CS Department Assessment Plan
Fall 2015 Update

Learning outcomes for majors, followed by potential measurement techniques:

1. **[2010, 2015]** Students should be able to write software in several programming languages, including intermediate to expert proficiency in one and at least intermediate proficiency in another. *Look at final projects for CS 257 (Java), and some programs from CS 251 (C). Classify each as indicating whether or not it shows intermediate-to-expert proficiency.*

2. **[2013]** Students should be able to quickly and independently learn new programming languages and software systems. This includes being able to find appropriate resources for learning those languages. *Survey seniors and alumni. Ask about learning new languages for a particular application, and how easily they were able to achieve it.*

3. Students should be familiar with imperative, object-oriented, and functional programming language paradigms, and can demonstrate ability to write code appropriate for each paradigm. *Look at projects from CS 111 (imperative), CS 257 (object-oriented), and CS 251 (functional).*

4. Students should be able to evaluate and trade off design decisions in programming languages, as well as demonstrate understanding of how a programming language is implemented. *Look at performance on targeted exam questions from CS 251.*

5. **[2017]** Given a particular application, students should be able to formulate the problem to be solved in an algorithmic framework, and appropriately pose algorithms that would solve the problem. *Look at performance on relevant assignments in CS 252. Construct such an assignment if necessary.*

6. **[2011]** Students should be able to mathematically analyze the time and space requirements of algorithms and their associated data structures, at least at a level appropriate for undergraduates. *Ditto to above.*

7. **[2014]** Students should be able to describe the theoretical limitations of computational processes, and utilize the techniques by which these limits are proved. *Ditto to above, this time in CS 254.*

8. **[2012]** Students should be able to communicate effectively in writing and in speech about both the technical aspects of algorithms, computers, and software, as well as the social and ethical impact of information technology. *Look at submissions for comps, specifically design documents, final web pages and/or papers, and final presentations.*

9. Students should be able to articulate organization of both the hardware and the system software that underlies modern computers. *Look at student response to data*

path question/prompt on CS 208 exams/projects.

10. (a) Students should be able to collaborate as a member of a team to design and complete a computing project of a scale that would be expected to take multiple terms to complete. Students should be able to do appropriate background research on the topic of interest, and integrate ideas learned into their projects. *Use comps oral exam notes, project adviser surveys, and peer evaluations.*

    (b) Students should be able to work on their own to design and complete a computing project of a scale that would be expected to take at least five hours to complete. *Look at projects from CS 111 or CS 201.*

Management:

- The department chair or department assessment coordinator will keep track of which tasks we want to do which year / time of year.
- In spring of each year, we can plan what the assessment goal for the following year will be. That way, if we're going to be using assessment techniques embedded in courses/comps, we can make sure that they are appropriately there.
- The compilation of the data and coding of the results will be done by an individual; in the case of most of the learning outcomes, there is a specialist in the department who can do this most easily. The aggregated results, however, will be shared with everyone by email before our department retreat. Discussion can then begin at our department retreat during the summer.
- We'll look at one of the outcomes each year.

Completing the Circle:

- At our department retreat and possibly in fall term, we'll discuss curricular changes to try to make improvements if necessary based on assessment results. We'll then implement those changes for the upcoming academic year.
- One particular challenge is going to be making sure that anyone teaching the course knows to implement those changes. We know that we perhaps don't coordinate as well as we could what material we teach in shared courses. We should probably set up some kind of shared resource, such as a document for each course in our department subversion repository, that lists particular tasks and/or goals for that course.

Finally: we should also consider producing a set of learning outcomes for students who take CS 111.