COURSE SUMMARY REPORT
Numeric Responses

University of Washington, Bothell
Sci, Tech, Engr. & Math
Science, Tech, Engr. & Math
Term: Spring 2018

CSS 342 A
Data Structures, Algorithms, And Discrete Mathematics I
Course type: Face-to-Face

Evaluation Delivery: Online
Evaluation Form: D
Responses: 30/37 (81% very high)

Taught by: Yusuf Pisan
Instructor Evaluated: Yusuf Pisan-Lecturer

Overall Summative Rating represents the combined responses of students to the four global summative items and is presented to provide an overall index of the class's quality:

Median	College Decile
4.1	4
(0=lowest; 5=highest)	(0=lowest; 9=highest)

Challenge and Engagement Index (CEI) combines student responses to several *IASystem* items relating to how academically challenging students found the course to be and how engaged they were:

CEI: 5.3
(1=lowest; 7=highest)

SUMMATIVE ITEMS

	N	Excellent (5)	Very Good (4)	Good (3)	Fair (2)	Poor (1)	Very Poor (0)	Median	DECILE RANK Inst	College
The course as a whole was:	30	33%	33%	30%	3%			4.0	4	5
The course content was:	30	43%	27%	27%	3%			4.2	5	6
The instructor's contribution to the course was:	30	40%	33%	27%				4.2	3	4
The instructor's effectiveness in teaching the subject matter was:	30	37%	33%	23%	7%			4.1	3	4

STUDENT ENGAGEMENT

Relative to other college courses you have taken:	N	Much Higher (7)	(6)	(5)	Average (4)	(3)	(2)	Much Lower (1)	Median	DECILE RANK Inst	College
Do you expect your grade in this course to be:	30	10%	20%	30%	10%	13%	10%	7%	4.8	2	3
The intellectual challenge presented was:	30	37%	27%	23%	7%	3%	3%		6.0	7	6
The amount of effort you put into this course was:	30	37%	17%	33%	7%	3%	3%		5.7	4	4
The amount of effort to succeed in this course was:	30	33%	27%	27%	10%		3%		5.9	5	5
Your involvement in course (doing assignments, attending classes, etc.) was:	30	43%	13%	20%	13%	10%			6.0	6	6

On average, how many hours per week have you spent on this course, including attending classes, doing readings, reviewing notes, writing papers and any other course related work?

Class median: 12.0 Hours per credit: 2.4 (N=30)

Under 2	2-3	4-5	6-7	8-9	10-11	12-13	14-15	16-17	18-19	20-21	22 or more
	3%	3%	13%	17%	7%	27%	7%	20%	3%		

From the total average hours above, how many do you consider were valuable in advancing your education?

Class median: 8.4 Hours per credit: 1.7 (N=30)

Under 2	2-3	4-5	6-7	8-9	10-11	12-13	14-15	16-17	18-19	20-21	22 or more
	10%	13%	17%	23%	10%	7%	10%	10%			

What grade do you expect in this course?

Class median: 3.5 (N=30)

A (3.9-4.0)	A- (3.5-3.8)	B+ (3.2-3.4)	B (2.9-3.1)	B- (2.5-2.8)	C+ (2.2-2.4)	C (1.9-2.1)	C- (1.5-1.8)	D+ (1.2-1.4)	D (0.9-1.1)	D- (0.7-0.8)	E (0.0)	Pass	Credit	No Credit
13%	37%	13%	23%	7%	3%	3%								

In regard to your academic program, is this course best described as:

(N=30)

In your major	A core/distribution requirement	An elective	In your minor	A program requirement	Other
67%	23%		3%	7%	

