### UNIVERSITY OF TEXAS AT AUSTIN

#### COURSE-INSTRUCTOR SURVEY

Downing, Glenn P  
C S371P  53115  
OBJECT-ORIENTED PROGRAMMING

Enrollment = 85
Surveys Returned = 74

### *** PROVISIONAL REPORT ***

<table>
<thead>
<tr>
<th>Item Description</th>
<th>Strongly Disagree</th>
<th>Disagree</th>
<th>Neutral</th>
<th>Agree</th>
<th>Strongly Agree</th>
<th>No. Replies</th>
<th>AVG</th>
</tr>
</thead>
<tbody>
<tr>
<td>COURSE OBJECTIVES DEFINED-EXPLAINED</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>18</td>
<td>48</td>
<td>74</td>
<td>4.4</td>
</tr>
<tr>
<td>INSTRUCTOR PREPARED</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>65</td>
<td>74</td>
<td>4.9</td>
</tr>
<tr>
<td>COMMUNICATED INFORMATION EFFECTIVELY</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>18</td>
<td>52</td>
<td>74</td>
<td>4.6</td>
</tr>
<tr>
<td>STUDENTS ENCOURAGED-ACTIVE ROLE</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>18</td>
<td>52</td>
<td>73</td>
<td>4.7</td>
</tr>
<tr>
<td>INSTRUCTOR AVAILABILITY</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>20</td>
<td>50</td>
<td>74</td>
<td>4.6</td>
</tr>
<tr>
<td>COURSE WELL-ORGANIZED</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>17</td>
<td>51</td>
<td>74</td>
<td>4.5</td>
</tr>
<tr>
<td>STUDENT FREEDOM OF EXPRESSION</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>14</td>
<td>51</td>
<td>74</td>
<td>4.5</td>
</tr>
<tr>
<td>HELPFUL COURSE MATERIALS</td>
<td>3</td>
<td>3</td>
<td>13</td>
<td>14</td>
<td>51</td>
<td>74</td>
<td>4.2</td>
</tr>
<tr>
<td>STUDENT PERCEPTION OF AMOUNT LEARNED</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>19</td>
<td>49</td>
<td>74</td>
<td>4.5</td>
</tr>
<tr>
<td>OVERALL INSTRUCTOR RATING</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>14</td>
<td>52</td>
<td>74</td>
<td>4.5</td>
</tr>
<tr>
<td>OVERALL COURSE RATING</td>
<td>1</td>
<td>7</td>
<td>5</td>
<td>20</td>
<td>41</td>
<td>74</td>
<td>4.3</td>
</tr>
<tr>
<td>STUDENT RATING OF COURSE WORKLOAD</td>
<td>5</td>
<td>30</td>
<td>38</td>
<td>0</td>
<td>1</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>OVERALL UT GRADE POINT AVERAGE</td>
<td>0</td>
<td>5</td>
<td>13</td>
<td>26</td>
<td>30</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>PROBABLE COURSE GRADE</td>
<td>22</td>
<td>23</td>
<td>28</td>
<td>1</td>
<td>0</td>
<td>74</td>
<td></td>
</tr>
</tbody>
</table>

For the computation of averages, values were assigned on a 5-point scale so that the most favorable response was assigned a value of 5 and the least favorable response was assigned a value of 1.
1. Foremost, this course has the wrong title. We did not learn about OOP, instead we were introduced to C, and the title blurb ought to reflect that because it’s a lie. Since I didn’t learn OOP, I still can’t learn what the “average” learn, and instead walked out with knowledge of a programming language I hope to never use. Secondly, Downing’s “Socratic dialogue” teaching style does encourage thinking through the material while he’s going over it, but mostly it is slow and got more annoying over time. On the other hand, I did get better at using github and unit testing, and I liked having quizzes at the start of every class.

