*** PROVISIONAL REPORT ***

UNIVERSITY OF TEXAS AT AUSTIN — COURSE-INSTRUCTOR SURVEY — Spring 2015 DEPARTMENT COPY
Downing, Glenn P — C S371P — 51992 — OBJECT-ORIENTED PROGRAMMING — Enrollment = 55
B100 BASIC — Surveys Returned = 49

	NUMBER CHOOSING EACH RESPONSE					NO. REPLIES THIS ITEM	AVG.
	Str Disag	Disagree	Neutral	Agree	Str Agree		
1 COURSE OBJECTIVES DEFINED-EXPLAINED	0	1	2	11	35	49	4.6
2 INSTRUCTOR PREPARED	0	0	0	4	45	49	4.9
3 COMMUNICATED INFORMATION EFFECTIVELY	0	0	0	8	41	49	4.8
4 STUDENTS ENCOURAGED-ACTIVE ROLE	0	0	1	11	36	48	4.7
5 INSTRUCTOR AVAILABILITY	0	0	1	11	37	49	4.7
6 COURSE WELL-ORGANIZED	0	1	2	9	37	49	4.7
7 STUDENT FREEDOM OF EXPRESSION	0	2	2	13	32	49	4.5
8 HELPFUL COURSE MATERIALS	0	2	9	13	25	49	4.2
9 STUDENT PERCEPTION OF AMOUNT LEARNED	0	1	1	10	37	49	4.7
	Vry Unsat	Unsat	Satisfact	Very Good	Excellent		
10 OVERALL INSTRUCTOR RATING	0	0	1	9	37	47	4.8
11 OVERALL COURSE RATING	0	1	3	15	30	49	4.5
	Excessive	High	Right	Light	Insuff		
12 STUDENT RATING OF COURSE WORKLOAD	0	16	33	0	0	49	
	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	1	5	19	24	49	
	A	B	C	D	F		
14 PROBABLE COURSE GRADE	13	25	7	2	2	49	

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: 40

---

1. I thought the first exam was a little bit unreasonable in that we had to know the STL list's methods to implement several solutions. For example, I thought that there was a method for adding an element to the front and another to add to the back, but couldn't remember the syntax. I wasted a lot of time trying to worry about the correct syntax for something that takes less than 15 seconds to google. It's fine to expect students to have that sort of thing memorized, but I feel like it could have been communicated that we'd have to know those methods beforehand, since it wasn't something we had covered at all in class.

---

2. I really enjoyed this course. The ins and outs of C were covered extremely well. However, I do wish there was more of an object oriented aspect. The first two projects didn't really demand much in the way of object oriented programming, though I understand it's partially necessary since most of us didn't know C well back then.

---

3. Git GitHub - Awesome. GTest - Good, though I don't know much about any other testing systems. Valgrind - I could read it and see we had no memory leaks, which was nice. gcov - Harder to understand than Valgrind, but useful. Doxygen - Starting up was a bit hard, but writing documentation is something that needs to be done more in the industry. The projects were all good and worthwhile. However, why did we wait until the 4th project to use objects? Square talk was good. Mutual Mobile was more boring - I didn't care for too many details about the tools they use. High level overview would be better. Overall, excellent course. Downing really knows his stuff, and I learned a ton.

---

4. Downing is a professor like no other. He is so knowledgeable of the subject, and communicates so well. He got me to think deeply about programming concepts I had never considered. I love his style of picking students to answer questions. I also likethe fact that there's a quiz every day. It keeps information fresh. The projects were interesting and challenging, but I found that sometimes useful concepts were discussed after the project was due, or the day before. I'm really not a fan of the all-or-nothing style of grading for projects 1 2. I spent a lot of time on project 2 but still got a zero because of a small issue. This class is also very difficult. It's no surprise that by the end of the semester almost half the class had dropped.

---

5. I feel like lecture focuses too much on details and not enough on things like good OOP design. Overall the class, while well defined, seems to kind of meander though various c concepts without a super ordered motivation for the material.

---

6. I liked all of the tools used in class. I think the first project, Collatz, could be made into something harder i.e. the same difficulty as the other 4 . I liked the extra credit opportunities. Also, I think the guest speakers were great, but I would keep it to two or less per semester more would take too many of Prof Downing's great lectures away .

---

7. I saw this mentioned in previous evaluations, but it is worth noting again, this did not feel like an OOP class, it felt like a Cpp class. Now while I came in expecting that, I'm sure others did not and were disappointed, especially considering Cpp isn't even mentioned in the course description. ---Projects felt almost completely removed from the lectures which themselves felt removed from the readings, though there was some convergence near the end when OOP material finally started to show upin class and in Life Darwin . I thought you were a spectacular lecturer, and I really enjoyed the question answer style of teaching, though it was certainly aggravating when you called on someone who seemed like they just didn't have a clue what was

---

8. Wonderful class and loved the exposure to tools such as git and TravisCI, the hardest part of the course must have been the proper file naming. Maybe tune down the god damns a little bit

---

9. One of the best courses I've taken at UT. Wish the tests were curved most of the class is left writing things down until the very last second. Github, gcov, gtest, and valgrind were nice. Doxygen not so much, documentation looked ugly. Projects were challenging but fair. Would say you need to have prior cpp experience in order to get an A.

