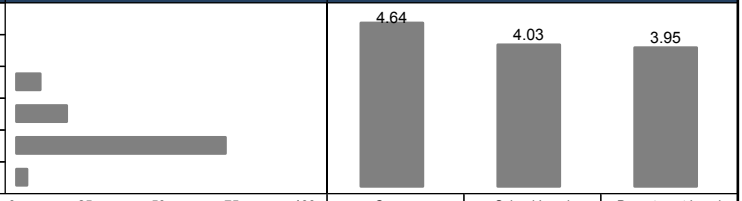
JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2015.Fall

Course: EN.600.120.04.FA15: Intermediate Programming

Instructor: Yair Amir

1 - The overall quality of this course is:

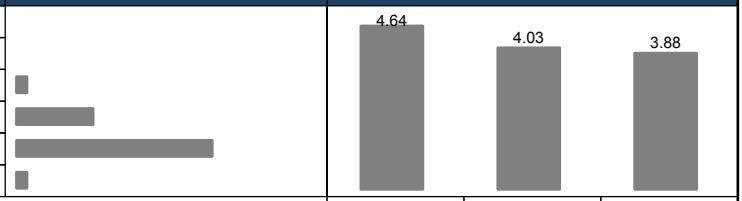
Response Option	Weight	Frequency	Percent
Poor	(1)	0	0%
Weak	(2)	0	0%
Satisfactory	(3)	2	8.7%
Good	(4)	4	17.39%
Excellent	(5)	16	69.57%
N/A	(0)	1	4.35%



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
23/24 (95.83%)	4.64	0.66	5.00	8,249	4.03	0.96	4.00	1,266	3.95	1.00	4.00

2 - The instructor's teaching effectiveness is: Yair Amir

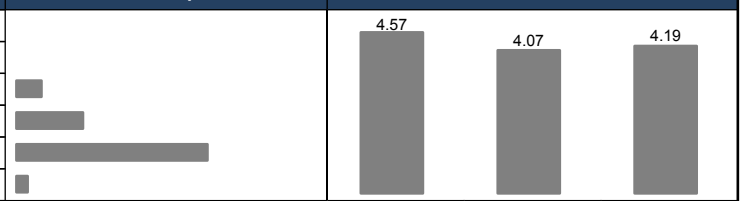
Response Option	Weight	Frequency	Percent
Poor	(1)	0	0%
Weak	(2)	0	0%
Satisfactory	(3)	1	4.35%
Good	(4)	6	26.09%
Excellent	(5)	15	65.22%
N/A	(0)	1	4.35%



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
23/24 (95.83%)	4.64	0.58	5.00	9,351	4.03	1.03	4.00	1,303	3.88	1.09	4.00

3 - The intellectual challenge of this course is:

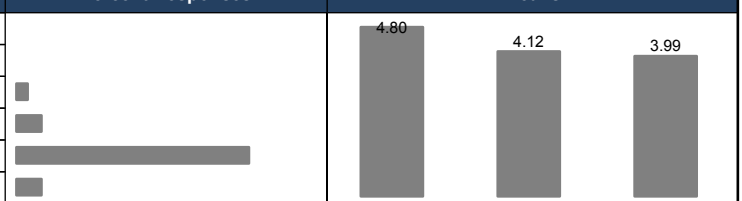
Response Option	Weight	Frequency	Percent
Poor	(1)	0	0%
Weak	(2)	0	0%
Satisfactory	(3)	2	9.09%
Good	(4)	5	22.73%
Excellent	(5)	14	63.64%
N/A	(0)	1	4.55%



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
22/24 (91.67%)	4.57	0.68	5.00	8,205	4.07	0.93	4.00	1,256	4.19	0.87	4.00

4 - The teaching assistant for this course is:

Response Option	Weight	Frequency	Percent
Poor	(1)	0	0%
Weak	(2)	0	0%
Satisfactory	(3)	1	4.55%
Good	(4)	2	9.09%
Excellent	(5)	17	77.27%
N/A	(0)	2	9.09%



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
22/24 (91.67%)	4.80	0.52	5.00	8,205	4.12	1.00	4.00	1,257	3.99	1.11	4.00

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2015.Fall

Course: EN.600.120.04.FA15: Intermediate Programming

Instructor: Yair Amir

5 - Please enter the name of the TA you evaluated in question 4:

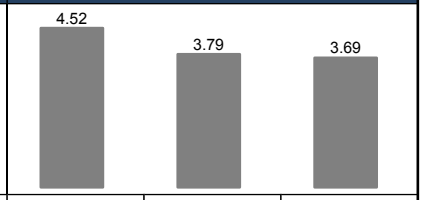
Return Rate	18/24 (75%)

- Emily
- Emily & CAs
- Amy Babay
- Emily
- Elizabeth
- Emily
- Amy Babay, Emily Wagner and others
- Emily, Amy
- Emily Wagner
- All of them
- Everyone
- Emily
- Amy Babay
- All of them, there were so many!
- Emily
- Amy Babay
- Amy Babay
- Emily Wagner

6 - Feedback on my work for this course is useful:

Response Option	Weight	Frequency	Percent	Percent Responses
Disagree strongly	(1)	0	0%	
Disagree somewhat	(2)	0	0%	
Neither agree nor disagree	(3)	2	9.09%	
Agree somewhat	(4)	6	27.27%	
Agree strongly	(5)	13	59.09%	
N/A	(0)	1	4.55%	

0 25 50 75 100

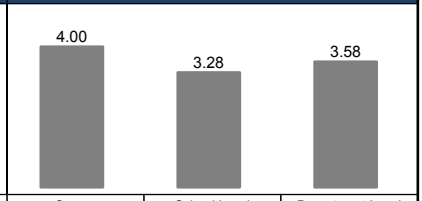


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
22/24 (91.67%)	4.52	0.68	5.00	8,190	3.79	1.09	4.00	1,257	3.69	1.13	4.00

7 - Compared to other Hopkins courses at this level, the workload for this course is:

Response Option	Weight	Frequency	Percent	Percent Responses
Much lighter	(1)	0	0%	
Somewhat lighter	(2)	0	0%	
Typical	(3)	5	22.73%	
Somewhat heavier	(4)	12	54.55%	
Much heavier	(5)	5	22.73%	
N/A	(0)	0	0%	

0 25 50 75 100



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
22/24 (91.67%)	4.00	0.69	4.00	8,199	3.28	1.07	3.00	1,255	3.58	1.09	4.00

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2015.Fall

Course: EN.600.120.04.FA15: Intermediate Programming

Instructor: Yair Amir

8 - What are the best aspects of this course?	
Return Rate	16/24 (66.67%)

- The many aspects of programming covered in this class and the enthusiastic and friendly professor/TA/CAs are the best aspects of this course.
- Highly interactive and feedback-based learning, and a good mix of lectures and working time in each class. The assignments were fair overall and interesting to do. The large number of TAs present was very helpful throughout the course. A dry-run before the midterm provided good preparation. Hearing about the professor's research in DSN and connections to the material covered in class was valuable and definitely interesting.
- projects
- The format of assignments allow students to learn the concepts and give them enough time to master the material
- The coding tutorials that we had were the best aspect of the course
- The best aspect of the course is the projects. The projects were effective in solidifying our understanding of the materials taught, while at the same time allowed us to build something of our own from scratch. I liked how the both quality and effectiveness of the code are emphasized throughout the projects; I think Dr.Amir really gave us a taste of how programming is like in the real world instead of simply teaching us the language. The TAs also gave extremely helpful feedbacks, which I think is crucial in helping us improve throughout the semester.
- The cohesiveness of the projects, they build on each other in a logical progression.
- Fun language to learn, class is very rewarding.
- Programming is fun, and the way it was taught was very good.
- I really enjoyed that the projects had some room for creativity and intellectual challenge. The print outs of the code that we went over in class was really helpful
- The professor is very dedicated to teaching the course materials and helping students understand the concepts and structures behind every assignment. The course has many CAs and TAs that work around the clock to help students with coding assignments or answer any questions. Assignments were not handed out on a weekly basis, but dependent on the concepts the student learned. The projects were therefore larger and extensive as a result, but I think that it was very appropriate and well balanced for this course. Instead of worrying about an assignment every work, students were able to apply all concepts into a bigger project. Through this method, I found it to be very effective in learning the material. Work days were also very beneficial since students could spend the entire class period with several TAs and CAs answering their questions and helping them with their code. Having printouts of the code while the instructor is going over it makes it very easy to mark and make notes on it.
- The best aspect of this course was the actual making of something useful, as opposed to the typical studying for Hopkins courses.
- We go in more depth than other classes might
- C is somewhat fun.
- The professor is probably the best I had in Hopkins. He talked to every single student one on one throughout the course. There are no paper exam. He really taught me how to code.
- The helpfulness of the TAs.

