CSC 303 Teaching Computer Science

1. CSC 303 Teaching Computer Science

2. **credit units** 2  **contact hours** 4

3. **Course Coordinator**: Clint Staley

4. **Textbook**: (and/or other required material) Course Notes

5. a. **Course Description**: Practical coverage of educational techniques appropriate for tutoring in CSC/CPE undergraduate courses, including Socratic methods for tutoring of technical topics, design of test questions and grading rubrics, and lecture presentation. Intended for CSC/CPE/SE students interested in tutoring, grading, or a career in teaching computer science. 1 lecture, 1 laboratory. Not available for technical elective credit.

   b. **Prerequisite**: CSC/CPE 103, with a grade of C- or better, or equivalent.

   c. **Required/Elective/Selective Elective for CPE, CSC, EE, SE**

<table>
<thead>
<tr>
<th></th>
<th>CPE</th>
<th>CSC</th>
<th>EE</th>
<th>SE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elective</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selective Elective</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6. a. **Course Goals/Outcomes**

   Students will be able to effectively assist professors in undergraduate or graduate courses as tutors, lab assistants, or graders.

   - Are able to explain computer science concepts using proper sequencing, gauging the difficulty of each concept and decomposing concepts where necessary to make each step in a presentation more accessible
   - Anticipate and head off common miscomprehensions of concepts, development of consistent nomenclature, and developing visualizations of technical ideas appropriate for pedagogy
   - Have the ability to rapidly read, comprehend and debug code of widely varying quality.
   - Design test questions or course exercises
   - Design rubrics
   - Design of online exercises.

   b. **How Student Outcomes addressed**

   (“B” = Basic level, “I” = Intermediate level, “A” = Advanced level)
7. **Major Topics Covered: (number of lecture hours each)**

1. Technical presentation and writing. The course covers, in detail, methods of explaining technical concepts, both verbally and in written form. This includes proper sequencing, gauging the difficulty of each concept and decomposing concepts where necessary to make each step in a presentation more accessible, anticipating and heading off common miscomprehensions of concepts, development of consistent nomenclature, and developing visualizations of technical ideas appropriate for pedagogy.

2. Fast analysis of code, and debugging. A major emphasis of 303 is tutoring CSC students. While tutors do not directly debug student code, they must understand the code and see the bugs in order to tutor effectively. Tutoring introductory students thus develops an ability to rapidly read, comprehend and debug code of widely varying quality. Indeed, most tutors would agree that the best experience they get at code *reading* comes from their tutoring experience, since reading others' code is not heavily emphasized in other coursework.

3. Design of exercises and associated rubrics. Designing test questions or course exercises is quite technical. One must design coding or theoretical challenges that cover a set of technical concepts, are easily understood (especially for test questions) and for which a clear grading rubric can be developed. This generally requires running through a number of different technical challenges, evaluating each according to these criteria, and picking the best.

4. Design of online exercises. The design of online exercises is remarkably technical. As an example, a popular if-statement exercise developed by the proposer involved sophisticated tree data structures, map-coloring algorithms, and automated solution-code generation. Similar challenges arise in most online exercise designs.