2. This was a great course, one of the best I’ve taken in the CS department. Professor Downing is one of the best instructors I’ve had at UT. Although I understand that the TAs are probably busy with their other courses, but the projects and quizzes are rarely ever graded promptly. Sometimes quizzes took a week to grade when they should’ve been graded within 2 days as stated in the syllabus. The same was true for the projects. I feel like I would have learned a great deal more from this class. Instead I had to try and decipher Professor Downing’s explanations of the code which were not recorded or online. If the lecture’s were recorded or the code based would be a good way to help us remember more material in my opinion. Also, on one of the tests I got 0 points on one question, though, It is hard to stay focused when all I see is code. Presentations and examples to relate the concepts that are not just subjects previous teachers had brushed over and never fully explained. I may not have enjoyed the in-class experience, but this professor was neither intimidating nor condescending, and was a really, really good teacher, managing to make clear several concepts that were important to me.

3. I felt uncomfortable and under a lot of pressure with the way questions were asked in the class, but outside of that the class itself was well organized and thought out. Unfortunately, it kind of felt like an “intro to writing C well” course. I had hoped it would be a bit more theory based, and talk more about things like design patterns, etc that OOP uses, and less about pointers and such.

4. Really enjoyed the course. I felt like it was more of a C course than OOP though. I mean I learned a ton and the projects were great but it just seems like I still don’t really know what OOP really is or what the main concepts for OOP are. I’ve gotten pretty darn good at C though and I look forward to using it more.

5. This class was a waste of time. It was more a class about C behavior than Object Oriented Programming. The lectures were hard to follow or take notes on because the professor would throw out scraps of examples. The only theory I learned was from the textbook which did not help prepare me for the exams or projects. It was hard to learn the type of material for the tests because there were barely any practice problems similar to the tests. The instructor was also very unprofessional during office hours. On multiple occasions he was very late or completely absent from office hours without warning. He was also condescending and rude during office hours and on the class discussion board.

6. Great lecturer, very well structured course. You and your classes are truly valuable assets to the CS department. Please continue to impart love and passion to the students.

7. Course was extremely well organized and thought out. Unfortunately, it kind of felt like an “intro to writing C well” course. I had hoped it would be a bit more theory based, and talk more about things like design patterns, etc that OOP uses, and less about pointers and such.

8. I felt that the project deadlines were too unforgiving. In particular, just for not having the right files when they were submitted ON TIME meant my project was late. There was even a time when I submitted an incorrect SHA and lost points, even though all my files were pushed before the deadline. Maybe you could add 1-2 late days. That way if I have to use one because I mess up a submission, I won’t have to lose points to learn my lesson the first time. I really enjoyed hearing talks from companies. I would definitely continue to do that. Sometimes the quizzes are ridiculously specific. I also think that a class named OOP should have more OOP projects. Voting and Allocutor only required C programming skills, not OOP design skills.

9. The energy and style that Downing uses I feel is extremely conducive for learning and he made learning enjoyable as he could and interesting and was good at attempting to keep everyone engaged. The class itself was very useful to learn more details of the ins and outs of C and java in comparison, with the projects backing up and reinforcing the information we learned in class, and teaching C along the way, although I feel the tests are hard to study for and are unrepresentative of our knowledge and the rest of the class.

10. I thoroughly enjoyed the allocator project - it helped me get a grasp on memory management and feel more comfortable. Learning various design patterns - singleton, handle, etc. this brought discipline and structure to my code. I had only one issue this semester based on function rule. It seems a bit arbitrary to me, and basing the tests off of gcov output would be better as it is a metric that actually conveys something about how good or bad the unit tests are. Also, in my experience in industry, supervisors want to see good unit test coverage and comprehensive functional tests. I think writing tests in accordance with these expectations would be more beneficial.

11. I really appreciated the preparation this class gave me for the ‘real world’ I found it super useful to learn about the different tools and have companies speak to us. Projects were fine, not too hard, not too easy.

12. Dr. Downing was a very good instructor. He encouraged to participate in the class and gave plenty of opportunity to ask questions. His exams were fair and I believe the grading scheme accurately reflects the amount of learning that goes on in the class. The only thing that I didn’t like was the frequency of the quizzes, but even that wasn’t too bad.

