concept of pseudocode within our programming exercises more than we have before.

   This year, in CS102, we have expanded the assessment of this skill to regular, weekly quizzes. This way, we can get feedback to the students more quickly and in time so that they (and we) can take corrective measures during the semester. The problem with assessing this skill on the final exam, is that it is then too late to fix gaps in the learning that were detected. So far, this approach has been successful. We will see the results on the final exam soon (watch this space for updates in spring!)
2. Program SLO -

Given a program with logic errors, correct the code by applying debugging and data validation skills.

   a. Indicate program assessment strategies used.
      i. Rubrics
      ii. Performance Assessment
      iii. Skills Assessment
      iv. Department Testing

   b. Describe the criteria and standards used to appraise student work.

   c. Enter assessment results and analyze student success in achieving this program SLO.

   d. Describe revisions in curriculum or teaching strategies implemented to promote student success.

3. Program SLO -

Demonstrate knowledge of fundamental computer science concepts (e.g. hardware, logic, discrete mathematics, software design, networks, and the internet.)

   a. Indicate program assessment strategies used.
      i. Rubrics
      ii. Performance Assessment
      iii. Department Testing

   b. Describe the criteria and standards used to appraise student work.

   c. Enter assessment results and analyze student success in achieving this program SLO.

   d. Describe revisions in curriculum or teaching strategies implemented to promote student success.

   e. Future Action (Improvements)

      Describe changes you will make to promote improved student learning

      We changed our intro courses (CS102 and CS116) to require all students to sign up for and attend a lab section (except 100% online students). This allows authentic, informal, assessments to take place while observing the students as they work on the labs. Besides that benefit, it also allows us to assign much more challenging labs since now they get immediate help from the instructor and are not be left to fend for themselves. We are continuing the idea of having the students write more literate, programs. This requires the students to write about the programs as well as code them for computers. It improves communication skills and forces them to think more critically about what they are doing.

      Implementation Plan
      Timeline:

      For the next few academic years, CS102 and CS116 will be taught in lecture/lab format by one instructor for consistency so we can determine if this new approach improves the student mastery of PSLO #1. All sections attend the same lecture in a room with no computers to help foster discussion and minimize distraction that each student having access to a computer affords. The labs will introduce new learning each week that complements the lectures.

      Key/Responsible Personnel:

      David Topham will teach the live lecture/lab sections.
Key/Responsible Personnel:
David Topham

SLO Matrix

Key: I-Introduced, P-Practiced with Feedback, M-Demonstrated at the Mastery Level

<table>
<thead>
<tr>
<th>Course</th>
<th>SLO-1</th>
<th>SLO-2</th>
<th>SLO-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIOT 131</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>BIOT 132</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>BIOT 141B</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>CNET 133</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>CNET 137</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>CS 101</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>CS 101L</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>CS 102</td>
<td>P</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>CS 104A</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>CS 104B</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>CS 113</td>
<td>I</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>CS 116</td>
<td>M</td>
<td>P</td>
<td>M</td>
</tr>
<tr>
<td>CS 118</td>
<td>P</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 124</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 125</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>CS 134</td>
<td>P</td>
<td>P</td>
<td>M</td>
</tr>
<tr>
<td>CS 145</td>
<td>P</td>
<td>P</td>
<td>M</td>
</tr>
<tr>
<td>CS 149</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>CS 152</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>CS 157</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>CS 170</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>CS 172</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 173</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CS 175</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>CS 178</td>
<td>P</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>MM 162</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
</tbody>
</table>

SLO Matrix Comments

Most of the courses in our dept are not in sequence. They can be taken in any order. Many do have prerequisite of CS102 or CS125, but other than that, students are free to choose.

This means there is no “capstone” course that represents the program as a whole and makes it more difficult to identify a cohort of students to assess for program SLO mastery.

Even the goal of "mastery" is misleading because these skills require many years to master. Similar to mastering a musical instrument, you can't do it in a year! It takes a lifetime.

But if we think of M as on the way to mastery, then I guess we use this as one of the measurements.

Course SLO & Assessment

CS 118 Introduction to Assembly Language Programming

1. Examine the decimal, binary and hexadecimal numbering systems. Convert values between each of the systems.
2. Demonstrate knowledge of functions and characteristics of an assembly language (such as MASM for Intel 80x86 and IA-32 Processor architecture)
3. Demonstrate knowledge of the use of registers, method of addressing, common instruction formats, stack processing, array and indirect processing.
4. Demonstrate the ability to program elementary programs in assembly language and provide appropriate documentation.
5. Evaluate basic boolean logic and analyze how it applies to programming and computer hardware.
6. Show how high-level languages translate statements into native machine code.
7. Compare ways that application programs communicate with the operating system using interrupt handlers, system calls, and common memory areas.
8. Survey CS concepts such as virtual machines, memory management, instruction execution, pipelining, cache, paging, and error-correction codes. Construct programs to demonstrate several of these concepts.

Indicate planned course assessment strategies

| Rubrics | Culminating Project | Performance Assessment | Skills Assessment | Indirect Assessment: Survey, Focus Group Discussion, Interview |

Describe the criteria and/or performance standards used to appraise student work.

Examine the decimal, binary and hexadecimal numbering systems. Convert values between each of the systems.

To examine these numbering systems, students will need to study the algorithms used to convert values between them.

When converting, they will be carrying out the instructions learned and demonstrated in class and applying them to new data as homework exercises. They will be checking the results of their efforts by using tools such as calculators and tables that compare values in the different number bases. To assess the mastery of these systems, we should ask the student questions that require successful application of the techniques. That means they must show the process that got them the answer and to use only paper and pencil rather than calculators or computers. Traditional paper quizzes will work well to assess this SLO.

Demonstrate knowledge of functions and characteristics of an assembly language (such as MASM for Intel 80x86 and IA-32 Processor architecture)
Students will be able to demonstrate this knowledge by planning and constructing computer programs that use the ASM instructions in the processor. Reading the textbook and listening to descriptions of the functions of the language enable to see how the functions work together. The lab exercises will allow them to experiment and observe the results. When the techniques are mastered, they will be able to construct the programs on their own. Reviewing the lab work and grading programming assignments will successfully assess this SLO.

