*** PROVISIONAL REPORT ***

UNIVERSITY OF TEXAS AT AUSTIN — COURSE-INSTRUCTOR SURVEY — Fall 2014 DEPARTMENT COPY
Downing, Glenn P C S371P 53115 — OBJECT-ORIENTED PROGRAMMING — Enrollment = 85
B100 BASIC — Surveys Returned = 74

	NUMBER CHOOSING EACH RESPONSE					NO. REPLIES THIS ITEM	AVG.
	Str Disag	Disagree	Neutral	Agree	Str Agree		
1 COURSE OBJECTIVES DEFINED-EXPLAINED	2	3	3	18	48	74	4.4
2 INSTRUCTOR PREPARED	0	0	0	9	65	74	4.9
3 COMMUNICATED INFORMATION EFFECTIVELY	1	1	2	18	52	74	4.6
4 STUDENTS ENCOURAGED-ACTIVE ROLE	0	1	2	18	52	73	4.7
5 INSTRUCTOR AVAILABILITY	1	0	3	20	50	74	4.6
6 COURSE WELL-ORGANIZED	1	3	2	17	51	74	4.5
7 STUDENT FREEDOM OF EXPRESSION	1	3	5	14	51	74	4.5
8 HELPFUL COURSE MATERIALS	3	3	13	14	41	74	4.2
9 STUDENT PERCEPTION OF AMOUNT LEARNED	3	1	2	19	49	74	4.5
	Vry Unsat	Unsat	Satisfact	Very Good	Excellent		
10 OVERALL INSTRUCTOR RATING	1	2	5	14	52	74	4.5
11 OVERALL COURSE RATING	1	7	5	20	41	74	4.3
	Excessive	High	Right	Light	Insuff		
12 STUDENT RATING OF COURSE WORKLOAD	5	30	38	0	1	74	
	Less 2.00	2.00-2.49	2.50-2.99	3.00-3.49	3.50-4.00		
13 OVERALL UT GRADE POINT AVERAGE	0	5	13	26	30	74	
	__A__	__B__	__C__	__D__	__F		
14 PROBABLE COURSE GRADE	22	23	28	1	0	74	

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.

COMMENTS:
Total Number of Comments: 46

---

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 dishonest. So I didn't learn what I came to learn, and instead walked out with knowledgeof 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. Same goes for the projects, they usually take 2 weeks when the syllabus stated they should've been graded within 1 week. One other recommendation is that the explanation of the projects take place earlier so students can have the full 2 weeks to do them. Overall the course was of great value to me, and professor downing is a 5 out of 5 lecturer.

---

3. worth

---

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 them and the instructor gave no help to review for them. 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 your knowledge to students as long as you can.

---

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 Allocator 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 compairson, 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 as a whole. On the topic of the tools we used I find the learning of github, git and valgrind to be useful but deoxygen and gtests I did not find useful and more a hassle to try to work into the projects. Out of chars

---

10. I thoroughly enjoyed 1. The allocator project - it helped me get a grasp on memory management and feel more comfortable. 2. Learning various design patterns - singleton, handle, etc. this brought discipline and structure to my code. I had onlyone issue this semester. I disagree with having a test quota based on the "average of three per 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 proyector having code all the time dose 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 notjust 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 expected to solve in the exam. Thank you for the class.

---

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 rediculous 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 explanations of the code which were not recorded or online. If the lecture's were recorded or the 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 were times through out the semester that the class was whelming. If someone had a bad week then they could fall behind. Collatz was a good project to become introduced to the class Voting good project for solving a real world problem Allocator The project needs to be better defined, as many people had difficulty understanding it Darwin and Life both great projects for incorporating Object Oriented Design Many of the talks seemed to focus on the tools they used, which is great for informing us of the many tools at our disposal. But by the end of the semester I wasn't really interested in source control X, obscure tool Y and payed service Z. Overall I enjoyed having you as a Professor and I enjoyed taking your class.

---

17. Great class. I was recently offered a full-time job after an interview in which the interviewers were extremely impressed with my familiarity with tool such as GitHub, GTest, GCov, and Doxygen. All of which I have been first exposed to in Downing's classes. I'll be forever grateful for what he does for his students, forcing us to actually use these tools and go through the processes that actual employers will expect us to follow. I always recommend to others that they take Downing classes such as SWE and OOP. The projects are fun and hammer in the concepts we covered in class. The speakers can be a bit hit and miss sometimes, particularly Schlumberger this year with their laser pointer waving all over the place and poor presentation skills.

---

18. 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 wasand what 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.

---

19. the tools git was incredibly awesome in a classroom environment, so was valgrind. I do wish there were alternatives to github, there were private projects I did not want to move over because they had issues etc and I like the github ui more than any other. gtest, gcov etc were good. I felt that 3 tests method was excessive and a lot of my time was spent writing tests, I feeel that coverage over 75 was enough to warrant a good test suite. Projects were good, I liked life and darwin specifically. I felt like this was more of a C and STL course rather than an OOP course though. Overall, I really enjoyed the course though and it taught me a lot.

---

20. I really enjoyed the class, however I do have a few complains. The daily quizes 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. Something that I really like is that code we go over is on Github for students to refer back to incase we want to play with it or try to understand how it works. Something that would be great help is that after the lecture, commented code of that wewent over in class were available for us. Sometimes, I didn't understand why things were written a certain way and having comments explaining why would have been super helpful. Obviously having comments next to the code during class would give answers away when you call on students during class, so maybe having 2 version of the same code, one commented and one not. Thats the only input I had that could make the class a better learning experience, apart from that I think its perfect !