9 - What are the worst aspects of this course?	
Return Rate	13/24 (54.17%)

- The projects were very large and time-consuming, and there were only 5 throughout the semester. I don't think this needs to be changed, but the deadlines were a fairly large burden. Thankfully, the instructors were all understanding and extended some deadlines.
- professor's accent
- Some of the assignments were quite difficult
- There were none
- Seg faults lol
- Difficult at first and long projects take lots of time
- There was nothing bad about it.
- Nothing I can think of.
- Learning both C and C++ in one semester seems to be very condensed sometimes. Especially when C++ was taught. There was so much code examples that it was hard to keep up let alone read through the entire text. It felt rushed when Project 4 came along and I had a hard time coding the project. Conceptually, it was very well taught and I understood what the assignments required. However, rushing through C++ code still left me unfamiliar with the syntax and made it harder to implement.
- The deadlines were to short, they were almost always pushed back though. I believe the professor and his TA's know the deadlines were too short.
- Grading is very harsh, no leniency on the midterm (since a simple mistake like an extra character can segfault the program entirely, and no time to fix)
- C++ is not fun.
- Although you can finish most of the work in class, it is still somewhat time consuming. But time well spent.

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2015.Fall

Course: EN.600.120.04.FA15: Intermediate Programming

Instructor: Yair Amir

10 - What would most improve this class?	
Return Rate	13/24 (54.17%)

- Nothing comes to mind - overall I really enjoyed the class.
- 1. Slow the course pace. 2. The projects are very difficult for people without any experience. Try to make the projects more accessible.
- -
- The class is good as it is in my opinion
- I would like to see maybe one or two demonstrations of entire projects in class instead of small chunks of programs. Maybe a data structure someone else has created and we can probe around, as a class, the source code a little to see how it actually works. I believe being able to read and understand someone else's code is very important in helping us to learn a language. Many times when I code, I would try to recall an instance of how a similar piece of code I've seen before functions, so having a database of that knowledge would be helpful. Also in class the suggestion to have people shifting one spot to the right to read another person's code could be helpful as well.
- Projects with less time commitment
- Nothing I can think of, maybe give more explicit examples/have a handout that had results from running sample programs.
- N/A I thought this was an awesome class.
- When going over the code in class, a slower pace might help students find the location in the file easier. I tend to use the handout since writing in pencil can make quicker edits than going through vim with arrows keys to add comments. Putting line numbers on the handouts would help immensely in locating the code, especially when it comes to large chunks of code.
- Having outlines of subjects covered to accompany the packets of code given in each week.
- More, smaller assignments so that we can have 'checkpoints' and not be overwhelmed by huge projects. Make design due before project maybe
- More short assignments, and less super long and intensive assignments.
- Ah it's pretty great already.

11 - What should prospective students know about this course before enrolling? (You may comment on any aspect of this course such as assumed background, readings, grading systems, and so on.)	
Return Rate	13/24 (54.17%)

- Prior knowledge of using bash is useful.
- The course is comprised of a small number of large projects, which require planning ahead and a significant time commitment. There are lots of opportunities for one-on-one help which make the work much more manageable.
- This course is very project-oriented. If you cannot devote a lot of your time on this course, you probably should not take it or you'll only learn a few if not nothing.
- Background in programming is assumed but the new information is taught well
- The course requires a lot of effort. There is a lot of great material to learn and only with enough effort that can be achieved. Also, if you realistically have the time to take the course instead of auditing, take it because you can learn much more that way.
- Spend sometime learning the unix environment and the debugging systems beforehand so one can be more efficient when coding. Also spend more time on design and don't always try to compile after fixing a small problem. Everytime one edits the code, one should have the mindset of trying to make it work rather than hoping that one small fix will miraculously fix the whole thing. And also don't be afraid to cut away huge chunks of code.
- Projects will take a long time
- It's a fun course!
- use the office hours to get questions answered. They really do want to help
- Be prepared to work hard in this course. This course is manageable as long as you dedicate enough time and effort into it.
- This is unlike the Intermediate Programming course from 2014. according to my friend. This is a course which demands working on projects on one's own for the entire time.
- Difficult, a lot of work debugging. Professor is very stubborn, but just listen to what he says
- go to class. The professor requires attendance and trust me it's worth it.