Demonstrate knowledge of the use of registers, method of addressing, common instruction formats, stack processing, array and indirect processing. These skills are very specific to a real machines architecture rather than being general concepts. But still all machines must have some version of each of them. Therefore is is valuable to compare different approaches and allow the student to contrast and compare how different architectures implement the general ideas. Reading of the textbook could be complemented by internet research to understand that each machine will have different registers and methods of addressing and instruction formats, etc. It would be useful to have the student write a short paper comparing two different architectures and deciding the benefits of each. Assessment of this SLO could consist of verifying the accuracy of the facts in the paper and the students depth of understanding when describing the differences.

Demonstrate the ability to program elementary programs in assembly language and provide appropriate documentation. Documentation of assembly programs is especially important since there is no illusion that this low-level code can every be self-documenting. This is a good place to introduce the techniques of "Literate Programming" as championed by Donald Knuth since it would help students develop technical writing skills at the same time as programming skills. A good place to start is by showing by example how to write complete programs in this style then assigning independent projects as the semester progresses and enough samples have been shown so that students have some confidence in how to solve the problems by themselves. Assess this SLO using labs and programming assignments.

Evaluate basic boolean logic and analyze how it applies to programming and computer hardware. Logic is at the heart of computer science. It is used to design both hardware and software. Reading the textbook and working through many exercises will work well to understand the general concepts of boolean logic. Evaluating expressions and learning how to simplify them will help the student to gain insights into how valuable these techniques are. The best assessment is to assign graded homework (so it isn't optional) and to ask test questions that allow student to demonstrate they have done analysis of the ways that logic is applied to solve computing problems.

Show how high-level languages translate statements into native machine code. It is unlikely that many (or even any) of the students will become professional assembly language programmers. Yet it is still an important part of computer science education because it allows us to study the history of language development and to see how the layers of languages work together to allow the productive programming at higher levels only because the low-level coding in ASM is well understood. The best demonstration of this is to show the relationship between the typical high-level code used in other programming classes (such as Java or C++) is translated into native machine code. To assess this understanding, students should be assessed by test questions that require them to relate the two languages and programming assignments that require the translation of standard code to ASM code.

Compare ways that application programs communicate with the operating system using interrupt handlers, system calls, and common memory areas. "No man is an island" and no program stand alone either! The operating system is an essential ingredient to accessing the machine. Different architectures provide various ways to accomplish this and the students should study these options. Assessment of this SLO could be done by having the student write a short paper explaining a technique in depth and/or by assigning a program that requires interaction with the OS.

Survey CS concepts such as virtual machines, memory management, instruction execution, pipelining, cache, paging, and error-correction codes. As an introductory course, there is no way to cover any of these topics in depth. The best idea is to select a few and use them to explore interesting ideas. Programs can be assigned the use the scientific method: Propose theories, plan and carry out experiments, analyze the results and revise the theory as needed until general principles are formulated. Probably only one
of the above topics could be done this way in a single semester. Assessment could be of a semester-end project that allowed the student to take the initiative to explore the idea(s). If students can choose their own topic, then presentation to the rest of the class would be very interesting.

Enter assessment results and analyze student success in achieving course SLOs.

In fall 2011 semester, SLOs #4 and 6 were assessed:

4. Demonstrate the ability to program elementary programs in assembly language and provide appropriate documentation.

Since this can be assessed by analyzing the student labs, they were used to see how well students are mastering the skill to write programs. It was most informative to concentrate on the last 3 labs, since the earlier labs are guided so closely in a step-by-step manner in order to present the techniques and concepts. The students were much more on their own when solving the final labs. The analysis was focused on dividing the students into 3 ranges of skill: excellent, acceptable, and unsatisfactory.

Summary of labs 6-8 (15 attempts)

<table>
<thead>
<tr>
<th></th>
<th>Excellent (A)</th>
<th>Acceptable (B-C)</th>
<th>Unsatisfactory (D-F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>23 (51%)</td>
<td>18 (40%)</td>
<td>4 (9%)</td>
</tr>
</tbody>
</table>

This is encouraging news to see that such a high percentage of the students have mastered this skill!

6. Show how high-level languages translate statements into native machine code.

This SLO was assessed by including 3 questions in the final exam that specifically focused on this skill. By comparing the results of these questions only with the overall grades in the course, we can see if this SLO is weaker or stronger in general than other techniques being taught. The 3 questions chosen total to 9 points.

Summary of final exam questions related to translation from C to ASM

<table>
<thead>
<tr>
<th></th>
<th>Excellent (7-9 points)</th>
<th>Acceptable (4-6 points)</th>
<th>Unsatisfactory (0-3 points)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>8 (45%)</td>
<td>6 (33%)</td>
<td>4 (22%)</td>
</tr>
</tbody>
</table>

The overall grades in the course are similar percentages:

Excellent (45%), Acceptable (40%), Unsatisfactory (15%)

From that I believe that this SLO has enough focus compared to others.

Describe revisions in curriculum or teaching strategies implemented to promote student success.

This course is only offered once per year. Based on the assessments done this year, there is no need to change the approach for those SLOs the next time it is scheduled.

Future Action (Improvements)

Maintain current student learning plan

I believe the focus on literate programming (where students describe the programs as well as create and test them) was responsible for the success rate in the SLOs assessed, so this approach should be continued.

Implementation Plan

Timeline: The next time this course is offered (Fall 2012), 2 different SLOs will be chosen to assess. The goal is find potential weaknesses in the skill set being taught and to address those by finding new strategies.

Key/Responsible Personnel:

David Topham is teaching cs118 and will gather the data and assess the results with advice and guidance from other CS faculty.

CS 118 Introduction to Assembly Language Programming

1. Examine the decimal, binary and hexadecimal numbering systems. Convert values between each of the systems.
2. Demonstrate knowledge of functions and characteristics of an assembly language (such as MASM for Intel 80x86 and IA-32 Processor architecture)
3. Demonstrate knowledge of the use of registers, method of addressing, common instruction formats, stack processing, array and indirect processing.
4. Demonstrate the ability to program elementary programs in assembly language and provide appropriate documentation.
5. Evaluate basic boolean logic and analyze how it applies to programming and computer hardware.
6. Show how high-level languages translate statements into native machine code.
7. Compare ways that application programs communicate with the operating system using interrupt handlers, system calls, and common memory areas.
8. Survey CS concepts such as virtual machines, memory management, instruction execution, pipelining, cache, paging, and error-correction codes. Construct programs to demonstrate several of these concepts.