---

10. Really enjoyed being randomly called on. Really enjoyed the daily quizzes.

---

11. You are highly recommended for a reason, I enjoyed your class a lot and your teaching methods. Asking students to answer questions kept us engaged and thinking which even as students we don't do enough of. The only regrets I have taking this class are my performance on the first test and forgetting reading assignments for quizzes. I don't blame you or your teaching methods just my poor test taking skills. If I'd prefer a change it would be including slip days, but maybe that's because I've gotten so used to them even though in reality I rarely use them. I hope to take SWE sometime in the following semesters.

---

12. This class was incredibly useful. I enjoyed the structure of the class. Pseudo-mandatory attendance was able to keep me engaged over what we covered in class. I have definitely become a better "programmer" after this class. The only thing I would want changed about this class is that we would begin talking about the "core" object oriented concepts sooner in the semester.

---

13. I enjoyed his energetic tone. At least I won't fell asleep in class.

---

14. I really like this course and I think it will be much useful for either academic purpose or industry preparation.

---

15. I enjoyed the various tools you had us use. I felt that having a required number of tests is a little silly, as in my opinion testing is more about covering all possible branches of your code. However,I understand that you need something more concrete for grading purposes. In terms of the projects, I felt they were all pretty good. My least favorite was life, but in the end I enjoyed it more as a design problem than the actual coding portion. Overall, I feel like the class was more of a learn cplusplus class as opposed to an OOP class except for the readings , and something like a design patterns in C or even an advanced data structures class would be more fitting for your teaching style.

---

16. Some things to consider Loosen up on the number of unit tests requirements and the Github issues requirements. While these things are great, having an absolute stance on the numbers seems kind of ridiculous and inflexible. While the startup speakers were fine, I think bring them in earlier in the class, by the time they spoke, opportunities for internships had already passed January and February are really the latest times for being able to get a summer internship . Things that made the class great however, are your comprehensive knowledge of C and design principles. But also your willingness to engage with the students outside of class, and your efforts for us to seek knowledge beyond the classroom. Keep it up!

---

17. My main gripe with the course was the ambiguity of the material in regards to quizzes and tests, certainly more so with tests. All we knew for the test is that there would be coding on it. Even after pouring a great deal of care into learning the details

of in class classes and how to write them, the classes on the test felt like a curve ball. The quizzes certainly are more straight forward. You'll be quizzed on something you learned in the past couple of classes. But I think we could benefitfrom getting a few bullet points of review and preparation. There's a quiz every class. It's about making sure people are keeping up with the material. That should be made as easy and as clear as possible.

---

18. I feel I learned more C rather than object oriented programming.

---

19. This is exactly the kind of course I was looking for in the UTCS department. Really nitty gritty details of C plus plus with comparisons to Java and Python. Great code examples to study, definitely going to hide away a clone of the code samples for future reference. Recommended to all of my friends entering CS at UT.

---

20. The tools I had already had a limited amount of experience with git and GitHub from CS 314, but the required use of these tools in OOP really made me appreciate them more and now I don't feel like I could live without source control. I had no experience with unit testing before OOP, and using GUnit I think was the second most valuable tool used in this class. Gcov was really nice just for giving me an idea of what constitutes good test coverage, and valgrind actually uncovered memory leaks that I would've otherwise been oblivious too. Overall, I feel like I am a much better programmer for having taken this course, and I actually wouldn't change a thing. It'd be nice to have discussion sections, but perhaps that is impossible.

---

21. The test format seems more akin to competitive programming based on the sheer volume of code expected of us, which doesn't mesh well with my thinking, coding, and learning style. Otherwise I loved the course and switched to pass fail because I felt I was learning a lot. Thank you very much.

---

22. Didn't sign up for an in depth analysis of Cpp I was hoping for more Object oriented design. It would be helpful to learn inheritance before the Darwin project. Also, doxygen, 20 git issues, and gtesting was unhelpful work. The gtests implementation"punishes" the use of more methods with more busy work. Unless you code the projects by guessing you don't need to test every simple function.

---

23. - The use of Github was extremely valuable to me and is a transferable skill into the real world. I previously was unaware of the issue tracking feature and am glad to have learned about it. GTest has also been a great tool to learn about and practice with although I do believe there should be a maximum number of unit tests to write. Too many tests will push students away from understanding the value. Valgrind, gcov, and Doxygen are tools that I feel indifferent about. - I enjoyed all of the projects and thought that they were an acceptable level of difficulty and amount of work, while also allowing me to learn a great deal. They were also graded fairly. - First test was really hard, maybe too hard.

---

24. If you wrote a textbook or print text materials for the course i feel like they would be so useful i would buy them in a heartbeat. The course is so dense but manageable and all the information taught is so useful. The textbook and the comments on the github slides dont help nearly as much as the actual lecture so i think a tertiary material would be worthwhile.

