*** PROVISIONAL REPORT ***

UNIVERSITY OF TEXAS AT AUSTIN — COURSE-INSTRUCTOR SURVEY — Fall 2015 DEPARTMENT COPY
Downing, Glenn P — C S371P — 50890 — OBJECT-ORIENTED PROGRAMMING — Enrollment = 83
E100 EXPANDED — Surveys Returned = 77

	NUMBER CHOOSING EACH RESPONSE					NO. REPLIES THIS ITEM	AVG.
	Str Disag	Disagree	Neutral	Agree	Str Agree		
1 COURSE OBJECTIVES DEFINED-EXPLAINED	0	0	2	23	52	77	4.6
2 INSTRUCTOR PREPARED	0	0	0	13	64	77	4.8
3 COMMUNICATED INFORMATION EFFECTIVELY	0	1	1	21	54	77	4.7
4 STUDENTS ENCOURAGED-ACTIVE ROLE	0	0	2	22	53	77	4.7
5 INSTRUCTOR AVAILABILITY	0	0	6	22	49	77	4.6
6 COURSE WELL-ORGANIZED	0	4	1	20	52	77	4.6
7 STUDENT FREEDOM OF EXPRESSION	0	4	4	21	48	77	4.5
8 CLASS PARTICIPATION ENCOURAGED	0	0	2	13	62	77	4.8
9 ENGAGING INSTRUCTION	0	1	1	17	57	76	4.7
10 INST. HAD THOROUGH KNOWLEDGE OF SUBJECT	0	1	0	10	66	77	4.8
11 INSTRUCTOR EXPLANATIONS CLEAR	0	0	3	26	48	77	4.6
12 GENUINELY INTERESTED IN TEACHING COURSE	0	0	1	15	61	77	4.8
13 HELPFUL COURSE MATERIALS	2	5	20	24	26	77	3.9
14 ADEQUATE INSTRUCTIONS FOR ASSIGNMENTS	0	3	5	37	32	77	4.3
15 ASSIGNMENTS AND TESTS RETURNED PROMPTLY	1	10	9	29	28	77	3.9
16 ASSIGNMENTS USUALLY WORTHWHILE	2	0	6	29	40	77	4.4
17 STUDENT PERFORMANCE EVALUATED FAIRLY	0	4	15	38	20	77	4.0
18 STUDENT PERCEPTION OF AMOUNT LEARNED	1	1	5	25	44	76	4.4
	Vry Unsat	Unsat	Satisfact	Very Good	Excellent		
19 OVERALL INSTRUCTOR RATING	0	2	1	18	56	77	4.7
20 OVERALL COURSE RATING	1	1	10	26	39	77	4.3
	Excessive	High	Right	Light	Insuff		
21 STUDENT RATING OF COURSE WORKLOAD	3	29	45	0	0	77	
	Less 2.00	2.00-2.49	2.50-2.99	3.00-3.49	3.50-4.00		
22 OVERALL UT GRADE POINT AVERAGE	0	3	10	36	28	77	
	__A__	__B__	__C__	__D__	__F		
23 PROBABLE COURSE GRADE	21	35	18	3	0	77	

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: 49

---

1. One of the best UTCS professors I've ever had.

---

2. I very much enjoyed your class. C is becoming my go-to language because of you, your class, and Computer Graphics. One note I would like to make is that as a graduating Senior I found the coding assignments a little light. However, Project 4 and 5 were fun and satisfying to write. As well, sometimes people take too long to answer your basic questions and you should move on to someone else. Those pauses add up and I felt my attention slipping from time to time here.

---

3. talked over students who were answering questions, was somewhat rude to students who asked questions

---

4. he is awesome

---

5. I felt that becoming accustomed to the different programming tools was a great learning experience. Additionally, I appreciated that the projects stressed good object oriented design. Finally, the speakers offered a good perspective of what occurs in industry.

---

6. I found the tools used to be helpful learning experiences. I learned about issue tracking, friend testing, and a neat documentation generator. I did feel that valgrind didn't feel necessary since we were almost NEVER allowed to dynamically allocate memory. The projects covered the material well, but I felt like the last two projects had motivations or constraints that felt like they shouldn't attacked with the Object Oriented toolset. For example, Having to find weird hacky ways of getting instructions to run in Darwin when simple accessor methods would have been fine felt cumbersome. Speakers The speakers were good but I would have preferred a lecture over a skill or tool rather than industry lifestyle, e.g. a talk over regex!

