A. ACTIONS TAKEN TO IMPROVE STUDENT LEARNING

The 2012-2013 academic year saw a complete redesign of all three computer science & engineering programs due to the university-wide transition to semester-based terms. As no assessment data exists for the new semester-based courses or programs of study, assessment efforts during the 2012-2013 cycle have been largely focused on the development of direct assessment instruments and collection of data for the newly offered programs/courses.

Three primary initiatives have been taken to improve student learning during this cycle:

- Delivery of inverted-lecture core sequence (SCALE-UP Classrooms - 152 RC & 355 RC)
- Development of program educational objectives with program constituents
- Preliminary development of infrastructure for continuous assessment of relevant retained knowledge

B. STUDENT LEARNING OUTCOMES ASSESSED AND EXAMINED

Program Educational Objectives (BACS) [DRAFT – Final version pending approval]

- EXPERT: Graduates of the Computer Science program are employable as computing professionals and will be recognized by their employers as well-prepared for their career in computing
- AGILE: Graduates understand that education is a lifelong process and are well prepared for continuing studies.
- ENGAGED: Graduates demonstrate appreciation for the professional, social, ethical and leadership roles of computing professionals.
- APPLIED: Graduates can apply computing and software development principles to a diverse range of domains, such as analytics, data science, informatics, management, etc.

Student Learning Outcomes (BSCA) [Draft – Final version pending approval]

Students who complete the BA in Computer Science will have:

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
c. An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs
d. An ability to function effectively on teams to accomplish a common goal
e. An understanding of professional, ethical, legal, security and social issues and responsibilities
f. An ability to communicate effectively with a range of audiences
g. An ability to analyze the local and global impact of computing on individuals, organizations, and society
h. Recognition of the need for and an ability to engage in continuing professional development
i. An ability to use current techniques, skills, and tools necessary for computing practice
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.
k. An ability to apply design and development principles in the construction of software systems of varying complexity.

Once the assessment infrastructure is complete, the computer science department expects to collect and examine assessment data for every program SLOs every term of every year. SLOs are mapped to specific knowledge topics developed in specific core courses. These knowledge topics are assessed in subsequent courses. These direct assessments form the
examination basis for program SLOs. The initial version of this data collection for this new process was deployed Fall 2012 (as part of the semester curriculum conversion). Development has continued throughout 2013. SLO coverage should be complete sometime in 2014. First analysis of the initial assessment data is presented in the supporting documents.

C. METHODS FOR COLLECTING DATA
Over the past decade, engineering programs nation-wide have devoted significant effort towards measuring educational objectives in the style recommend by ABET Engineering Curriculum 2000 to monitor and improve program effectiveness. In many cases the collection and interpretation of this data has taken place in a labor intensive ad hoc fashion which limits utility of the collected data to drive curricular or pedagogic improvement. We have deployed a data collection infrastructure designed to measure success in retaining specific knowledge area topics deemed critical by our discipline’s professional societies. Where possible, assessment points are deployed at that start of courses that use knowledge topics developed in prerequisite core courses. This infrastructure allows evaluation of retention of expertise, allows assessment of differences in outcomes between learning pathways, and is less subject to instructor, course format, or other bias. Equally important, this infrastructure requires minimal resources post-deployment yet collects critical program data while also providing immediate feedback to students and course instructors regarding the preparation of students as they enter courses that allows for focused review and reinforcement of knowledge areas not retained due to variation in preparation.

We have worked with our program faculty to produce a mapping Knowledge Topics prerequisite to or developed in each of the core/mandatory courses in our program. Our initial assessment framework is limited to mandatory “core” courses. For each course, the faculty has indicated what knowledge topics are developed or assumed (pre-requisite) in the semester-based course offerings. These Knowledge Topics are mapped to one or more SLOs.

We attempt to take a direct assessment of each knowledge topic is made for every student enrolled in every core course in the program every term. These direct assessments take place not in the course that develops the knowledge topic but, when possible, at the beginning of a subsequent course (or courses) that utilize(s) and build(s) on that knowledge topic.