STANDARD FORMATIVE ITEMS

									DECILE RANK	
	N	Excellent (5)	Very Good (4)	Good (3)	Fair (2)	Poor (1)	Very Poor (0)	Median	Inst	College
Course organization was:	30	30%	37%	30%	3%			4.0	3	4
Sequential presentation of concepts was:	30	27%	43%	30%				4.0	3	4
Explanations by instructor were:	30	37%	33%	27%	3%			4.1	4	4
Instructor's ability to present alternative explanations when needed was:	30	33%	37%	27%	3%			4.0	3	4
Instructor's use of examples and illustrations was:	30	40%	33%	27%				4.2	3	4
Quality of questions or problems raised by the instructor was:	30	30%	50%	13%	3%	3%		4.1	3	4
Contribution of assignments to understanding course content was:	30	40%	27%	23%	7%	3%		4.1	4	4
Instructor's enthusiasm was:	30	63%	17%	20%				4.7	5	5
Instructor's ability to deal with student difficulties was:	30	43%	20%	37%				4.2	4	4
Answers to student questions were:	30	37%	30%	33%				4.1	3	4
Availability of extra help when needed was:	30	27%	40%	23%	10%			3.9	2	2
Use of class time was:	30	37%	40%	13%	7%	3%		4.2	4	5
Instructor's interest in whether students learned was:	30	53%	33%	10%	3%			4.6	5	5
Amount you learned in the course was:	30	47%	23%	27%	3%			4.4	5	6
Relevance and usefulness of course content were:	30	63%	23%	13%				4.7	7	7
Evaluative and grading techniques (tests, papers, projects, etc.) were:	30	40%	37%	17%	3%	3%		4.2	4	5
Reasonableness of assigned work was:	30	43%	27%	23%	7%			4.2	4	5
Clarity of student responsibilities and requirements was:	30	43%	27%	20%	10%			4.2	4	5

CSS 342 A
Data Structures, Algorithms, And Discrete Mathematics I
Course type: Face-to-Face

Evaluation Delivery: Online
Evaluation Form: D
Responses: 30/37 (81% very high)

Taught by: Yusuf Pisan
Instructor Evaluated: Yusuf Pisan-Lecturer

STANDARD OPEN-ENDED QUESTIONS

Was this class intellectually stimulating? Did it stretch your thinking? Why or why not?

1. not really, the pace of the course moves really fast and is intense so that you haven't even gotten to understand one concept completely before the next concept is introduced.

2. I felt some of the stuff was already covered in 142 and 143

3. Yes, it was very tough, but very interesting. The projects given for the class were difficult, but helped me improve drastically.

4. Yes, it was very interesting and helpful.

5. Yes, difficult assignments as usual for CS classes. Many new/unfamiliar concepts.

6. It's challenging because the assignments were big

7. This class had me thinking a lot, especially when there are many different solutions for any given problem.

8. Yes it made me think more about the question and the instructor taught it very well. I am able to think freely and go through the homework while over coming difficult parts.

9. The last 3 programs were very challenging and increased my programming skills immensely.

10. Overall I thought that the class was easier than I originally thought it would be. A lot of the topics were repeats from the Java courses just in C++. Since both languages are object-oriented the implementations were fairly similar. Pointers and references were the only things that I thought were intellectually stimulating.

11. Yes, because it is expanding my knowledge in coding.

12. Absolutely. I was a bit scared to start C++ due to what I had heard about 342, but it really gave me a new perspective on programming and how different languages suit different situations. I also found that the way logic was presented in the course was extremely interesting.

13. Yes.

14. Yes

15. Yes, it stretched my thinking for sure.

16. Yes, it was.

18. Sort of. This is not a fair question for me to answer because I have been developing for 23 years professionally. It did however give me an excellent refresh of my C++ skills

19. Yes, the material was very good in stimulating thinking

20. Yes, very challenge and stretch my thinking.

21. Yes, the class assignments challenged us to apply all the coding information we learned into a working program. Assignment 5 was especially challenging in putting together the program design and making the classes work together.

22. Yes, learned new concepts and challenging ways of applying them

23. Yes, the class brought up concepts that we're very interesting and they related to what I plan on doing in the future.

24. I've taken the data structures at UW Seattle which I preferred much more than this one. The only new content that I learned was discrete mathematics.

26. This class was stimulating. It presented problems that required critical thinking skills.

27. yes, it was very intellectually engaging. There were many complex ideas presented that seemed easy to understand, but once you got into it, you found it to be much more difficult to actually implement the ideas.

What aspects of this class contributed most to your learning?

1. The assignments

2. The lectures.

3. The projects and class lectures

4. The in class examples

5. I would say the lectures contributed the most as the content was solid and well-explained. The assignments were also great for applying what I learned.

6. Assignments

7. One thing that helped the most was the in-class coding from the Professor, as well as the many examples that had the entire class involved.