---

7. 1. I really liked the tools you "strong-armed" us into using. These tools are not only used in industry, they also illustrate the general workflow that most workplaces in the real world follow. 2. The Projects were nice and gave us a good opportunity to use the tools we talked about and use what we learned in class. 3. The speakers were able to illustrate some nice points about how their workplaces function. Being able to listen to these speakers allowed us to hear about the real world without actually entering the workforce.

---

8. Good class, but I wish it was less about C Plus Plus and instead more about Object Oriented patterns, UML, etc. Making us use git, GitHub, GTest, etc is good experience and good to do. No issues. The projects are pretty reasonable and took a good amount of time. Maybe the getter setter thing could be explained more. Speakers were all good and a nice break from class.

---

9. Few thing -Please give better instructions. I understand that in the real world specs and descriptions are incomplete but it's not helpful when all I need is a 2 sentence answer and I need to spend 20 minutes on piazza hoping someone asked the samequestion -Although you put the code up on the website the parts you don't put up on the text editor are sometimes crucial and really should be put on the site -The organization is kinda all over the place. Hell it took 3 4 of the semester to get to inheritance. This seems like a c class rather than an OOP class. -I don't think I ever looked at the results of gcov -I never looked at doxygen -Travis is meaningless if we ran make test locally -No getters and setters is a loose definition

---

10. My only complaint was a rare personal issue, in that despite a series of unfortunate events during the first two weeks of class, I was not given an extension on the first assignment. If I had, I would probably have an A in this course easily, but due to that fiasco I have been struggling to make a B. That does annoy me quite a lot, but other than that I'm glad I took the course. The course itself has been an invaluable learning experience for programming in c as well as Java, and Professor Downing teaches the material in an incredibly human-friendly way. He's an excellent professor. Just don't let anything go terribly wrong outside of your control when you're in his course, and you'll do fine. Regardless, you'll learn a lot.

---

11. If the exercises available to us had comments of why something was wrong or why something is right that would be helpful.

---

12. Professor Downing is one of the best CS professors I have taken at UT. He genuinely seems interested in making sure his students learn. The Projects were rather difficult, but they were all doable in the time we were given to complete them. Downing's random calls on students in class forced me to pay more attention in his classes so I knew what I was talking about in case I was called upon. Overall, the course was great and Downing was excellent.

---

13. The class was enjoyable. Things I liked gtest, gcov, valgrind, learning c , travis-ci, github, google submission forms, tophat. Things I disliked trying out different tech in my humble opinion too often that it would disrupt flow that was already set. Having to setup our files for project submissions was a great experience, but a little more coverage on how it all works together would give a greater vector to explore and learn more. Also, some discussion on what gtest options are available and doing similar with the other tech at our disposal would facilitate greater incentive to learn more. Using piazza as discussion was great and helped me a lot. Overall great course, wish there were more like it.

---

14. Professor Downing is one of my favorite Professors at UT. Had him for software engineering and for OOP now. He really knows the material well, engages students in class, and quizzes every day to test the students knowledge of the last lecture. I feel like the tests are weighted too heavily though.RightnowI did extremely poorly on the first test and need to do well on the second test to just pass the class. As of now my quiz average is about a 75 and up to project 4 grade average is like 97.5. So I feel like since the projects are where the experience and a lot of learning comes from, and since the time required on them is so high they should have a higher weight. The tests are two hours each but can decide more than the projects if you pass

---

15. You did an amazing job teaching this course. At first the course seemed intimidating but I eventually found myself comfortable and engaged in the material. I feel like a much stronger programmer thanks to your class. Only a couple of things may improve the course in my opinion. 1. Add a little bit more slack points or drops for students who miss class for interviews. It can really make a big impact on their grades if they miss quizzes. Missing a class can also impact their performance on the next quiz. I just say this in the context that some companies can't schedule their interviews that fit with student's schedules better. 2. Having a small list that summarizes the topics of the class may make it easier to prepare for quizzes. Thanks!

---

16. Great class overall. I wish the class had been more about object oriented design patterns using C Plus Plus, instead of how to implement object oriented code in C Plus Plus. Not a big fan of using tophat, especially when we switched to using canvas for quizzes later in the semester. I wonder why the ubiquitous iclicker isn't used for this course, practically everyone already has

one, and it doesn't require a subscription - at least for the students. Overall I liked the class, am looking forward to other classes taught by Downing in the future.