D. ASSESSMENT MEASURES
For each course, the faculty has indicated what knowledge topics are developed or assumed (prerequisite) in the semester-based course offerings. We perform a direct assessment of each knowledge topic not only in the course that develops that knowledge, but when possible, at the beginning of a subsequent course (or courses) that utilizes and builds on the topic. Summative grading rubrics are, when possible, deployed at the start of the next course in the core course. These assessment points allow better evaluation of the retention of expertise as measured prerequisite knowledge coming into each course. Assessment of prerequisite knowledge also allows assessment of differences among learning pathways, and are less subject to instructor-related or course-related bias.

Indirect assessments are obtained from two formal groups, the department’s external advisory board and the department’s student advisory board. The external advisory board consists of alumni, employers, and other professionals familiar with our discipline, program, and students. The department’s student advisory board represents a sample of our undergraduate students, including freshman through seniors, both honors and non-honors students, both traditional and non-traditional students, and both minority and non-minority students.

E. SIGNIFICANT FINDINGS
Given the paucity of historical data under the new semester-based program (delivered for the first time during the 2012-2013 academic year), no findings based on our assessment data can be considered significant. Some potential concerns have been noted and flagged for long-term observation. Initial findings are provided in supporting materials.

F. DISCUSSION OF RESULTS
The results of assessment are shared each term with the department’s undergraduate curriculum committee. Assessments that lead to program/course modification are shared with the department faculty when such changes are proposed for formal action and faculty vote.

G. ACTIONS PLANNED TO IMPROVE STUDENT LEARNING
Development of the new semester-term based infrastructure for assessing retained relevant knowledge continues. A meeting of the department’s undergraduate curriculum committee is scheduled early in Spring term 2014 to discuss the assessments of knowledge developed in Fall 2013 offerings. No specific data-driven actions are planned as a result of the initial data available as of Fall 2013.
H. SUPPORTING DOCUMENTS (recommended)

- Curriculum Assessment document
- Presentation of initial assessment data (from courses taken Fall 2012 and Spring 2013)
- Excerpts from the minutes of the undergraduate studies committee (UGSC)
Infrastructure for continuous assessment of retained relevant knowledge

Travis Doom, Kathleen Timmerman, Michael Raymer
Wright State University, Dayton, OH 45431-0001
Email: travis.doom@wright.edu

Abstract
Over the past decade, engineering programs nation-wide have devoted significant effort towards measuring educational objectives in the style recommend by ABET Engineering Curriculum 2000 to monitor and improve program effectiveness. In many cases the collection and interpretation of this data has taken place in a labor intensive ad hoc fashion which limits utility of the collected data to drive curricular or pedagogic improvement. Herein, we present a data collection infrastructure designed to measure success in retaining specific knowledge area topics deemed critical by each engineering discipline’s professional society. Where possible, assessment points are deployed at that start of courses that use knowledge topics developed in prerequisite core courses. This infrastructure allows evaluation of retention of expertise, allows assessment of differences in outcomes between learning pathways, and is less subject to instructor, course format, or other bias. Equally important, this infrastructure requires minimal resources post-deployment yet collects critical program data while also providing immediate feedback to students and course instructors regarding the preparation of students as they enter courses that allows for focused review and reinforcement of knowledge areas not retained due to variation in preparation.

Introduction
Institutions of higher learning world-wide have embraced continuous improvement models to measure and increase effectiveness of student learning. This is particularly true in undergraduate engineering programs as the ABET 2000 criteria prompted engineering departments to adopt continuous program outcome assessment to satisfy basic level accreditation criteria. All ABET accredited engineering programs are now expected to have some model for continuous program outcome assessment in place. The key to continuous improvement is an effective assessment program. Without a solid measure of student learning, a cycle of improvement is driven by the variations and vagaries of the data and is less likely to result in meaningful positive change.

Learning is multidimensional and requires multiple methods of collection in order to produce meaningful data. Direct methods of assessment measure student performance against some rubric of success. Indirect methods of assessment more often measure the student’s (or observer’s) perception of attainment. While both methods of assessment have their place, direct measures of assessment have been used for decades to provide a means for quality assurance. Historically, direct examinations such as the ACT and SAT have been used to measure the educational achievement of high-school students applying to college. Similarly, examinations such as the GRE, subject GRE, and Fundamentals of Engineering (FE) examination have been used to measure student educational achievement in University and to partially gauge professional competency.
Examinations of this sort provide validation against a set of external criteria that demonstrate that the retained knowledge of each student is relevant to the current national standard. Unfortunately, end-of-program examinations of this sort make poor tools for continuous program improvement. It is difficult, if not impossible, to provide a linkage between overall examination performance and specific actions or pedagogies employed in the educational process that led to greater or lesser success.