13. I felt uncomfortable and under a lot of pressure with the way questions were asked in the class, but outside of that the professor was neither intimidating nor condescending, and was a really, really good teacher, managing to make clear several subjects previous teachers had brushed over and never fully explained. I may not have enjoyed the in-class experience, but this class itself was invaluable, and the teacher excellent.

14. Great class. I liked the question answer format made me stay focused. The project having code all the time does not help much though, It is hard to stay focused when all I see is code. Presentations and examples to relate the concepts that are not just code based would be a good way to help us remember more material in my opinion. Also, on one of the tests I got 0 points on one answer because I did not understand the question well. A simple input output expectation on the exam would help understand what we are supposed to solve in the exam.

15. The excessive issue tracking on github especially the 10 issues from the requirements was a very large hindrance since there seemed to be only 5 or 6 concrete requirements. This meant that I had to pick 4 ridiculous requirements to solve among them was typically the requirement that stated I needed 10 issues for requirements. Additionally, the heart of lecture really focused on Professor Downing’s explanation of the course. For the concept of writing tests for code, I thought the written explanations written down I feel like I would have learned a great deal more from this class. Instead I had to try and decipher
what was being implied by the code that was posted. I really like that the code was online though.

16. There was too much week then too few if they talked more about the culture and an average day in addition to the tools they use. The two tests were challenging projects using Git and Github. The tools learned in the class are now a regular part of my workflow. The projects were well 31. Before taking this class, I had no experience with C and little experience using Git and GitHub. Now, I do all of my personal input I had that could make the class a better learning experience, apart from that I think its perfect !

20. I really enjoyed the class, however I do have a few complains. The daily quizzes were fine, but I had no idea what to study for them. Also, comments on the example code that is posted online would be helpful for understanding it a little better!

21. I feel like he tried to squeeze a bit too much information into the class. It was hard to wrap my brain around so much material in such a short time. Also I feel like we were not given enough time on some of the programming assignments and the requirements seem a bit excessive. the unit tests make sense but they encourage you to write very few methods.

22. This was a fantastic course. I wish in-lecture participation were more encouraged. I really appreciated getting to know some of the software tools used in the industry. The speakers were interesting, and the projects were of appropriate difficulty. I enjoyed how passionate you are about the course material and the "high points" of the class.

23. I really enjoyed the guest speakers, but felt that a guideline should be set in words

24. Thanks Dr.Downing.

25. Dr. Downing is the Best!

26. I feel like this was more of a C and STL course rather than an OOP course though. Overall, I really enjoyed the class and thought it was amazing for my first upper division course, but the tests killed it for me.

27. A gem for UTCS. Definitely one of my favorite professors at UT. What a pleasure to sit in his class. He genuinely cares for the student and is very approachable. He's got some new interesting concepts that industry professionals to talk to us. His assignments further solidified the information he taught in class. Super organized. My only gripe with him would be that studying for his exams are entirely dependent on the student. He offers no supplemental material or past exams to get a sense of what his tests might be. Nevertheless, his lecture material is extensive and is posted online. I would have liked to have assignments that employers would be interested in seeing, but that's a minor complaint.

28. The projects were some high points of the class with them ranging from writing our own allocator to writing a board game with creatures in it. The only project that wasn't great was voting just because of not being able to tell what a valid case was and wasn't. Lectures were also pretty good which Professor Downing going in some serious details about everything. Tests... boy do I hate tests... The issues with the tests are that most of the questions are vague at best. During the first exam I got up a few good times to make sure what I was doing was right. Quizzes were alright. Overall I learned a lot from this course and thought it was amazing for my first upper division course, but the tests killed it for me.

29. Sometimes certain aspects of the project were unclear to begin with and then there seemed to be a lot of confusion on the projects. I really liked most of the project. I had a lot of good times to make sure what I was doing was right. Quizzes were alright. Overall I learned a lot from this course and thought it was amazing for my first upper division course, but the tests killed it for me.

30. Thanks for teaching me so much about OOP! Even though I thought that I had sufficiently learn C via Generic, this class just made me feel that I have a lot more to learn! Thank you again for being such a great instructor!
but fair. The questions did a good job of implementing what we learned in class without regurgitating the material.