Indicate planned course assessment strategies

Rubrics
Culminating Project
Performance Assessment
Skills Assessment
Indirect Assessment: Survey, Focus Group Discussion, Interview

Describe the criteria and/or performance standards used to appraise student work.
Examine the decimal, binary and hexadecimal numbering systems. Convert values between each of the systems.

To examine these numbering systems, students will need to study the algorithms used to convert values between them. When converting, they will be carrying out the instructions learned and demonstrated in class and applying them to new data as homework exercises. They will be checking the results of their efforts by using tools such as calculators and tables that compare values in the different number bases. To assess the mastery of these systems, we should ask the student questions that require successful application of the techniques. That means they must show the process that got them the answer and to use only paper and pencil rather than calculators or computers. Traditional paper quizzes will work well to assess this SLO.

Demonstrate knowledge of functions and characteristics of an assembly language (such as MASM for Intel 80x86 and IA-32 Processor architecture).

Students will be able to demonstrate this knowledge by planning and constructing computer programs that use the ASM instructions in the processor. Reading the textbook and listening to descriptions of the functions of the language enable to see how the functions work together. The lab exercises will allow them to experiment and observe the results. When the techniques are mastered, they will be able to construct the programs on their own. Reviewing the lab work and grading programming assignments will successfully assess this SLO.

Demonstrate knowledge of the use of registers, method of addressing, common instruction formats, stack processing, array and indirect processing.

These skills are very specific to a real machines architecture rather than being general concepts. But still all machines must have some version of each of them. Therefore is it valuable to compare different approaches and allow the student to contrast and compare how different architectures implement the general ideas. Reading of the textbook could be complemented by internet research to understand that each machine will have different registers and methods of addressing and instruction formats, etc. It would be useful to have the student write a short paper comparing two different architectures and deciding the benefits of each. Assessment of this SLO could consist of verifying the accuracy of the facts in the paper and the students depth of understanding when describing the differences.

Demonstrate the ability to program elementary programs in assembly language and provide appropriate documentation.

Documentation of assembly programs is especially important since there is no illusion that this low-level code can every be self-documenting. This is a good place to introduce the techniques of "Literate Programming" as championed by Donald Knuth since it would help students develop technical writing skills at the same time as programming skills. A good place to start is by showing by example how to write complete programs in this style then assigning independent projects as the semester progresses and enough samples have been shown so that students have some confidence in how to solve the problems by themselves. Assess this SLO using labs and programming assignments.

Evaluate basic boolean logic and analyze how it applies to programming and computer hardware.

Logic is at the heart of computer science. It is used to design both hardware and software. Reading the textbook and working through many exercises will work well to understand the general concepts of boolean logic. Evaluating expressions and learning how to simplify them will help the student to gain insights into how valuable these techniques are. The best assessment is to assign graded homework (so it isn't optional) and to ask test questions that allow student to demonstrate they have done analysis of the ways that logic is applied to solve computing problems.

Show how high-level languages translate statements into native machine code.

It is unlikely that many (or even any) of the students will become professional assembly language programmers. Yet it is still an important part of computer science education because it allows us to study the history of language development and to see how the layers of languages work together to allow the productive programming at higher levels only because the low-level coding in ASM is well understood. The best demonstration of this to show the relationship between the typical high-level code used in other programming classes (such as Java or C++) is translated into native machine code. To assess this understanding, students should be assessed by test questions that require them to relate the two languages and programming assignments that require the translation of standard code to ASM code.
Compare ways that application programs communicate with the operating system using interrupt handlers, system calls, and common memory areas.

"No man is an island" and no program stand alone either! The operating system is an essential ingredient to accessing the machine. Different architectures provide various ways to accomplish this and the students should study these options. Assessment of this SLO could be done by having the student write a short paper explaining a technique in depth and/or by assigning a program that requires interaction with the OS.

Survey CS concepts such as virtual machines, memory management, instruction execution, pipelining, cache, paging, and error-correction codes.

As an introductory course, there is no way to cover any of these topics in depth. The best idea is to select a few and use them to explore interesting ideas. Programs can be assigned the use the scientific method: Propose theories, plan and carry out experiments, analyze the results and revise the theory as needed until general principles are formulated. Probably only one of the above topics could be done this way in a single semester. Assessment could be of a semester-end project that allowed the student to take the initiative to explore the idea(s). If students can choose their own topic, then presentation to the rest of the class would be very interesting.

Enter assessment results and analyze student success in achieving course SLOs.

In fall 2011 semester, SLOs #4 and 6 were assessed:

4. Demonstrate the ability to program elementary programs in assembly language and provide appropriate documentation.

Since this can be assessed by analyzing the student labs, they were used to see how well students are mastering the skill to write programs. It was most informative to concentrate on the last 3 labs, since the earlier labs are guided so closely in a step-by-step manner in order to present the techniques and concepts. The students were much more on their own when solving the final labs. The analysis was focused on dividing the students into 3 ranges of skill: excellent, acceptable, and unsatisfactory.

Summary of labs 6-8 (15 attempts)

<table>
<thead>
<tr>
<th>Excellent (A)</th>
<th>Acceptable (B-C)</th>
<th>Un satisfactory (D-F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>23 (51%)</td>
<td>18 (40%)</td>
<td>4 (9%)</td>
</tr>
</tbody>
</table>

This is encouraging news to see that such a high percentage of the students have mastered this skill!

6. Show how high-level languages translate statements into native machine code.

This SLO was assessed by including 3 questions in the final exam that specifically focused on this skill. By comparing the results of these questions only with the overall grades in the course, we can see if this SLO is weaker or stronger in general than other techniques being taught. The 3 questions chosen total to 9 points.

Summary of final exam questions related to translation from C to ASM

<table>
<thead>
<tr>
<th>Excellent (7-9 points)</th>
<th>Acceptable (4-6 points)</th>
<th>Un satisfactory (0-3 points)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 (45%)</td>
<td>6 (33%)</td>
<td>4 (22%)</td>
</tr>
</tbody>
</table>

The overall grades in the course are similar percentages:

Excellent (45%), Acceptable (40%), Unsatisfactory (15%)

From that I believe that this SLO has enough focus compared to others.

Describe revisions in curriculum or teaching strategies implemented to promote student success.

This course is only offered once per year. Based on the assessments done this year, there is no need to change the approach for those SLOs the next time it is scheduled.