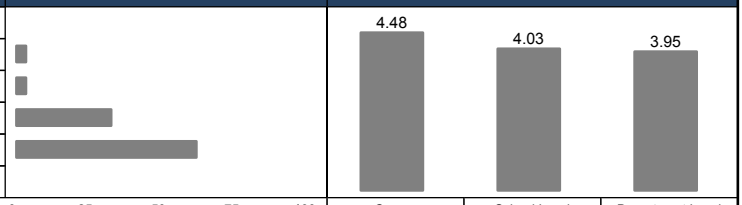
JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2015.Fall

Course: EN.600.120.03.FA15: Intermediate Programming

Instructor: Yair Amir

1 - The overall quality of this course is:

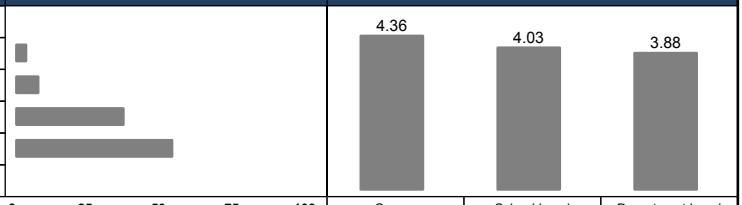
Response Option	Weight	Frequency	Percent
Poor	(1)	0	0%
Weak	(2)	1	4%
Satisfactory	(3)	1	4%
Good	(4)	8	32%
Excellent	(5)	15	60%
N/A	(0)	0	0%



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/26 (96.15%)	4.48	0.77	5.00	8,249	4.03	0.96	4.00	1,266	3.95	1.00	4.00

2 - The instructor's teaching effectiveness is: Yair Amir

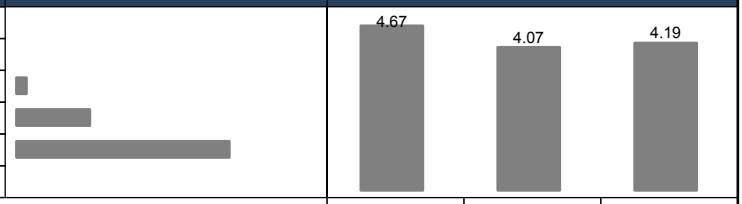
Response Option	Weight	Frequency	Percent
Poor	(1)	0	0%
Weak	(2)	1	4%
Satisfactory	(3)	2	8%
Good	(4)	9	36%
Excellent	(5)	13	52%
N/A	(0)	0	0%



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/26 (96.15%)	4.36	0.81	5.00	9,351	4.03	1.03	4.00	1,303	3.88	1.09	4.00

3 - The intellectual challenge of this course is:

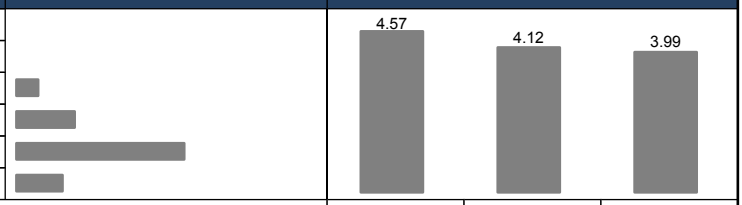
Response Option	Weight	Frequency	Percent
Poor	(1)	0	0%
Weak	(2)	0	0%
Satisfactory	(3)	1	4.17%
Good	(4)	6	25%
Excellent	(5)	17	70.83%
N/A	(0)	0	0%



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
24/26 (92.31%)	4.67	0.56	5.00	8,205	4.07	0.93	4.00	1,256	4.19	0.87	4.00

4 - The teaching assistant for this course is:

Response Option	Weight	Frequency	Percent
Poor	(1)	0	0%
Weak	(2)	0	0%
Satisfactory	(3)	2	8%
Good	(4)	5	20%
Excellent	(5)	14	56%
N/A	(0)	4	16%