32. I liked the class a lot. The questions on the tests are sometimes very ambiguous and it is unclear what it is expected. When I tried to come up with enough git issues was sometimes silly. But that was the only complaint. Projects I liked all of the projects. Allocotor was similar to a project we did in 439H, but they were good on the whole. Speakers Bloomberg and Twitter were great. Twitter was a letdown - they were unprepared and made it awkward at times. Schlumberger - no strong feelings one way or the other.

33. I underestimated the difficulty of this class. I already had some experience with c , and I think this made me overconfident. I did not like the tests, but I really wanted to get us to come to class. I think there was an over-emphasis on the use of git early in the course. It was kind of tedious dealing with all the github issue tracking stuff I found myself making up fake issues to meet the project requirements. I wish the projects had been more object oriented, projects 4 & 5 were the only ones that seemed to really benefit from oop. I didn’t care for the speakers, nothing personal, I just didn’t get much out of these particular speakers.

34. I found myself to be a pretty good programmer. But this course was tough. Learning all of the tools was cool. Thanks.

35. I feel like we used valgrind, gco, and Doxygen without actually knowing how to use them well. They were just incorporated into the provided makefile, so we could just “make x” and then be done. It would be good to maybe spend half a lecture explaining what valgrind is, how it works at a high level, why it’s useful to us and c programmers in general. projects I think the projects were all good.

36. This class was excellent. I also took CS 378 over the summer, and during that course, I think one of the proctors made the mistake of letting the overall results of the Google forms be viewable. Gladly, the names weren’t viewable, but I was able to look at how much time people took to do the work. I took about 30 minutes to do the work. -- being able to see how much time people took on average gave me a good estimate of where my programming skills are. With that in mind, even though I’m done with Dowling classes, maybe somebody down the line might appreciate this sort of information.

37. I never felt like his rules were “draconian” - they were strict, but fair. I am extremely glad that I took this course and I wish I had the time to take this class for several years. Doxygen was a bit tricky to get to work at times and I didn’t feel as though I learned anything from it. Same with valgrind and gco. I ran the makefile in order to get the project points and that was it. Projects Darwin and Life were a lot of fun. Other projects felt more like software engineering projects. Speakers Bloomberg and Square were great. Twitter was a letdown - they were unprepared and made it awkward at times. Schlumberger - no strong feelings one way or the other.

38. It was hard sometimes to keep track of the lecture and take notes. Not all of the code made in class are posted on the website or they change constantly as he writes them so it’s easy to lose information. I pretty much enjoy all of the speakers with the exception of Schlumberger. I feel they were looking for engineers rather than programmers so it was kind of boring. Learning how to use Github was one of the best thing I can get from this course it was really helpful for my personal projects and other classes. However I feel I didn’t get a good understanding of the other tools. I enjoy the projects they were interesting and fun to make but it would be better to get more detailed instructions for the bigger ones except Life and Darwin.

39. I really appreciate you having us use various software engineering tools on our projects. At first I thought it was arbitrary to make us simply go through the motions of use valgrind, Doxygen, etc., but the exposure to all of these is really valuable, and I’m sure will benefit all of us in the workplace. The projects ramped up really well in terms of their difficulty levels. Collatz was a good introduction into the tools we needed to use for the projects, and each one added one or two new OOP paradigms that we learned to use now. To be honest, I did not gain a lot from the speakers. Most of them simply talked about things their company did, and occasionally tools they used, but I didn’t find them very interesting or beneficial.

40. Final was like nothing anyone expected. Very unclear about what he tests over. All in all good class.
give Exam 2 to your guest speakers, none of them would get an A hahaha. I learned a lot, thank yo
46. I enjoyed this course a lot. Nonetheless, the grading system seems to be very harsh. Other than that, I think it is a very useful
course. To be honest, Pr. Downing is the best CS instructor I have had so far. I have taken OOP courses before, but I could not
understand it fully. After taking this course, I think I can claim that I am knowledgeable in OOP. Thanks!