---

22. 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 .

---

23. 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.

---

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

---

25. Dr. Downing is the Best!

---

26. Thanks Dr.Downing.

---

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 was happy for me when I showed up to his office hours announcing my new internship opportunity. He brought 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. Tools I'm very glad this course used GitHub I hadn't done much with source control, and this course took my level of comfort from "I'd rather not" to "I get it!" I'm still a bit fuzzy on Doxygen, but it's cool to know it's out there. GTest and valgrind very helpful, and I'll definitely use them on future projects. Projects I enjoyed the projects. They ramped up in difficulty at a good pace and presented different styles of problems to solve. I would have liked to have a project or two in Java, but that's mostly because I'm a lot more comfortable with Java than with C . Speakers I liked having recruiters present in class, although I was a bit underwhelmed by Schlumberger see http monoclleman.wordpress.com 2014 11 24 week-13 .

---

29. I really enjoyed the class, I felt that the material was practical and very useful in real-world applications. I'm really glad we had to use git and GitHub, I've started to used it in my personal projects! I'm also glad we had to write unit tests onthe projects, it really helped me to catch when I messed up on the project. I really liked most of the project. I had a lot of trouble with Ballot though. My only suggestion is that I sometimes would get confused on what to do on the programming projects. Sometimes certain aspects of the project were unclear to begin with and then there seemed to be a lot of confusion of Piazza it just took a lot of time to clear up some confusion on the projects. Thanks for a great semester!

---

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!

---

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 projects using Git and Github. The tools learned in the class are now a regular part of my workflow. The projects were well structured. At first I was annoyed at the long list of requirements but by the end of the course, my attention to detail improved. The speakers were one of the best parts of the class. I enjoyed some more than others, and I would have enjoyed it more if they talked more about the culture and an average day in addition to the tools they use. The two tests were challenging

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 can get out of the cramped aisles in the test rooms to ask the TAs questions, most of the time they dont even know whats expected either. For the projects, sometimes very little time is given between when they are fully explained and when they are due which can put excessive pressure on one's schedule. If they could be explained a consistent 1.5-2 weeks before the due date that would be better.

---

33. Git and valgrind are really useful, but I haven't felt any first hand experience in the importance of gcov and Doxygen. The first three projects weren't as object oriented as I'd hoped. I'm still a little fuzzy on how object should behave around each other and the readings were a difficult for me to understand. Darwin and Life did help a little bit in that regard but I still have loose ends. I did enjoy having speakers in our class, it was insightful to know how different companies work. And it was encouraging to know what they all seek in potential employees all align.

---

34. 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 all the quizzes. They were effective for motivating attendance, but I think there are better ways 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 theprojects had been more object oriented, projects 4 5 were the only ones that seemed to really benefited from oop. I didn't care for the speakers, nothing personal, I just didn't get much out of these particular speakers.

---

35. tools I feel like we used valgrind, gcov, 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 many hours people took estimated time vs. actual time . I try to go by the aphorism "know thyself" -- 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 Downing 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 wish I had the time to take more classes with Professor Downing. Tools I have been using Git and GitHub 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 gcov. 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 made but it would be better to get more detail instructions for the bigger ones ex 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 in class. 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

---

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

---

42. I took this course because I was tired of my computer science courses being overwhelmingly theoretical, and it did not disappoint in that regard. Tools I thought learning the tools we used in class was a useful exercise. My only complaint would bethat trying to come up with enough git issues was sometimes silly. But that was the only complaint. Projects I liked all of the projects. Allocator was similar to a project we did in 439H, but they were good on the whole. Speakers Bloomberg, Twitter, and Square were all very similar, but good. Schlumberger was the only disappointment, as they had no idea how to appeal to computer science students. If I changed anything about the class, I would have two or more projects instead of the tests

---

43. I didn't like how much of a role the daily quizzes play into your grade. Most classes' attendance is 10 of your grade, in here it was worth 21 because it's also a quiz, which I'm fine with except for when he asks questions the specifics the aboutthe presenters. Oh, and missing 5 quizzes automatically drops you a letter grade. If you clone directly from his repo be sure to check the filenames. IF YOU CLONE A TXT FILE IT WILL COME BACK AS A .TXT, Why does this matter? The auto-graders won't find your file, think it's missing late take 20 off your grade. Loosing 20 on a project messing up on the 1st test worth 22 brought me down to a B, but 5 quizzes later I'm at a C even though I did well on everything else.

---

44. The course was extremely useful to me because of its emphasis on getting employed. The tools used are fine. The projects, though sometimes seeming a little bit disconnected from the course material with perhaps one technique required for it to work,were very engaging. The speakers were nice, though some of them such as Schlumberger were a little unprepared. I'm not really sure if I would have preferred more lessons rather than the speakers they did provide a nice change of pace. My biggest suggestion would be to provide some sort of sample exam. I felt much more comfortable going into the second midterm because I was aware of the format. Even though we were told that the exam would be all code, seeing and actual exam always helps. Thanks!

---

45. The tests were really long and we didn't have enough time. I know that if you study enough for them you can do them however, in my opinion they don't reflect 100 your knowledge or skills in computer science. When you are working at a company youcan always work with your computer. There are also so many different languages and technologies that you can use and learn, that nobody knows them all. For this reason, I feel like studying for a test like this is like hardcoding your knowledge toknow c instead of learning and developing problem solving skills with information available to you when solving the problem. I think that if you

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!