8. The lectures were very helpful and that homeworks were due every two weeks made them more manageable. Instead of having smaller assignment every week I think it should always be every two weeks for the homework assignments.

9. I enjoyed the in class examples as well as the programming assignments. The Slack community also helped increase my learning.

10. Lectures definitely helped more than the book. Sometimes it seemed that we lingered on topics for longer than necessary though. The slide shows themselves I thought were only OK but sufficient enough to still be useful. Slack was helpful in one instance but not enough to significantly affect my grade in the class.

11. In-class coding examples on an IDE.

12. The lectures were very good, the assignments were okay, but my favorite bits were the practice quizzes and practice midterm/final. Most of the practice questions for mid/final were really fun to solve, and felt like I was learning quite a bit while doing them.

13. Yusuf's use of examples and in class demonstrations were key. He changed about half way through the quarter and started doing this more and it really helped move my grade in the right direction.

14. Learning new things

15. Assignments

16. The in-class work was very helpful, as it allowed me to see where i needed help before i started the homework.

18. The assignments, they were excellent representations of the work

19. Homework assignments were very good and helped me understand the material very well.

20. Almost everything

21. The in class exercises presented small problems that the entire class solved. The assignments presented larger coding problems that allowed us to apply the knowledge learned in class lectures.

22. Textbook, assignments, in class exercises

23. In class activities were the most helpful at learning the concepts of the class.

24. The class format was relatively effective, although not as much as Seattle's course. I was pushed to ask questions which pushed for activity. Yet the content was often not very stimulating.

26. The lectures were the best part. He covers material that is on the final and is relevant. He answers questions thoroughly and with examples if needed.

27. working with the code, general review was helpful, but I wanted to work with code more in class.

What aspects of this class detracted from your learning?

1. The time. There wasn't enough time to go over everything, this class realy shouldn't be just two time a week.

2. Nothing much

3. None

4. Setting up a program should have been done before class so we didn't have to spend 5-10 minutes of setting up code already given by the problem.

5. I was not very motivated to read the textbook, in part because reading assignments were not particularly enforced.

6. N/A

7. N/A

8. Office hours could be better and when I go it sometimes it was useful sometimes it wasn't.

10. Not knowing when certain chapters should have been read before class.

11. Randomly calling on students and sometimes lectures are dry.

12. I felt like the exams were weighted a bit too high, but this is common with CSS courses. The problem is that I get testing anxiety and seem to do marginally worse during tests. That being said, I left the midterm feeling confident but was met with a grade that dropped me to an immediate B in the class. This was incredibly disheartening and I wish that it felt more related to the effort that we're putting in.

13. Some aspects like UNIX and other extras were difficult to use without more guidance and did nothing but add time to assignments. If given more help, it would greatly aid the student's learning and interest.

14. The use of class time, espcially in the first half of the class was pretty poor. A lot of it seemed like unncessary review.

15. quizzes

18. Only myself due to the fact I have lots of experience already

19. Midterm material different very slightly from class material.

20. N/A

21. The first draft of assignment specifications was sometimes confusing. Learning the C++ language presented an extra challenge toward the beginning of the class. Lecture slides sometimes had mistakes in the code or other content.

22. Tangents

23. Nothing comes to mind.

24. How content was conveyed in slides was not as clear and concise as I wanted. Finding content split between personal website and canvas was very irritating.

26. The lack of homework. It threw me off tempo so I tended to forget quizzes or assignments.

27. only looking at a general view of coding until working on the assignment.

What suggestions do you have for improving the class?

1. Make the tests easier and more comparable to what we have learned in class. provide solutions to study guides and overall go over concepts as though students are not familiar with concepts.

2. Nothing comes to mind at the moment.

3. More office hours

4. Give answers to the practice test!!!! It's very hard to study when you're not sure what is correct or incorrect. Put under the resources how to run cpplint and cppcheck (or whatever it's called) so we can use tets our code with it.

5. I like the way the class is, but I would perhaps like more clear reading schedules, perhaps in the form of actual "assignments" that show up in the To Do list on Canvas.

6. Better time management and just one less assignment

7. The last assignment in the class felt very rushed, since we only had one week to complete a project. It felt like the project needed another week to finish. Other than that, everything is fine.