Continuous periodic direct measurements provide the best opportunity for measuring the performance effects of specific changes to programs, courses, and pedagogies. However, such data collection efforts are practically limited due to the sometimes massive effort required from administration, faculty, and students.

We propose here an infrastructure to assess program effectiveness with the following goals:

1. The assessment provides continuous periodic direct measurements of retained relevant knowledge.
2. The assessment outcome is immediately valuable to the assessment participants (students and faculty) as well as the continuous improvement of the program.
3. The assessment is not unduly burdensome.

**Assessment knowledge topics**

The goal of assessment is to provide data to measure (or illustrate a need for) improvement. The definition of the assessment standards then set a target goal towards which a program continuously strives to better meet. Although program objectives differ significantly among institutions, certain knowledge and skills are expected of graduates of engineering programs. We believe that the standard towards which programs should strive in Engineering is best communicated not only by the accreditation agencies but also by the appropriate discipline-specific international professional society. These societies maintain and regularly update the themes, knowledge areas, and professional practices expected of those entering their discipline.

For example, in computer science, the Joint Task Force on Computing Curricula between the Association for Computing Machinery (ACM) and IEEE-Computer Society provides regularly updated standards in curriculum, most recently in the volume Computer Science Curricula 2013 (CS2013) [1]. The CS2013 Body of Knowledge organizes the expectations of Computing graduates into 18 Knowledge Areas (KA) which are created, revised, and removed as the discipline changes over time (Figure 1, below). Each of these KAs is further specified as a set of Knowledge Units (Figure 2, below) each of which specifies a set of Knowledge Topics (Figure 3, below) expected at the time of graduation.

<table>
<thead>
<tr>
<th>AL</th>
<th>Algorithms and Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR</td>
<td>Architecture and Organization</td>
</tr>
<tr>
<td>CN</td>
<td>Computational Science</td>
</tr>
<tr>
<td>DS</td>
<td>Discrete Structures</td>
</tr>
<tr>
<td>GV</td>
<td>Graphics and Visualization</td>
</tr>
<tr>
<td>HC</td>
<td>Human Computer Interaction</td>
</tr>
<tr>
<td>IAS</td>
<td>Information Assurance</td>
</tr>
<tr>
<td>IM</td>
<td>Information Management</td>
</tr>
<tr>
<td>IS</td>
<td>Intelligent Systems</td>
</tr>
</tbody>
</table>
For computer science programs, CS2013 can serve as a “gold standard” for contemporary computing education. The professional societies of other engineering disciplines provide similar international curricular standards along with, in many cases, examinations which new graduates are expected to pass in order to be fully qualified to work in the discipline. In recognition that program objectives differ, CS2013 identifies topics as being either core tier-1 (required knowledge for every student in every program), core tier-2 (generally essential topic for which the vast majority should be covered but which may differ by student or program), or elective. CS2013 makes the categorizations by the process of “widespread consensus for inclusion” and further notes that “at least a preliminary treatment of most of these [core tier-1] topics typically comes in the first two years”. The explicitly stated coverage target for core tier-2 topics is “90-100% for every student, with 80% [as measured in lecture hours] considered as a minimum”.

*Proceedings of the 2013 ASEE North-Central Section Conference*
*Copyright © 2013, American Society for Engineering Education*
CS2013 contains 163 lecture hours of Core Tier-1 material and 142 lecture hours of Core Tier-2 material. For comparison, assume that semester-based course has roughly 45 lecture contact hours. The CS2013 then consists of the equivalent of roughly four 3-credit hour semester courses worth of Tier-1 material and roughly three 3-credit hour semester courses worth of Tier-2 material, spread throughout the entire curriculum.