---

17. The biggest problem I have with this class is the exam and quiz weight. In my opinion, they are just way too high for a class called OOP. Downing is excellent and I really learned a lot in this course however, my grade does not reflect how much I learned, and it really defeats the purpose of a "grade." If we have to write code for the exam, why not give me a compiler also? After all, it's about understanding the concept and problem solving, not being perfect. To make this class better, please consider drop the exam and quiz weight down to 40 total. That's right, just 40 . Then, please try to add at least one more project for the students, ideally two additional projects. They are fun and are the reason why we are in CS. RunningOutOfChar..

---

18. I think that this may be Dr. Downing's technique to teach, but it seeemed that after every project was due, his following lecture would teach us something that would be extremely useful for the project we had just turned in. It may be to give us a good learning experience but I think he should have that lecture before the project is due. I'm sure he has a good reason for it but just to get us to figure it out on our own without such a nice coding technique he shows us right after doesn't feel quite right. Also, grading on projects is a little tough when certain things can get you an automatic zero even if work was done. No partial credit on hours of work seemed unfair. Nonetheless, it was still a great course by a great professor.

---

19. I enjoyed the class very much. I think some of the grading on assignments is a bit ridiculous. TAs are much to quick to mark things as late for silly mistakes. It is infuriating that the punishments is so high in that regard. Another improvement which would be nice would be more materials, and by that I mean more sample code that covers the same material, but it is not covered in class. For instance, writing an FI.h and FI.c instead of just the Range.h and Range.c . Overall, I really enjoyedthe course. Downing is an amazing teacher, and I wish some of the grading scheme was not so stressful quizzes and late points on projects . I had a great time, in the end.

---

20. One of the best professors I've had in UTCS. I will attempt to take any course Professor Downing teaches while I am still here.

---

21. Professor Downing is a master at his craft. I can't wait to take his software engineering course next semester. I care more about the fact that I learned so many useful things more than my grades, which personally matters more to me. While in class he has a lot of energy and makes jokes, in his office hours he's incredibly kind and willing to listen to you. Keep him for as long as you can!

---

22. Downing is an excellent professor who teaches in a really engaging way. However, the course material is not as engaging, and it seems like we spend a lot of time learning c instead of OOP. For example, lambda expressions, pointers, and references,are covered much more than abstraction and inheritance. If I wanted to learn C , I would take a C class. If I want to learn OOP, I should learn about OOP. That being said, this class is excellent if you know neither and want to learn both. The assignments were a drag too. No setters getters only made it harder, and the last two projects were extremely similar. The first projects were mostly tackling the software instead of the actual problem, and they were unnecessarily tiresome as a result.

---

23. Getting Travis CI to work was annoying a lot of times and I don't think the TA's had a lot of experience in Travis either. But other than that, the course was well organized and TA's were helpful most of the time.

---

24. The first time I used GitHub is in this class, I found it very helpful during the partner project. And Travis-Ci will tell you whether you are missing a file or not, which is extreme helpful before turing the project. I consider the Darwin the most interesting project among all 5 of them.

---

25. I am very impressed with Glenn Downing's teaching style and ability to engage the class with difficult concepts. I like that his projects have us use tools that actual software companies use, such as version control and a CI service. I would have preferred Bitbucket over Github, since it allows an unlimited number of private repos no educational account required and still uses git. I think the requirements on the projects to use a coverage tool and provide documentation are good, but I would emphasize code coverage over quantity when it comes to unit tests. I would have liked a little more guidance on the 4th and 5th projects with regards to Cpp OOP design, such as .h versus .cpp files as well as making classes with no getters or setters.

---

26. Fantastic course! I think it should be split into two courses Advanced C and Object Oriented Programming. Teach the C requirement in another course because that's what this course mostly was or just teach OOP in Java which is more OOPy anyway . The tool requirement was great, I actually wish it changed a bit from project to project to give me more exposure. Projects were nice but did not focus on OOP very much especially the first three . Speakers gave a nice change of pace. For large chunks of each I was bored, however, because I am a senior and they described generic working environment or what SCRUM is. My biggest complaint is the quizzes having no replacement. I interviewed many times and took zeroes ' which is unfair D

---

27. The class had a little bit too much focus on random parts of C in my opinion, instead of focusing on large-scale programming. But a lot of the stuff we learned is absolutely necessary, including the tools we used. It's a good taste of industry programming.