---

25. Course seems to be graded very tough for a CS course. I did learn a lot however and loved all the speakers that came to class. I feel most of the quizzes were trying to trick us more than teach us same with the tests and this leads to the very toughgrading. This class is one were you have to look at the exact wording to see the teachers intention.

---

26. Not enough time on exams. Testing on UVA was kind of tedious because there weren't really any helpful error messages. Besides that, the other projects were well defined and the class was very informative.

---

27. This class was organized really well. Prof Downing was able to keep me engaged for the whole hour and that's what I think, aside from the material itself, made this course enjoyable and made me feel like I've come away with a lot. The projects were a good extension of what was going on in class. Overall I enjoyed this class very much and look forward to taking any other course Prof Downing might offer in the future.

---

28. I liked the speakers. The quizzes were a little difficult.

---

29. Great course, very little was ambiguous. The quizzes everyday seemed difficult to get high grades on because nearly everything from the previous class needed to be memorized, while this seems fair, my learning is strongly enhanced by repetition so Ifrequently did poorly.

---

30. The readings on Thursdays were a little excessive, and the quizzes were at times too involved. Otherwise, the class was quite well done.

---

31. This course was excellent! I learned a lot and I feel like this course is and will be very useful in the future and I thank you for that. The only problem I really had was the first assignment, Collatz, where I was trying to learn all the tools, Doxygen and GitHub. I think the grading was a little harsh especially since you're doing that project on your own and might not know all the tools used. Still, i think it's a great class and definitely recommend it to others.

---

32. While the majority of the course was well organized, I feel that the midterm was mostly irrelevant to the material we learned in class. The instructions on the midterm were vague and unclear. Please improve the midterm to follow the course material and to accurately evaluate the students.

---

33. When I signed up I thought this was going to be an OOP class, not a C class. While I understand C is OO the class focused on the nuances of the language rather than OOP design. I would have liked to design more containers and talked about why wewant OOP abstractions etc. The time alloted for quizzes was rather short most of the time. Having to look up at a screen and understand what is going on so that you could answer the question was difficult. Most of the time I got the answers wrong were not because I did not understand the material, but I didn't have enough time to read and understand the question. There were several times I was ignored during a test where I needed clarification.

---

34. I enjoyed this class a great deal, in general. I liked the programs and found them to be the right balance of challenging but not overly time consuming. The biggest, and only real, problem I had with this course was the exam. I did not have nearlyenough time to work and only completed 4 of the 6 questions. This may have been a problem strictly limited to me but the few other people I spoke with about the exam said that they felt rushed for time as well. I feel that exam should have warranted at least 2 hours. Other than that I loved the class. Interesting subject matter and Professor Downing does a great job of teaching the material in a helpful and fun manner. Maybe try to get Gcov and valgrind used more, I only used them on collatz.

---

35. A few minor gripes about the class - I didn't like the quiz requirement, just personally not a fan of required attendance, and

the first half of the class or so dragged on a bit, having come in with basic knowledge of Cpp. The tools seemed excessive for Collatz and Voting but reasonable for the other projects. Some of the requirements seemed somewhat excessive, e.g. why 20 issues and why not just 10, why 3 tests per function. Would have liked to see more done with Valgrind, gcov, and Doxygen. Thought the test requirement should based on code coverage, e.g. at least 75 percent lines executed for tests. For me, already having a job lined up, I personally didn't find the speakers too useful, though I can see them being valuable to others.

---

36. Overall I thought the class was very good! The only thing that I thought was "meh" was the ability to ask questions in class. I often found it really hard to ask questions because I would have my hand up for a long time but because someone is alwaysbeing called on, it made it hard to get a question in. Also, I kind of wish the class talked a bit more about design patterns and more OOP like program designs and a bit less on the nuances of C .

---

37. Downing does an excellent job of teaching this course and has a passion for the subject. This maks learning the material much better. I wish there was something I could think of to improve the course but I am afraid not.

---

38. My favorite class this semester! I think the tools GitHub, gtests, etc. were really helpful, and many I wouldn't have learned on my own without this prompt. Speakers were all very good. Often, you don't notice hands probably due to the layout lighting of the room - next semester, you might let students call out instead of raising their hands. I like the format of having github code and quizzes on the website instead of traditional slides - the context helps a lot. The extra credit opportunities were great! I particularly enjoyed the extra UVa project. Have a good summer!

---

39. Great class, great lecturer. Really felt like I learned the intricacies of C and how to write idiomatic C . Wish we spent more time learning good OOP design and some time on how to best organize large projects.

---

40. I really enjoyed this course and definitely think the daily quizzes were the best way to get good attendance. I felt though at least the first test haven't taken the second test yet was very ambiguous and the teacher should have reviewed the test format before hand like maybe an example question and how he wants us to answer it. I think Downing is a great teacher and definitely cares about his students progress in the class, and although the material wasn't that difficult, Downing definitely makes earning a good grade in the class hard. Projects were easy but time consuming if you did them as best as possible and followed what he taught you. I think a better layout for the class would be a project every week instead of 5 with 2 tests.