While acknowledging that every program has differing educational objectives, use of professional society standards provides metrics which can gauge the success of the program against a national model. Such metrics suggest an infrastructure for direct assessment that allows comparison against discipline-wide expectations and to allow reflection on the need, causes, and appropriateness of any major deviations from the widespread consensus proposed by the discipline’s professional society.

**Continuous periodic direct measurements of retained relevant knowledge**

We have worked with our program faculty to produce a mapping of which CS2013 Knowledge Topics are prerequisite to or developed in each of the core/mandatory courses in our computer science program. Our initial assessment framework is limited to mandatory “core” courses. Students will gain additional experience in many core knowledge topics in their elective coursework. However, the topics and amount of coverage will necessarily vary based upon the selected electives. Thus, initial observations are limited only to core/mandatory courses.

For each course, the faculty has indicated what knowledge topics are developed or assumed (prerequisite) in the semester-based course offerings. We propose that an appropriate method to assess relevant retained knowledge is to perform a direct assessment of each knowledge topic not only in the course that develops that knowledge, but when possible, at the beginning of a subsequent course (or courses) that utilizes and builds on the topic.

Summative grading rubrics are, when possible, deployed at the start of the next course in the core course sequence (Figure 4, below). These assessment points allow better evaluation of the retention of expertise as measured prerequisite knowledge coming into each course. Assessment of prerequisite knowledge also allows assessment of differences among learning pathways, and are less subject to instructor- or course-related bias.
Immediate value to participants

We propose that the assessments be required of all students entering every core course. Furthermore, we propose that the results not affect their course grade. As this effort is not associated with a course grade there is no need to proctor or use valuable classroom time on the assessment. The assessments are simply delivered as on-line standardized quizzes (Figure 5, below). We have found that restricting access to on-line classroom materials until the assessment quiz for the course is completed gives complete class participation in the assessment.

In our experience, students are very open about their level of mastery of concepts assessed in not-for-credit surveys of prerequisite knowledge. The feedback from these assessments is immediately useful to students as it calls up old ideas (helping them to be ready for new related knowledge). This immediate feedback can also reduce anxiety regarding the sufficiency of their mastery of assumed prerequisite knowledge or identify specific areas where they can be coached to better prepare for succeed in a new course. As students find the feedback valuable to them personally, they are more likely to give significant and frank effort in the assessment process.

As a direct assessment of student preparedness, this data should be less biased than indirect assessments that ask students their opinion of their ability. Differences in self-expectation that may exist among students due to experience or demographic are removed. Thus, students/faculty get a more accurate measure of how well each student is prepared.
Consider the following segment of code in a Java-like programming language. Assume that there are no syntax errors.

```java
int[] m = {2,3,4,5,6};
int n = 0;
int x = 0;
for (int val = 0; val < m.length; val++)
{
    if (val % 2 == 1)
    {
        n = n + val;
        x = x + 1;
    } // end-if
} // end-for
```

What is the most likely use for the code segment above?

A) Calculating the total sum of the values held in array m.
B) Calculating the average of the values held in array m.
C) Calculating the number of even values held in array m.
D) Calculating the average of odd values held in array m.
E) Calculating the number of values held in array m.

Equally important, the results of these prerequisite surveys can be made immediately available to the faculty teaching the course in which the examination is held. If the faculty member sees weakness in prerequisite knowledge then they are able to act to help address the problem immediately. The assessment can help identify individual students that might require additional help as well as identify potential systemic deficiencies introduced by previous poor instruction, variation in schedule due to weather/emergency, differing pathways for preparation (such as transfer courses), or the like. Based upon assessed performance, the faculty can tailor any necessary review of prerequisite topics appropriately to the needs of each term’s student preparation.

Assessment overhead and administrative burden

Ease of assessment delivery allows the potential direct assessment of every student every term in every core course. As these assessments are delivered as on-line standardized examination, they require very little class time or faculty effort to administer. Each knowledge topic is mapped to relevant ABET engineering criteria 2000 CAC/EAC a-k criteria listed in ABET’s Criterion 3: Student Outcomes [2,3]. This allows the data to be used by class or longitudinally by student to assess continuous improvement of the program overall against ABET Engineering criteria in a well-defined and straightforward manner.