---

28. GIt is great but using travis Ci sometimes cause some version trouble which means that programs run well locally but might fail there Projects are all good Speakers are all good.

---

29. I was surprised when the classroom time was used less for OOP and more for good programming practices and learning the ins and outs of C , but I really enjoyed it. For the most part, I enjoyed the projects and felt that they required an appropriateamount of time to complete well. The only thing that I wasn't as fond of was that I could do perfectly or near-perfectly on all of the projects, but still get a C in the class for a low test grade on the very challenging tests. I think the projects should count for just a little bit more, or the test just a little bit less.

---

30. I think that there should be more resources on learning the c language itself, such as online materials and references the textbook didn't really cover the language that much, and the code posted online sometimes was not enough to really learn the language . Also, I thought that we can focus less on just c and more on the object oriented part of the cass.

---

31. Overall, the class was very well designed. Every day is accounted for on the calendar so I knew what to expect. I liked learning about tools that professionals use and being able to use them makes me feel better about my skills. The projects were fun and challenged me and I was able to see some of the concepts we learned in class in action. I have no complaints about those. I liked the speakers they gave insight into the skills of a real programmer and made me more informed about the different kinds of computer science fields out there. There were nothing that I really disliked about this class.

---

32. Overall the course provided a very good introduction to C and the lower level workings of object-oriented programming. The

projects looked fairly basic on the surface but when working within the restrictions, they all had their own challenges thatreally helped to further understand of the material. The gtest tools were very useful for setting up consistent and readable test files for all projects. One negative about the course was that the turn in restrictions seemed very strict. Losing 20 of a project grade for having an incorrect git SHA or not up to dat git log despite the main part of a project working perfectly seemed harsh. This could maybe be remedied by having a smaller penalty or slip days to catch mistakes.

---

33. I think the 3 unit test per function is a good idea, but can be kind of ridiculous ie if we overload a function write small functions, we are penalized , and I think it encourages writing huge functions for the sake of fewer tests instead of writing nice, modularized code. I think there's too much emphasis on the quizzes, and not enough on the actual projects which is what the class should be oriented .

---

34. I didn't like how the ideas for the projects were taught after the assignments were due, I feel like it would've been a better learning experience to understand what the teacher does in class and then being able to implement what we learned on our projects rather than reflecting on what we should had done after the project was over

---

35. Really great class format, I like that he calls on people individually. It makes you stay on your toes, and makes you want to show you were paying attention as well as make class feel more involved when he gets in a one-on-one conversation with you.The assignments were interesting and the right amount of work, the first assignment was especially useful to introducing all the tools to me. I also liked the quizzes, and how they reward you for paying attention. I do feel a little unprepared when it came to the test and I had to write everything out. It was more me needing to get used to writing everything out more than whether or not I understood the concepts. I think maybe the tests could have a small section that have quiz-like questions.

---

36. I appreciate the clear grading and quizzes to keep us on track This class has a very clear grading scheme and rubric for assignments I prefer canvas quizzes because we don't have to pay for extra materials

---

37. Downing seems like a good guy but to be blunt this course was a hot mess. Look, the first 2 3 of the semester had NOTHING TO DO with object oriented programming. It was a class on C data structures and writing STL clones for more than HALF THE COURSE. Downing, you have an obligation as an honest teacher to not call this course OOP in the future if you are not going to change the content. This is how your Software Dev. class was as well you taught us essentially nothing that actually qualifies as "Software Development." I could not in good conscience recommend this class to people without giving them a 5 minute explanation of your teaching quirks and the actual content of the class. Please be a good teacher and change some shit up.

---

38. the tools i felt that the usage of these tools were a bit contrived, but they weren't so inconvenient that i felt they were unnecessary and ruined the development experience. the projects i enjoyed doing the projects, but i wish some things were more explicitly stated in the grading rubric, specifically the deep copying. not having skeleton files for P2 was a little intimidating for me, since i wasn't comfortable at all with c still. the speakers to be completely honest, the main thing i enjoyed about having a speaker was having a day off. most of the things that the speakers talked about i already knew, like recruiting, workflow in the real world, etc. the only one i really didn't find boring was the JPL talk.