Future Action (Improvements)

Maintain current student learning plan

I believe the focus on literate programming (where students describe the programs as well as create and test them) was responsible for the success rate in the SLOs assessed, so this approach should be continued.

Implementation Plan

Timeline:
The next time this course is offered (Fall 2012), 2 different SLOs will be chosen to assess. The goal is find potential weaknesses in the skill set being taught and to address those by finding new strategies.

Key/Responsible Personnel:

David Topham is teaching cs118 and will gather the data and assess the results with advice and guidance from other CS faculty.
CS 124 Programming With Data Structures

1. Apply a systematic approach to the design, construction and management of computer programs, emphasizing programming style, documentation, and debugging techniques.
2. Demonstrate knowledge of the basic data structures of stacks, lists, trees, graphs, queues, and sets.
3. Implement (program) these structures in their appropriate applications, such as sorting and searching.
4. Analyze and determine which of several methods involving data structures is most appropriate for solving a particular problem.
5. Evaluate the purpose and design of selected other data structures such as hash tables, heaps, etc.

Indicate planned course assessment strategies

<table>
<thead>
<tr>
<th>Rubrics</th>
<th>Performance Assessment</th>
<th>Skills Assessment</th>
<th>Department Testing</th>
</tr>
</thead>
</table>

Describe the criteria and/or performance standards used to appraise student work.

These are the learning objectives from the ACM guide cited above:

<table>
<thead>
<tr>
<th>ACM Learning Objectives</th>
<th>related ACM Learning Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td></td>
</tr>
<tr>
<td>8.</td>
<td></td>
</tr>
<tr>
<td>9.</td>
<td></td>
</tr>
</tbody>
</table>

CS124 SLOs

<table>
<thead>
<tr>
<th>Objective</th>
<th>related ACM Learning Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>5, 7, 9</td>
</tr>
<tr>
<td>2.</td>
<td>1, 5, 6, 7, 9</td>
</tr>
<tr>
<td>3.</td>
<td>5, 7</td>
</tr>
<tr>
<td>4.</td>
<td>2, 3, 4, 6, 8, 9</td>
</tr>
<tr>
<td>5.</td>
<td>6, 8, 9</td>
</tr>
</tbody>
</table>

Our task is to design assessments that determine how well students meet each of these objectives. For example, SLO #2 (Demonstrate knowledge of the basic data structures of stacks, lists, trees, graphs, queues, and sets.) can be realized if objectives 1, 5, 6, 7, and 9 are met.

This course assesses students using quizzes for each topic throughout the semester, along with a dozen programming labs to strengthen the understanding of the topics, then a comprehensive final exam to allow the students to demonstrate their knowledge.

Enter assessment results and analyze student success in achieving course SLOs.

Spring 2012: For this assessment I placed 10 multiple choice questions within the final exam to determine how well the students could differentiate between the various data structures. ACM objective 6 is particularly relevant to this knowledge since the students need to understand the differences among the structures in regards to their performance on the computer in order to compare them. The results were excellent for this: 22 out of 24 students scored very well on these questions.

In order to determine if the students could write about some of the structures in their own words rather than choosing answers listed, I put one question within the final asking them to compare two structures that are similar to see if they could isolate and explain the significant attributes needed to choose the right one for a particular situation. In this question asking them to compare a singly-linked list to a doubly-linked list, the performance was not as perfect, but still very good. Here 18/24 scored high, 4 showing some understanding but not enough, and 2 unable to find the important ideas at all.

Describe revisions in curriculum or teaching strategies implemented to promote student success.

This is the first assessment of this course, and since objective 6 was successful, I will plan additional assessments of other objectives the next time the course it taught (It is a spring-only offering currently).

Future Action (Improvements)

Maintain current student learning plan

Students have understood how to compare the performance of the important data structures, so no changes need to be made to the curriculum for this objective.

Describe changes you will make to promote improved student learning

In thinking about methods of assessment, I question if multiple choice is sufficient to measure knowledge. I would like to experiment with more skills-based demonstrations in the future. This may be more "authentic", but will require more time with students. I will request that the current "hybrid" lab be offered as a "live" lab when the budget permits.

Implementation Plan

Timeline:
Additional data will gathered in Spring 2013 to assess one or more of these SLOs

Key/Responsible Personnel:

Xisheng Fang is teaching CS124 and will gather data and analyze it with advice from other CS faculty.
CS 124 Programming With Data Structures

1. Apply a systematic approach to the design, construction and management of computer programs, emphasizing programming style, documentation, and debugging techniques.
2. Demonstrate knowledge of the basic data structures of stacks, lists, trees, graphs, queues, and sets.
3. Implement (program) these structures in their appropriate applications, such as sorting and searching.
4. Analyze and determine which of several methods involving data structures is most appropriate for solving a particular problem.
5. Evaluate the purpose and design of selected other data structures such as hash tables, heaps, etc.

Indicate planned course assessment strategies

Rubrics
Performance Assessment
Skills Assessment
Department Testing
Indirect Assessment: Survey, Focus Group Discussion, Interview

Describe the purpose and design of selected other data structures such as hash tables, heaps, etc.

This is the most advanced of the transfer programming courses. It closely follows the CSIII curriculum of the ACM.

This link has suggestions for assessing the skills that students as expected to develop in the course: http://www.acm.org/education/curricula-recommendations

These are the learning objectives from the ACM guide cited above:

ACM Learning Objectives

1. Describe the representation of numeric and character data.
2. Understand how precision and round-off can affect numeric calculations.
3. Discuss the use of primitive data types and built-in data structures.
4. Describe common applications for each data structure in the topic list.
5. Implement the user-defined data structures in a high-level language.
6. Compare alternative implementations of data structures with respect to performance.
7. Write programs that use each of the following data structures: arrays, strings, linked lists, stacks, queues, and hash tables.
8. Compare and contrast the costs and benefits of dynamic and static data structure implementations.
9. Choose the appropriate data structure for modeling a given problem.