The most significant administrative burden is in the initial development, validation, and continuous improvement of the assessment questions. The initial burden of assessment development requires significant faculty involvement and may require multiple years of effort to construct assessment questions for every core course. The measurements for a knowledge area may be skewed by a set of poor assessment questions, thus continuous improvement of the questions in parallel with the improvement of curriculum remains an ongoing administrative effort.
Conclusion
This assessment infrastructure allows for an assessment of retained knowledge, topic by topic, for each individual student, course, and term. When collected with appropriate demographic information, these assessments allow the differential measurements of knowledge retention under any number of pedagogical variables. The success of new instructional styles, laboratory techniques, or technologies for developing knowledge can be assessed against different approaches.

Every contemporary engineering discipline has a professional society that helps identify the core concepts of the discipline. Indeed, most engineering disciplines have standardized examinations of some sort that are used to demonstrate student proficiency for licensure or graduate studies. Questions of this sort can be used at the start of core courses or time points to assess student knowledge of prerequisite topics developed earlier in any program of study. These assessments can be delivered as online questions to minimize cost and maximize participation. When collected with appropriate demographic information, this rich set of data can guide program improvement more effectively than many existing program assessment plans. Although we present this infrastructure in the context of Computer Science, we believe that the approach can be applied to implement an infrastructure for effective assessment program for any engineering discipline.

Bibliography

Relevant excerpts from the minutes of the Undergraduate Studies Committee (UGSC) for Computer Science and Engineering (CSE) in the College of Engineering and Computer Science (CECS)

Oct 26 2012
- Initial discussion of 2012-2013 CS Program Semester Curriculum Assessment v1.0

Nov 16 2012
- Review of BSCEG PEOs
- Doom will distribute his current slides with possible items to include in the PEOs. The committee will contribute their suggestions for the upcoming Dept. Advisory Board Meeting.

April 05 2013
- Review of Assessment Data
  - Review of initial Knowledge Topics that have success rates of 30% or lower.
  - Several 'poor' questions identified for correction. Associated data to be removed.
  - Potential issues discussed regarding simple data structures: Stacks/Queues and Linked Lists
  - Data this year is insufficient for action
  - Starting next semester one UGSC meeting each term will be devoted to consideration of this assessment data and determining if action should be considered.

September 11, 2013
- Consideration of Program Educational Objectives for update

September 25, 2013
- Program Educational Objectives
  - PEOs revised based on both outside and internal feedback
  - Reduced to four key points for each program
  - CS program provides more depth with FOCUS on software
  - CEG provides a BROAD range
  - Has been set up so it is easily assessable
  - CEG program will be including operating systems to its program with the changes.
  - Formal Vote for the adoption of the new PEO’s
  - Vote to adopt the BSCS PEO – APPROVED UNANIMOUSLY
  - Vote to adopt the BSCEG PEO – APPROVED UNANIMOUSLY

November 6, 2013
- CS BA Remove PHL2230 and reintroduce software Engineering CEG4110 to the BACS
  - With the math already in the program this class is redundant
  - Philosophy may have changed the semesters that the class is offered in
  - 1200 to 2200 is the replacement for the math pathway
  - What are we replacing the 3 credit hours with
  - Program directives of the BA program may effect this
- PEO’s for BACS program
  - Keeping unaccredited at the moment keeps it flexible
  - Where should BA program be
  - Same as both BS programs keep Expert, Agile, Engaged
  - Under Agile removed graduate as they are not ready for our graduate program when they graduate
  - 4th section Diverse/Eclectic was changed to Applied
  - Add in software development/management, data science, cyber systems provide a larger set of skills
  - Flexible with other fields
  - Changed data sources to domains
- BA Computer Science is not just the BS program for those who can’t pass the math requirements!
  - Graduates can apply computing and software development principles to a diverse range of Data sources such as analytics, data science, informatics, management, etc
- Difference between BS and BA: More applied, Able to add a minor, for Research/Graduate school – BS, for breadth and variety – BA

- Take PEOs to external Advisory Board for comment as follows:

- BA CS program educational objectives
  - Expert: Graduates of the Computer Science program are employable as computing professionals and will be recognized by their employers as well-prepared for their career in computing
  - Agile: Graduates understand that education is a lifelong process and are well prepared for continuing studies
  - Engaged: Graduates demonstrate an appreciation for the professional, social, ethical, and leadership roles of computing professionals
  - Applied: Graduates can apply computing and software development principles to a diverse range of domains, such as analytics, data science, informatics, management, etc.