---

39. I enjoyed using the tools. My only issue was that we weren't really forced to learn how to use them. For every single project I just copied the makefile and the.travis.yml file and replaced the word collatz with whatever project name. If you asked me to make one myself I would have no idea what to do. Especially for the makefiles, it would be a useful thing to learn to make yourself. The projects were all bueno, don't have much to say about them The speakers were nice, but it's more interesting to hear the talks like the one with the guy that did the mars rover thing and made a 3D model of the rover's images. It's interesting to hear what neat things they're working on rather than just hear about the tools they use and their workflow.

---

40. Absolutely taking Downing's other class. I've learned more actual programming skills this semester than the last two years.

---

41. Grading for the projects try to prioritize "completeness" so much that just for missing one small piece of the project, it can be evaluated as "incomplete" and students are forced to turn it in later.. deducting them 20 of the grade instead of the little proportion that the initially missing piece was worth. This doesn't seem like a fair evaluation of the amount of work put into the project at all.

---

42. Being able to talk about the Google Unit Testing suite and Travis CI and Valgrind have been effective during recruiting. However, I feel like for the scope of these projects some of these tools may be unnecessary. A lot of what we do in this class imitates working in the industry in the industry we utilize these tools because of how large the scope of projects can be. In this class, projects are much smaller and are pair programming projects at the most. I think these tools can become much more effective if we worked in larger groups and on larger projects - pintOS sized projects people would kill me for saying this . But because our projects aren't so large, they can just become "paper pushing", or "busy work" that accompanies a project.

---

43. I learned a lot in this course, both about C , OOP concepts, and being a better software developer. Having to use Git, Travis CI, and Doxygen helped cement good habits for me as a programmer, though having to hit 15 issues for each project was a chore at times. Projects were graded fairly, and were challenging, without being excessive. The 11 days or so for each project flew by quickly, and starting late was a recipe for disaster and stress. The in class quizzes were quite difficult as a whole. They never felt unfair, but it was very easy to lose points on them. Neither of the two tests were unfair, but both of them felt like they tested certain topics extensively, while leaving other topics untouched.

---

44. I found your teaching style to be frightening at times, but a good and necessary experience for us who typically do not speak up in class. 1. The tools were good to have experience with. Some of them, like gcov, I'm not sure were fully utilized. 2. No problems with projects. 3. I appreciated when the companies discussed different aspects of their work. Having different groups discuss a variety of topics, such as open source and sprint environment, makes them more than a long advertisement.

---

45. Projects were challenging but good, though I feel that some of the requirements and penalties, were a little severe. 20 percent for 1 day late minor mistakes in submission details seems to be too much. Partial credit seemed to be lacking as well.

---

46. It would be helpful to have a visual of the concepts we cover in class, like a powerpoint to refer to later, or some more detailed comments in the posted code. We go over the whys when we look at and write code in class, but it was hard to recall later on. Online references helped some, but didn't always clarify everything. It was difficult to jot down notes and pay attention at the same time since people generally write slower than we speak. I also think that depending on who students had for 312 and 314, and how much previous exposure to programming concepts and practice greatly affected the difficulty level of the course for each student.

---

47. I found the exams were way longer than I would be able to finish in the allotted time. I studied thoroughly, reviewing all

lecture notes, creating an extensive note sheet, reviewing all the topics on the list, and re-implementing the code from class. Despite this, I found that I would really struggle to finish on time simply because I did not have enough time to think through problems, and this would severely hurt my grade. Alternatively, if tests are long, it would be nice if they stayed closer to the topics covered in class, so that way we don't have to spend as long formulating a solution. Other than that, I found that this was an excellent class. The projects in particular were fun and informative learning experiences.

---

48. Assignments Only complaint is why measure by number of tests and not test coverage? Tools Doxygen is probably the most unclear of them Attendance Maybe leave it running the entire class, or through the quiz, for people that show up 5 minutes late. Very Specific Feedback for Presenters You seem to care something about women in CS due to your articles and the events on the calender - why not bring in women speakers, or minority speakers? All the speakers, if I remember right, were just white men usually 20sish too and diversifying the speakers could be one of the most influential ways to empower women and minorities in CS.

---

49. I generally disliked using the Doxygen and html files because I had no idea why we were even using the tools in the first place or what use they had in the task at hand. All of the other tools we used, particularly GitHub, felt appropriate for the task at hand. I also wish the tests covered a more expansive area of what we had gone over in class. It sort of felt like a class that covered all of American history giving a test about a single decade.