CS124 SLOs related ACM Learning Objective

<table>
<thead>
<tr>
<th>Objective</th>
<th>Description</th>
<th>Related ACM Learning Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Apply a systematic approach to the design, construction and management of computer programs, emphasizing programming style, documentation, and debugging techniques.</td>
<td>5,7,9</td>
</tr>
<tr>
<td>2</td>
<td>Demonstrate knowledge of the basic data structures of stacks, lists, trees, graphs, queues, and sets.</td>
<td>1,5,6,7,9</td>
</tr>
<tr>
<td>3</td>
<td>Implement (program) these structures in their appropriate applications, such as sorting and searching.</td>
<td>5,7</td>
</tr>
<tr>
<td>4</td>
<td>Analyze and determine which of several methods involving data structures is most appropriate for solving a particular problem.</td>
<td>2,3,4,6,8,9</td>
</tr>
<tr>
<td>5</td>
<td>Evaluate the purpose and design of selected other data structures such as hash tables, heaps, etc.</td>
<td>6,8,9</td>
</tr>
</tbody>
</table>

Our task is to design assessments that determine how well students meet each of these objectives. For example, SLO #2 (Demonstrate knowledge of the basic data structures of stacks, lists, trees, graphs, queues, and sets.) can be realized if objectives 1,5,6,7, and 9 are met.

This course assesses students using quizzes for each topic throughout the semester, along with a dozen programming labs to strengthen the understanding of the topics, then a comprehensive final exam to allow the students to demonstrate their knowledge.

Enter assessment results and analyze student success in achieving course SLOs.

Spring 2012: For this assessment I placed 10 multiple choice questions within the final exam to determine how well the students could differentiate between the various data structures. ACM objective 6 is particularly relevant to this knowledge since the students need to understand the differences among the structures in regards to their performance on the computer in order to compare them. The results were excellent for this: 22 out of 24 students scored very well on these questions.

In order to determine if the students could write about some of the structures in their own words rather than choosing answers listed, I put one question within the final asking them to compare a singly-linked list to a doubly-linked list, the performance was not as perfect, but still very good. Here 18/24 scored high, 4 showing some understanding but not enough, and 2 unable to find the important ideas at all.

In order to determine if the students could write about some of the structures in their own words rather than choosing answers listed, I put one question within the final asking them to compare two structures that are similar to see if they could isolate and explain the significant attributes needed to choose the right one for a particular situation. In this question asking them to compare a singly-linked list to a doubly-linked list, the performance was not as perfect, but still very good. Here 18/24 scored high, 4 showing some understanding but not enough, and 2 unable to find the important ideas at all.

Describe revisions in curriculum or teaching strategies implemented to promote student success.

This is the first assessment of this course, and since objective 6 was successful, I will plan additional assessments of other objectives the next time the course is taught (the current offering is spring-only).

Future Action (Improvements)

Maintain current student learning plan
Students have understood how to compare the performance of the important data structures, so no changes need to be made to the curriculum for this objective.

Describe changes you will make to promote improved student learning
In thinking about methods of assessment, I question if multiple choice is sufficient to measure knowledge. I would like to experiment with more skills-based demonstrations in the future. This may be more "authentic", but will require more time with students. I will request that the current "hybrid" lab be offered as a "live" lab when the budget permits.

Implementation Plan
Timeline: Additional data will gathered in Spring 2013 to assess one or more of these SLOs

Key/Responsible Personnel:
Xisheng Fang is teaching CS124 and will gather data and analyze it with advice from other CS faculty.
CS 101 Introduction to Computers and Information Technology
1. Identify basic structures and organizations of computers, computer systems, computer networks, and computer information systems.
2. Define and use appropriate vocabulary pertaining to computers.
3. Explain how computer works.
4. Examine input, output, and storage devices.
5. Distinguish and recognize the components on the motherboard and the peripherals in a computer and computerized device.
6. Examine the hardware and software in a computer system.
7. Describe and categorize computer software and usages.
8. Interpret how the Internet, computer networks, and computer information systems are developed, organized and maintained.
9. Recognize and classify common used computer programming languages.
10. Develop a high degree of awareness of the implications of technology in today's society.
11. Criticize and analyze the safety, security, ethics and issues in the current use of the Internet and computer networks.

<table>
<thead>
<tr>
<th>Indicate planned course assessment strategies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rubrics</td>
</tr>
<tr>
<td>Culminating Project</td>
</tr>
<tr>
<td>Department Testing</td>
</tr>
<tr>
<td>Other</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Describe the criteria and/or performance standards used to appraise student work.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLOs 1, 2, 4, and 6 can be assessed using this rubric:</td>
</tr>
<tr>
<td>Approaches Goal: Identifies components of a computer system and recognizes the meanings of the vocabularies, and apply them to real world applications.</td>
</tr>
<tr>
<td>Meets Goal: Meets Goal: With some guidance provides correct answers to solve given problems in recognition of computer components and computer systems.</td>
</tr>
<tr>
<td>Surpasses Goal: Independently solves the problems in recognition of computer components and computer systems and provides solutions and/or suggestions.</td>
</tr>
<tr>
<td>SLO 3, 7, 8 and 9 can be assessed using this rubric:</td>
</tr>
<tr>
<td>Approaches Goal: Be able to use a searching engine in the Internet to investigate the current developments, trends and issues in computer and information technologies.</td>
</tr>
<tr>
<td>Meets Goal: Summarizes, outlines the current developments, trends and issues in computer and information technologies.</td>
</tr>
<tr>
<td>Surpasses Goal: Critiques, compares and contrasts, provides personal opinions in the investigations of the current developments, trends and issues in computer and information technologies.</td>
</tr>
<tr>
<td>SLO 5, 10 and 11 can be assessed using this rubric:</td>
</tr>
<tr>
<td>Approaches Goal: Produces project individually or in team-work that explores the history and future developments in computer science and computer applications.</td>
</tr>
<tr>
<td>Meets Goal: Produces a term paper that meets goals described in the SLOs.</td>
</tr>
<tr>
<td>Surpasses Goal: Successfully produces a comprehensive term paper with personal suggestions and criticism.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Enter assessment results and analyze student success in achieving course SLOs.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spring 2012: For the assessment I picked up the SLO #9: Recognize and classify common used computer programming languages. There were 10 out of 70 questions in the final exam dealing with this SLO. The result was good: 82% students answered these questions correctly and they could recognize and classify current commonly used computer programming languages in the test. In order to determine if students can master this SLO, Multiple choice questions dealing with all aspects about recognizing and classifying common used computer programming languages were given in the test. There were about 10 popular computer programming languages, including Java, C, C++, C#, .Net, Visual Studio, Scripting, and Report Generators covered in the questions. I consider these are not easy questions and students must have solid knowledge and understanding about computer science and technology to be able to answer these questions correctly. Percentage of Excellent Answers: 82% Percentage of Faire Answers: 10% Percentage of Incorrect Answers: 8%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Describe revisions in curriculum or teaching strategies implemented to promote student success.</th>
</tr>
</thead>
<tbody>
<tr>
<td>There is no revision in curriculum or teaching strategies necessary at this point.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Future Action (Improvements)</th>
</tr>
</thead>
</table>
Implementation Plan
Timeline:
Maintain current student learning plan.
Key/Responsible Personnel:
Yong Gao and Xisheng Fang are teaching CS101 and will gather data and analyze it with advice from other CS faculty.