December 4, 2013
Program Evaluation Meeting

- New system to evaluate programs
  - Quiz given to collect data
  - Almost done for every core CS cores
  - Seeing if students are learning what they need

- Committee is now going to evaluate the program

- Only 2 semesters of data

- This is the baseline not a complete picture

- Quiz not enforced as it is not graded
  - Should be taken seriously by students

- Need to make the time and data collected as useful as possible

- Most data is on the CS1160, CS1161 vs CS1180 pathways

- CS1181 Data
  - Students in CS1160 spring 13, CS1161 summer 13
  - Students in CS1180 fall 12, spring, or summer 13
  - Overall students do the same on both pathways
    - Primitive types students who take CS1160/1 do half as well (25%) at students who take CS1180 (48%)
      - Might speak to mathematical preparation or focus
      - Where in discreet math
      - May need to focus more on the math skills
      - 1160/1 generally take CS1200 and 2200 CS1180 take MTH 2530
      - Math levels should be equal for both classes
  - Event driven and reactive programming
    - 1160/1 students do better (75%) than 1180 students (52%)
      - Listener objects and listening libraries understood better by 1160/1
      - 1160/1 had more time in the laboratories
      - Very small sample not large enough to be acted on
  - Test students ability to find the answer as well as solve it
  - Question posed asking what the students did choose for the answers vs just the final percentage
  - In future have a percentage breakdown
  - SDF/Fundamental Programming Concepts
    - 1160/1 do better (75%) than 1180 (45%)
    - Not a pattern is interesting
    - Need to watch this
    - Possible issue with student just choosing 1st right answer
    - Need to make student look for right answer not just choose 1st one
    - Next question both sets of students did not do well on so possibly too hard
  - SDF/Fundamental Data Structures: Arrays
- 1160/1 (0%) 1180 (20%)
  - Any question that required students to trace through code and figure out what it will do they did terrible on.
  - Students are not good at walking through algorithms
  - Next questions show a pattern of questions like this that students do not do well on
  - Need to consider what to do about this

CEG3310 Data
- Students took CS1160 fall, CS1161 spring, CS1181 summer
  - Basic type systems: Compound types building
    - None of the 1160/1 (0% all 4 students) got it right but the 1180 (73%) did well.
    - Question probably not too hard but 1160/1 students did not do well
    - 1160 students don’t cover arrays
      - Students have 1160/1 and 1181 at this point
    - First offering in 1160, 1161, and 1181
      - Take with grain of salt
      - Only 4 students
    - Need to watch if see this next time
      - Means we need to spend more time on arrays
      - 1181 doesn’t spend a lot of time on arrays
      - If don’t get in 1161 then not able to fix problem in 1181
      - Problem could be with the need to trace patterns and the students fear of this
  - Fundamental Programming concepts: conditional and iterative control structures
    - Basic and simple question
    - Not good for 1160/1 (0%) not great for 1180 (58%)
    - A number of students in computer org don’t seem to know how to program
    - Don’t have the ability to put together primitives to accomplish a task in any context
    - All students in this class have either 1160, 1161, and 1181 OR 1180, and 1181 before going to this class
    - Possible reason is the way classes were weighted last year
    - Students were able to keep their grades up by completing projects where they could get help
    - This has been changed this year now exams carry more weight so this may make a difference
  - Algorithms and Design: Iterative and recursive mathematical functions
    - CS1160/1 0% CS1180 61%
    - Question as to how many students put recursion for the answer
    - This speaks to the pathway that was taken to get to 1181
    - Need to watch so students that are put into 1181 get the skills necessary to move on
    - Sample is too small to be acted on but if still happening a year from now will need to do something about it
  - Basic type systems: Reference types
    - In Java/C-like languages, variables that store reference types are:
      - Again 1160/1 pathway small sample (0%) 1180 (42%)
      - Hope that students on both pathways take this seriously
  - Fundamental Data Structures: Strings
    - 44% CS1180 able to answer question 0% 1160/1
    - Need rewrite and fix C-like programming
    - By time get through 1180 wording should not be an issue
  - Fundamental data structures: Simple linked structures
    - Nobody did well on this 1160/1 0% 1180 16%
- Need to let all instructors that teach data structures know the importance of the quiz being taken
  - Bring this up at the faculty meeting
- Retention rate used to be 85% failure in 1st 2 years now it’s approximately 60%.
<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>An ability to apply knowledge of computing and mathematics appropriate to the discipline</td>
</tr>
<tr>
<td>b</td>
<td>An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution</td>
</tr>
<tr>
<td>c</td>
<td>An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs</td>
</tr>
<tr>
<td>i</td>
<td>An ability to use current techniques, skills, and tools necessary for computing practice.</td>
</tr>
<tr>
<td>j</td>
<td>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>k</td>
<td>An ability to apply design and development principles in the construction of software systems of varying complexity.</td>
</tr>
</tbody>
</table>
**SLO’S FOR ALL QUESTIONS**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1455</td>
</tr>
<tr>
<td>b</td>
<td>818</td>
</tr>
<tr>
<td>c</td>
<td>770</td>
</tr>
<tr>
<td>l</td>
<td>63</td>
</tr>
<tr>
<td>j</td>
<td>294</td>
</tr>
<tr>
<td>k</td>
<td>450</td>
</tr>
</tbody>
</table>