8. The linux lab part of the homework should be reviewed in class so we know how to clearly do it and what the error messages mean when you submit your assignment and get errors in the linux lab. (Maybe go over just the most common ones).

9. No suggestions

10. Having a stricter calendar for the reading, possibly posting them as empty assignments on canvas so that students can have a schedule of when readings need to be done.

11. More active learning like the sort functions with decks of cards.

12. I would suggest to have a second midterm, or a small tweak to the scaling for exams. The problem is that I feel so mixed - I've enjoyed this course more than ANY other programming class, and probably more than any other course I've taken at UW thus far - but I don't even know if I'll pass the class. I also feel like I've learned more than other classes, but it's not enough.

13. Same as above, more guidance toward out of class responsibilities and techniques would help.

14. Pacing. The first half of the class was pretty slow and then things ramped up in the second class.

18. None, it's a well formatted class

19. Study guides closer to class material

20. N/A

21. Fix the slides before the lecture so that information is not confusing, and finalize assignment specifications before letting students take a look at them to reduce the need to redo code or solutions to the problem.

22. Clearer, less verbose explanations

23. Have a couple more smaller assignments in the class so students can practice programming more often.

24. I am going to continue to compare it to Seattle's format. As such https://courses.cs.washington.edu/courses/cse373/18wi/ was the one I took. If a teacher is going to use their own site, they should format it properly to clearly illustrate what I may need to do or use as well as including tutorials for transitioning to new IDEs or formats or languages. We do it in class already, so why not include an online step-by-step guide as well?

25. Have assignments due weekly that cover more class content, instead of semi weekly assignments, that cover less of the learned material. I felt that by the time of the midterm some concepts were not covered enough in assignments and were more dependent on how much you absorbed from the lectures and the readings. More hands on experience with the material would have helped.

26. Potentially force readings of the book, so the students have to understand the reading material. Although it is not necessary as the lectures cover the material.

27. I would love to have practice sets to work on inside of class, or at least outside of class on my own. Maybe even offer small amounts of extra credit for them. Being able to work hands on with the code has been the best way of learning for me. Getting a general idea only takes a short amount of time as compared to actually figuring out how to get it to work in the compiler. The in class time was well spent and learning how to do things with just pencil and paper is a great way to learn as well, I would just like to be able to work with the code a bit more before having to implement it in an assignment. I think a mix of 60-40 would be great for lecture and questions on pencil and paper, with the other 40 being spend on a computer where the students can ask questions and mess around with the code. I'm aware there was also another class you could take that let you work with code and ask questions, but I couldn't make it due to work, so having practice sets and videos online would be very helpful, even if they just mirrored what was taught in the other class.

INSTRUCTOR-ADDED OPEN-ENDED QUESTIONS

What advice would you give to a student taking this course that will help them succeed?

1. Program as much as possible, and look to other third party resources as well

2. Start assignments early and test them often.

3. Do your reading before class and work on your project long before the deadline.

4. Take 332 with this. Start weeks early on a project.

5. Get started on the interfaces and design of your projects early so you can focus on the implementation for the rest of the time.

6. Start early, do not procrastinate.

7. Attend all the classes, do a lot of programming outside of class to brush up.

8. Start homework assignment early so you don't stress out later. Even though they are due every two week they are bigger assignment and you should at least have an outline the second day it comes out.

9. Start programs early! Ask and answer questions on Slack. Go to office hours if you have questions.

10. Attend the lectures, it is super useful. Even if you don't post or answer questions on Slack it is often useful to see where other students are having problems so keep Slack on your phone always.

11. If the lectures are not going to be on Panopto, record them. Sometimes handwriting notes and code is more beneficial.

12. Study, do questions on paper and on a whiteboard, do the practice exams. Be ready for fairly hard grading on the exams.

13. Time management, for sure.

14. Start the assignments early

15. start the assignments early

17. Make sure to READ THE TEXTBOOK! It is essential to passing the class!

18. Take the C++ primer course that is offered to help ramp up if you haven't taken a C++ course.

19. As long as you understand basic OOP principles, you'll do just fine.

20. Start your work early, read the book,

21. Always ask the instructor if anything is confusing. Take extra time toward the beginning or before taking the class to get familiar with the C++ language.