CS 101 Introduction to Computers and Information Technology
1. Define and use appropriate vocabulary pertaining to computers.
2. Explain how computer and system work.
3. Examine the hardware and software in a computer system.
4. Describe and categorize computer software and usages.
5. Interpret how the Internet, computer networks, and computer information systems are developed, organized and maintained.
6. Recognize and classify common used computer programming languages.
7. Develop a high degree of awareness of the implications of technology in today’s society.

Indicate planned course assessment strategies

<table>
<thead>
<tr>
<th>Rubrics</th>
<th>Culminating Project</th>
<th>Department Testing</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Describe the criteria and/or performance standards used to appraise student work.

SLOs 1, 2, 4, and 6 can be assessed using this rubric:
- **Approaches Goal:** Identifies components of a computer system and recognizes the meanings of the vocabularies, and apply them to real world applications.
- **Meets Goal:** Meets Goal: With some guidance provides correct answers to solve given problems in recognition of computer components and computer systems.
- **Surpasses Goal:** Independently solves the problems in recognition of computer components and computer systems and provides solutions and/or suggestions.

SLO 3, 7, 8 and 9 can be assessed using this rubric:
- **Approaches Goal:** Be able to use a searching engine in the Internet to investigate the current developments, trends and issues in computer and information technologies.
- **Meets Goal:** Summarizes, outlines the current developments, trends and issues in computer and information technologies.
- **Surpasses Goal:** Critiques, compares and contrasts, provides personal opinions in the investigations of the current developments, trends and issues in computer and information technologies.

SLO 5, 10 and 11 can be assessed using this rubric:
- **Approaches Goal:** Produces project individually or in team-work that explores the history and future developments in computer science and computer applications.
- **Meets Goal:** Produces a term paper that meets goals described in the SLOs.
- **Surpasses Goal:** Successfully produces a comprehensive term paper with personal suggestions and criticism.

Enter assessment results and analyze student success in achieving course SLOs.

**Spring 2012:** For the assessment I picked up the SLO #9: Recognize and classify common used computer programming languages. There were 10 out of 70 questions in the final exam dealing with this SLO. The result was good: 82% students answered these questions correctly and they could recognize and classify current commonly used computer programming languages in the test. In order to determine if students can master this SLO, Multiple choice questions dealing with all aspects about recognizing and classifying common used computer programming languages were given in the test. There were about 10 popular computer programming languages, including Java, C, C++, C#, .Net, Visual Studio, Scripting, and Report Generators covered in the questions. I consider these are not easy questions and students must have solid knowledge and understanding about computer science and technology to be able to answer these questions correctly.
- Percentage of Excellent Answers: 82%
- Percentage of Faire Answers: 10%
- Percentage of Incorrect Answers: 8%

**Fall 2012:** For the assessment I picked up the SLO #3: Examine the hardware and software in a computer system. There were 10 out 40 questions in the Quiz dealing with this SLO. The result was excellent: 91% students answered these questions correctly and they could exam and recognize hardware and software in a computer system in the exam. In order to
determine if students can master this SLO. Multiple choice questions dealing with all aspects about recognizing and classifying common hardware and software used in a computer system were given in the test. I consider these are relatively easy or common questions and students should answer them correctly if they follow the instruction and study in the assigned homework.

Percentage of Excellent Answers: 92%
Percentage of Faire Answers: 7%
Percentage of Incorrect Answers: 1%

Describe revisions in curriculum or teaching strategies implemented to promote student success.

There is no revision in curriculum and teaching strategies necessary at this point. SLOs are modified in current major revision to be better reflect student learning outcomes and teaching.

Future Action (Improvements)

Implementation Plan
Timeline:
Maintain current student learning plan.
Key/Responsible Personnel:
Yong Gao and Xisheng Fang are teaching CS101 and will gather data and analyze it with advice from other CS faculty.

CS 116 Object-Oriented Programming using C++

1. Compare and contrast object-oriented programming with procedural programming
2. Practice creating modules using encapsulation, information hiding, inheritance, and polymorphism
3. Recognize the concept of types as a set of values together with a set of operations
4. Design and construct Exception handling methods
5. Create Linked Structures using pointers
6. Formulate procedures, functions, and iterators as abstraction mechanisms
7. Construct parameterized types (i.e. class templates in C++)
8. Propose and evaluate the separation of specification and implementation

Indicate planned course assessment strategies

Culminating Project
Skills Assessment
Indirect Assessment: Survey, Focus Group Discussion, Interview

Describe the criteria and/or performance standards used to appraise student work.

Fall 2012: SLOs 2 and 3 are a good place to start in assessing this course since they represent the most basic understanding of object-oriented programming (OOP) techniques. The level of skill mastered by the students would be evident by evaluating their lab work. The first labs will be guided in detail using literate programming (where students write about code in the same document as the code). Then, the students will be expected to determine for themselves where to apply the correct techniques. The last weeks of the course will be team projects which will help students learn from each other about the best ways to organize the programs.

Enter assessment results and analyze student success in achieving course SLOs.

...in progress...data to be gathered at end of fall semester and entering here in spring...

Describe revisions in curriculum or teaching strategies implemented to promote student success.

The strategy of using live labs already implemented based on results from CS102. We are anxious to see if better results come from introducing the labs in CS116.

Future Action (Improvements)

Describe changes you will make to promote improved student learning

No data yet, but base on ongoing observations, we still need to find a way to lower the drop out rate of the course. For some reason, 50% of students drop this class every semester by the halfway point! No other class has that much fans. Even with the live labs this semester, the same thing as happened again. It seems that the material is more difficult than students expect, and they give up too soon. New ideas are needed and are being researched...