- If a question tests multiple SLO’s it is counted multiple times.
SLO FOR ALL QUESTIONS

![Bar chart showing SLO for all questions with categories a, b, c, l, j, k. The chart indicates the proportion of false and true responses for each category.](chart.png)
<table>
<thead>
<tr>
<th></th>
<th>Fall 2012</th>
<th>Spring 2013</th>
<th>Summer 2013</th>
<th>Fall 2013</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>92</td>
<td>321</td>
<td>52</td>
<td>172</td>
</tr>
<tr>
<td>b</td>
<td>46</td>
<td>161</td>
<td>26</td>
<td>86</td>
</tr>
<tr>
<td>c</td>
<td>46</td>
<td>161</td>
<td>26</td>
<td>86</td>
</tr>
<tr>
<td>j</td>
<td>23</td>
<td>78</td>
<td>13</td>
<td>43</td>
</tr>
<tr>
<td>k</td>
<td>23</td>
<td>78</td>
<td>13</td>
<td>43</td>
</tr>
</tbody>
</table>
CS 1181: THROUGH TIME

- Fall 2013
- Summer 2013
- Spring 2013
- Fall 2012
## DATA CS 3100 KNOWLEDGE AREA

<table>
<thead>
<tr>
<th>Knowledge Area</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL (Algorithms &amp; Complexity)</td>
<td>32</td>
</tr>
<tr>
<td>AR (Architecture &amp; Organization)</td>
<td>41</td>
</tr>
<tr>
<td>DS (Discrete Structures)</td>
<td>95</td>
</tr>
<tr>
<td>PL (Programming Languages)</td>
<td>16</td>
</tr>
<tr>
<td>SDF (Software Development Fundamentals)</td>
<td>31</td>
</tr>
<tr>
<td>KA</td>
<td>Knowledge_Unit</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>AL</td>
<td>Basic Analysis</td>
</tr>
<tr>
<td>AL</td>
<td>Fundamental Data Structures and Algorithms</td>
</tr>
<tr>
<td>AR</td>
<td>Assembly level machine organization</td>
</tr>
<tr>
<td>AR</td>
<td>Machine-level representation of data</td>
</tr>
<tr>
<td>DS</td>
<td>Basics of Counting</td>
</tr>
<tr>
<td>DS</td>
<td>Discrete Probability</td>
</tr>
<tr>
<td>DS</td>
<td>Graphs and Trees</td>
</tr>
<tr>
<td>PL</td>
<td>Object-Oriented Programming</td>
</tr>
<tr>
<td>SDF</td>
<td>Algorithms and Design</td>
</tr>
<tr>
<td>SDF</td>
<td>Fundamental Data Structures</td>
</tr>
</tbody>
</table>
CS 3100 BY KNOWLEDGE UNIT