22. Practice

23. c++ isn't as difficult as it seems at the beginning. There are a lot of similarities between c++ and java/c# so don't let c++ scare you away from the class.

24. cplusplus.com, geeksforgeeks, c++ documentation, stackoverflow were all helpful. Using a lot of cout is very useful for pinpointing where code may break that may be difficult to reach in debug.

25. Practice the material on your own, because you might not have practice with assignments before a test that covers the material comes around.

26. Read the book. I did not and I had difficulty during the quizzes. Other than that, attend class and you will succeed.

27. Study the textbook, don't wait to start on assignments, and get help form the instructor and other students. The more help you have, the better.

IASystem Course Summary Reports summarize student ratings of a particular course or combination of courses. They provide a rich perspective on student views by reporting responses in three ways: as frequency distributions, average ratings, and either comparative or adjusted ratings. Remember in interpreting results that it is important to keep in mind the number of students who evaluated the course relative to the total course enrollment as shown on the upper right-hand corner of the report.

Frequency distributions. The percentage of students who selected each response choice is displayed for each item. Percentages are based on the number of students who answered the respective item rather than the number of students who evaluated the course because individual item response is optional.

Median ratings. *IASystem* reports average ratings in the form of item medians. Although means are a more familiar type of average than medians, they are less accurate in summarizing student ratings. This is because ratings distributions tend to be strongly skewed. That is, most of the ratings are at the high end of the scale and trail off to the low end.

The median indicates the point on the rating scale at which half of the students selected higher ratings, and half selected lower. Medians are computed to one decimal place by interpolation.[1] In general, higher medians reflect more favorable ratings. To interpret median ratings, compare the value of each median to the respective response scale: *Very Poor, Poor, Fair, Good, Very Good, Excellent (0-5); Never/None/Much Lower, About Half/Average, Always/Great/Much Higher (1-7); Slight, Moderate, Considerable, Extensive (1-4).*

Comparative ratings. *IASystem* provides a normative comparison for each item by reporting the decile rank of the item median. Decile ranks compare the median rating of a particular item to ratings of the same item over the previous two academic years in all classes at the institution and within the college, school, or division. Decile ranks are shown only for items with sufficient normative data.

Decile ranks range from 0 (lowest) to 9 (highest). For all items, higher medians yield higher decile ranks. The 0 decile rank indicates an item median in the lowest 10% of all scores. A decile rank of 1 indicates a median above the bottom 10% and below the top 80%. A decile rank of 9 indicates a median in the top 10% of all scores. Because average ratings tend to be high, a rating of "good" or "average" may have a low decile rank.

Adjusted ratings. Research has shown that student ratings may be somewhat influenced by factors such as class size, expected grade, and reason for enrollment. To correct for this, *IASystem* reports **adjusted medians** for summative items (items #1-4 and their combined global rating) based on regression analyses of ratings over the previous two academic years in all classes at the respective institution. If large classes at the institution tend to be rated lower than small classes, for example, the adjusted medians for large classes will be slightly higher than their unadjusted medians.

When adjusted ratings are displayed for summative items, **relative rank** is displayed for the more specific (formative) items. Rankings serve as a guide in directing instructional improvement efforts. The top ranked items (1, 2, 3, etc.) represent areas that are going well from a student perspective; whereas the bottom ranked items (18, 17, 16, etc.) represent areas in which the instructor may want to make changes. Relative ranks are computed by first standardizing each item (subtracting the overall institutional average from the item rating for the particular course, then dividing by the standard deviation of the ratings across all courses) and then ranking those standardized scores.

Challenge and Engagement Index (CEI). Several *IASystem* items ask students how academically challenging they found the course to be. *IASystem* calculates the average of these items and reports them as a single index. *The Challenge and Engagement Index (CEI)* correlates only modestly with the global rating (median of items 1-4).

Optional Items. Student responses to instructor-supplied items are summarized at the end of the evaluation report. Median responses should be interpreted in light of the specific item text and response scale used (response values 1-6 on paper evaluation forms).

[1] For the specific method, see, for example, Guilford, J.P. (1965). Fundamental statistics in psychology and education. New York: McGraw-Hill Book Company, pp. 49-53.