CS 116 Object-Oriented Programming using C++

1. Compare and contrast object-oriented programming with procedural programming
2. Practice creating modules using encapsulation, information hiding, inheritance, and polymorphism
3. Recognize the concept of types as a set of values together with a set of operations
4. Design and construct Exception handling methods
5. Create Linked Structures using pointers
6. Formulate procedures, functions, and iterators as abstraction mechanisms
7. Construct parameterized types (i.e. class templates in C++)
8. Propose and evaluate the separation of specification and implementation

Indicate planned course assessment strategies

Culminating Project
Skills Assessment
Indirect Assessment: Survey, Focus Group Discussion, Interview
### Describe the criteria and/or performance standards used to appraise student work.

Fall 2012: SLOs 2 and 3 are a good place to start in assessing this course since they represent the most basic understanding of object-oriented programming (OOP) techniques. The level of skill mastered by the students would be evident by evaluating their lab work. The first labs will be guided in detail using literate programming (where students write about code in the same document as the code). Then, the students will be expected to determine for themselves where to apply the correct techniques. The last weeks of the course will be team projects which will help students learn from each other about the best ways to organize the programs.

Enter assessment results and analyze student success in achieving course SLOs.

...in progress...data to be gathered at end of fall semester and entering here in spring...

Describe revisions in curriculum or teaching strategies implemented to promote student success.

The strategy of using live labs already implemented based on results from CS102. We are anxious to see if better results come from introducing the labs in CS116.

**Future Action (Improvements)**

Describe changes you will make to promote improved student learning.

No data yet, but based on ongoing observations, we still need to find a way to lower the drop out rate of the course. For some reason, 50% of students drop this class every semester by the halfway point! No other class has that much loss. Even with the live labs this semester, the same thing as happened again. It seems that the material is more difficult than students expect, and they give up too soon. New ideas are needed and are being researched...

---

### Indicate planned course assessment strategies

#### Department Testing

Describe the criteria and/or performance standards used to appraise student work.

The ACM published guidelines are useful in selecting criteria for assessment of computer science topics. For SLO #2:

1. list, describe, and apply the mathematical tools used to examine the theoretical foundations of computer science.
2. construct valid mathematical arguments using logical connectives and quantifiers and verify the correctness of a mathematical arguments using symbolic logic and truth tables.
3. use discrete math and logic to specify computer applications and reason about programs in a systematic way.

Enter assessment results and analyze student success in achieving course SLOs.

Preliminary data has been gathered in the most recent quiz on inductive proofs. Rather than multiple choice, the students had to demonstrate skill with constructing a proof. These results will help form the question(s) to be placed on the final so we can see the success rate for the semester.

Describe revisions in curriculum or teaching strategies implemented to promote student success.

Data is being gathered this semester, results entered here in spring...

**Future Action (Improvements)**

CS 113 Discrete Mathematics for Computers

1. list, describe, and apply the mathematical tools used to examine the theoretical foundations of computer science.
2. construct valid mathematical arguments using logical connectives and quantifiers and verify the correctness of a mathematical arguments using symbolic logic and truth tables.
3. use discrete math and logic to specify computer applications and reason about programs in a systematic way.

#### Indicate planned course assessment strategies

#### Department Testing

Describe the criteria and/or performance standards used to appraise student work.

The ACM published guidelines are useful in selecting criteria for assessment of computer science topics. For SLO #2:

2. construct valid mathematical arguments using logical connectives and quantifiers and verify the correctness of a mathematical arguments using symbolic logic and truth tables.

We will use selected questions from the final exam to analyze results "horizontally". That is, not the total scores of students, but success on particular questions. The idea is to find weak spots in the knowledge demonstrated by students in a more fine-grained way than we would see looking at total scores.

Enter assessment results and analyze student success in achieving course SLOs.

Preliminary data has been gathered in the most recent quiz on inductive proofs. Rather than multiple choice, the students had to demonstrate skill with constructing a proof. These results will help form the question(s) to be placed on the final so we can see the success rate for the semester.

Describe revisions in curriculum or teaching strategies implemented to promote student success.

Data is being gathered this semester, results entered here in spring...

**Future Action (Improvements)**

CS 113 Discrete Mathematics for Computers

1. list, describe, and apply the mathematical tools used to examine the theoretical foundations of computer science.
2. construct valid mathematical arguments using logical connectives and quantifiers and verify the correctness of a mathematical arguments using symbolic logic and truth tables.
3. use discrete math and logic to specify computer applications and reason about programs in a systematic way.
Program Improvement Objectives:

1. Program Analysis

   a. Course completion rates
   b. Course retention rates
   c. Program completion rates

   - Preliminary data has been gathered in the most recent quiz on inductive proofs. Rather than multiple choice, the students had to demonstrate skill with constructing a proof. These results will help form the question(s) to be placed on the final so we can see the success rate for the semester.
   - Describe revisions in curriculum or teaching strategies implemented to promote student success.
   - Data is being gathered this semester, results entered here in spring.

2. Future Action (Improvements)

   - Promote student success.
   - Describe revisions in curriculum or teaching strategies implemented to promote student success.
   - New systems for two designated rooms and lab
   - Year 1:
     - IT personnel reassigned under Computer Science Dean for full time support to bottom floor Hyman Hall
     - Maintenance and upgrade of overhead projectors in each classroom and teaching related supplies
   - Year 2:
     - 84 new systems for two designated rooms and lab

   - Student Achievement: A series of measures including course completion, course retention, persistence, program completion, and others.

   - List expected student achievement outcomes:
     - Apply for an appropriate Ohlone certificate or AS degrees.
     - Increase successful preparation and transfer rate to 4-yr schools.
     - Increase skills that prepare students for employment.
     - Analyze changes in data, identify trends, and provide possible contextual explanations for each measure used. (Example measures include: course completion, course retention, persistence, program completion).
     - Based on the previous review survey (see attached file) we recognize the need to increase the number of students who apply for certificates and degrees. We gathered data from Datatel showing CS course completion. We plan to use these data to identify the success of CS students and encourage them to apply for certificates and/or degrees, which they have almost or already earned.
     - This idea did not come to fruition due to lack of time to do the research and implementation. I believe to be effective with this task, we need programming support from IT/Datatel to automatically identify potential students to apply for degrees and certificates using knowledge of which course they have completed and comparing that to those needed for the degrees. Faculty do not have access to that data.
     - Analyze program budget trends and expenditures. Comment on how the program can best use budget resources.
     - Looking at the updated statistics in link above (Instructional Summary Data by Department) is somewhat alarming. It shows that spring 2010 was the lowest success rate so far! It is not easy see why that is the case, since most categories of students show improved success rates, yet overall the rate has dropped (to 58% from usually well over 60%). I think our CS students need more help in lab than they are getting. Due to budget cuts, we have trimmed the lab assistant scheduled time to less than 1/2 of what it used to be. It is possible that this is a reason why the success rate is lower. Of course, instructional effectiveness may also be a factor.
     - Analyze the program's current use of staff, equipment, technology, facilities, and/or other resources. Comment on how the program can best use these resources.
     - We should find ways to increase the hours that lab has skilled assistance available to students. Instead of just convering the desk for inquiries, we need hands on help. Perhaps alternate funding could be found to increase that budget?
     - Describe any additional notable program achievements(optional).
     - Additional Program Table Data
     - Current levels of student achievement indicators maintained.