- Basic Analysis
- Fundamental Data Structures and...
- Assembly level machine...
- Machine-level representation of...
- Basics of Counting
- Discrete Probability
- Graphs and Trees
- Object-Oriented Programming
- Algorithms and Design
- Fundamental Data Structures

Colors:
- AL: AL
- AR: AR
- DS: DS
- PL: PL
- SDF: SDF

Legend:
- Wrong
- Correct
<table>
<thead>
<tr>
<th>Pro</th>
<th>Num</th>
<th>sem</th>
<th>year</th>
<th>Took</th>
<th>in class</th>
<th>possible in class</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>1181</td>
<td>Fall</td>
<td>2012</td>
<td>25</td>
<td>41</td>
<td>54</td>
</tr>
<tr>
<td>CEG</td>
<td>3320</td>
<td>Fall</td>
<td>2012</td>
<td>34</td>
<td>50</td>
<td>54</td>
</tr>
<tr>
<td>CS</td>
<td>1181</td>
<td>Spring</td>
<td>2013</td>
<td>166</td>
<td>88</td>
<td>165</td>
</tr>
<tr>
<td>CS</td>
<td>3100</td>
<td>Spring</td>
<td>2013</td>
<td>6</td>
<td>22</td>
<td>27</td>
</tr>
<tr>
<td>CS</td>
<td>3100</td>
<td>Spring</td>
<td>2013</td>
<td>5</td>
<td>21</td>
<td>27</td>
</tr>
<tr>
<td>CEG</td>
<td>3310</td>
<td>Spring</td>
<td>2013</td>
<td>32</td>
<td>49</td>
<td>54</td>
</tr>
<tr>
<td>CEG</td>
<td>3320</td>
<td>Spring</td>
<td>2013</td>
<td>37</td>
<td>41</td>
<td>48</td>
</tr>
<tr>
<td>CS</td>
<td>1181</td>
<td>Summer</td>
<td>2013</td>
<td>13</td>
<td>18</td>
<td>54</td>
</tr>
<tr>
<td>CS</td>
<td>3100</td>
<td>Summer</td>
<td>2013</td>
<td>23*</td>
<td>22</td>
<td>30</td>
</tr>
<tr>
<td>CS</td>
<td>1181</td>
<td>Fall</td>
<td>2013</td>
<td>43</td>
<td>68</td>
<td>108</td>
</tr>
<tr>
<td>CS</td>
<td>3100</td>
<td>Fall</td>
<td>2013</td>
<td>1</td>
<td>9</td>
<td>35</td>
</tr>
<tr>
<td>CS</td>
<td>3100</td>
<td>Fall</td>
<td>2013</td>
<td>5</td>
<td>12</td>
<td>35</td>
</tr>
<tr>
<td>CEG</td>
<td>3310</td>
<td>Fall</td>
<td>2013</td>
<td>17</td>
<td>55</td>
<td>108</td>
</tr>
</tbody>
</table>
DATABASE UPDATES

- Primary Keys
- CRN became CRN plus year
- Mapped SLOs to PEO
DISCUSSION

• If student retakes the quiz keep both or update answer
  • Current keep both
<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>An ability to apply knowledge of computing and mathematics appropriate to the discipline</td>
</tr>
<tr>
<td>b</td>
<td>An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution</td>
</tr>
<tr>
<td>c</td>
<td>An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs</td>
</tr>
<tr>
<td>d</td>
<td>An ability to function effectively on teams to accomplish a common goal</td>
</tr>
<tr>
<td>e</td>
<td>An understanding of professional, ethical, legal, security and social issues and responsibilities</td>
</tr>
<tr>
<td>f</td>
<td>An ability to communicate effectively with a range of audiences</td>
</tr>
<tr>
<td>g</td>
<td>An ability to analyze the local and global impact of computing on individuals, organizations, and society</td>
</tr>
<tr>
<td>h</td>
<td>Recognition of the need for and an ability to engage in continuing professional development</td>
</tr>
<tr>
<td>i</td>
<td>An ability to use current techniques, skills, and tools necessary for computing practice.</td>
</tr>
<tr>
<td>j</td>
<td>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>k</td>
<td>An ability to apply design and development principles in the construction of software systems of varying complexity.</td>
</tr>
</tbody>
</table>