   - Program Analysis

     - After assessing student learning outcomes/impacts, student/program achievement, and the status of previous program improvement objectives (PIOs), analyze the data and any identified trends, and summarize your findings. Use these data and trends to prioritize, revise, or develop new PIOs

     - We have assessed the first of our SLOs and begun the process of analyzing the success rates. Results are discouraging so far since we have an upside down bell curve with a split between students that are successful and those not so successful. There is only a very small group in the middle. We are rethinking the best way to increase the success rate of this middle group and to be sure we are effectively teaching the skills needed rather than just assessing the natural talents of students.

     - One possible reason for this is the huge variety of background experience coming into the program. Many students have already been exposed to computer programming thorough work or hobby and are here to learn more. In a sense, they are already successful and here to grow. Another group has no idea of what to expect and tend to be shy and quiet about asking for help. Many of these do very poorly. We should explore uncovering the weaknesses sooner and try not to teach to the most vocal in the group.

     - I am looking at constructivist as one possible approach. In my lecture courses, I am stopping every few minutes and posing questions to students (randomly to force shy people to participate and not let the aggressive, confident people monopolize the discussions). I am discovering that concepts that I thought were explained, were not understood at all! For me, the highest priority PIO is to address that weakness. Fortunately that doesn't require a big budget; it just relies on honest, real-time assessment of the reality in the room.

   - Program Improvement Objectives:

     1. Objective:

        - Keep up with the current technology needed to achieve the computer science program SLO's.

        a. Action Plan

           Year 1:

           - Get input from the industry as to current and future platforms skills needed for entry level positions, identify and improve hardware and software platforms in need of replacement or upgrade.

        b. Staffing

           Year 1:

           - IT personnel reassigned under Computer Science Dean for full time support to bottom floor Hyman Hall

        c. Equipment (Include items that fit under department budget codes)

           Year 1:

           - Maintenance and upgrade of overhead projectors in each classroom and teaching related supplies

        d. Technology (Include items that fit under IT budget codes)

           Year 1:

           - 84 new systems for two designated rooms and lab
Year 2:
84 new systems for other designated rooms and lab
Year 3:
84 new systems for remaining high-priority rooms and lab
e. Facilities (Include items that fit under the Facilities budget codes)
Year 1:
Install and support multiple OS platforms in the HH119
f. Other (Include other resources needed)
Year 1:
Involve Computer Science faculty and staff with IT in the decision making of equipment purchases
g. Assessment Plan: List Assessment Strategies
Year 1:
Record of CS meeting evaluating the progress
Year 2:
Record of CS meeting evaluating the progress
Year 3:
Record of CS meeting evaluating the progress

h. Which college goal(s) does this program improvement objective work to achieve? Clearly describe how your PIO will help achieve one or more of the college goals and objectives, has impact beyond the particular department, and contributes to student learning/success.
1. Through innovative programs and services, improve student learning and achievement.
Rationale:

2. PIO Assessment
a. Enter assessment results with analysis.

New computers were installed in HH112 & HH117 resulting in big improvements toward student satisfaction.

Another PIO was to add live-labs to our courses rather than rely on the hybrid approach that we had adopted in the past. By combining sections into one lecture, we offset some of the increased cost of the lab section.
b. Describe how PIO achieved one or more of the college goals and objectives, had an impact beyond the particular department, and contributed to student success/learning.

The new computers installed in HH112 & HH117 are used by other classes besides computer studies and have improved access to current software and saved time since the computers load and run programs faster.

A benefit of the lecture change is we needed to use a larger room which does not have computers on each students desk. This turns out to be a plus because in lab environments, it is too easy for students to tune-out whenever they wish by using the computers to entertain themselves rather than participate in the discussion. (This doesn't happen during lab sessions since then they have activities that need to consume their attention.) This doesn't impact other departments, but does enhance the college goals of increasing student success rates.
c. Analyze the impact of reallocation or addition of resources. If money or resource was not used, give rationale.

Due to the SERP (early retirement), CS has lost one of our most valuable players. This will make progress more difficult and we should plan on filling that vacant position as soon as the budget permits. The new instructor needed should focus on development of web programming.
d. Future Action

Strategies to promote improvements. Specify.

There is no possibility right now of hiring a new faculty to cover this gap, so we will need to rethink how we can keep this part of our program growing and developing rather than slipping into stagnation. I am most optimistic that we somehow use the talents and energy of current students as lab assistants to augment the instruction and provide needed help for those struggling to keep up.

● Outside Review Results
1. List each team members name and title.

This department has no official advising team; however we meet regularly with faculty from UC and CSU in order to align our curriculum with those of the 4-yr schools. For example, monthly meetings with CSUEB, SFSU, and SJSU are taking place to improve transfer for Computer Engineering students.

Several years ago, we split computer studies into its component parts (CAOT, CNET, CS) in order to focus on our different goals (e.g. career-skills development vs transfer); however, we recently have recombined informally to benefit from the advising role and outside review we can each provide each other by holding monthly meetings together to share information.

2. Discuss key feedback provided by team and how it was incorporated into the report.

We have developed 2-yr road maps for students to maximize their ability to enter the local 4-yr schools with junior standing. Feedback from CSU is that students are very well prepared by our program with success rates even higher than students that enter CSU as freshman. Here is the result of our efforts: Engineering Pathways

The monthly computer studies meetings are playing an important function towards program review skill development and assessment/implementation techniques. At our most recent meeting, the chair of the curriculum committee attended and helped us in kicking off our 6-yr course review with confidence. It has also given us an opportunity to work with our new dean in a relaxed, productive way.

● Attached Files