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/26 (96.15%)	4.57	0.68	5.00	8,205	4.12	1.00	4.00	1,257	3.99	1.11	4.00

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2015.Fall

Course: EN.600.120.03.FA15: Intermediate Programming

Instructor: Yair Amir

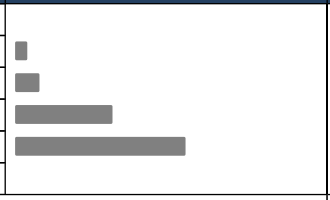
5 - Please enter the name of the TA you evaluated in question 4:

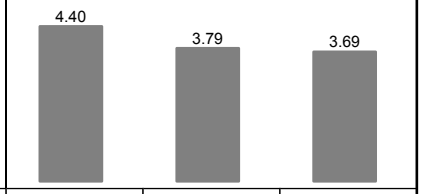
Return Rate	18/26 (69.23%)

- Emily Wagner
- Emily Wagner
- Emily Wagner
- Emily and Amy
- Amy
- Emily Wagner
- Amy
- N/a
- All of them
- 3
- Emily Wagner
- Emily Wagner
- Amy Babay
- Emily
- Emily
- Amy Babay
- Amy
- Emily

6 - Feedback on my work for this course is useful:

Response Option	Weight	Frequency	Percent
Disagree strongly	(1)	0	0%
Disagree somewhat	(2)	1	4%
Neither agree nor disagree	(3)	2	8%
Agree somewhat	(4)	8	32%
Agree strongly	(5)	14	56%
N/A	(0)	0	0%

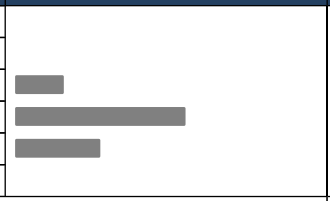


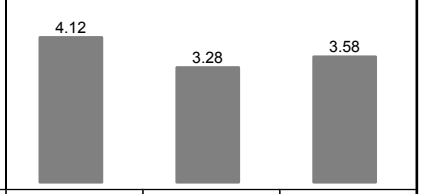


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/26 (96.15%)	4.40	0.82	5.00	8,190	3.79	1.09	4.00	1,257	3.69	1.13	4.00

7 - Compared to other Hopkins courses at this level, the workload for this course is:

Response Option	Weight	Frequency	Percent
Much lighter	(1)	0	0%
Somewhat lighter	(2)	0	0%
Typical	(3)	4	16%
Somewhat heavier	(4)	14	56%
Much heavier	(5)	7	28%
N/A	(0)	0	0%





Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/26 (96.15%)	4.12	0.67	4.00	8,199	3.28	1.07	3.00	1,255	3.58	1.09	4.00

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2015.Fall

Course: EN.600.120.03.FA15: Intermediate Programming

Instructor: Yair Amir

8 - What are the best aspects of this course?

Return Rate	22/26 (84.62%)

- The projects are very helpful in understanding material. The professors, TA, and CAs are engaged and committed to making sure we excel in the class.
- The projects are always very interesting and at the same time very doable. They are the best part of the class and makes you learn new things about programming.
- - Instructors designate ample amounts of class time for allowing students to work on/helping students with projects - Instructors, teaching assistant, and course assistants are available to help students plan their designs for projects - Project feedback is extremely thorough and helpful
- The focus on data structures really pushes you to think about what you're doing before you do it. I think a lot of people learned how important it is to design things before you code so that your structure doesn't fall apart, and I don't think that was emphasized a lot in other CS classes I've taken. In addition, Dr. Amir and the TA/CAs are all very helpful. Dr. Amir does a great job of explaining everything thoroughly and Amy is fantastic at helping students to debug their programs. Also, the CS-120help email is really helpful for questions while working on the projects!
- Yair Amir demonstrates passion not only for the subject, but for his whole field. Being taught by someone who shows love for the material is motivating. The tasks were very creative, fun to program, and interesting.
- Several projects instead of weekly assignments
- The engaging and excited professors and staff who are so willing to help students succeed.
- The class is organized very well in helping me understand stuff
- Professor Yair genuinely cares for and wants his students to succeed to the best of their abilities. The CA's were extremely helpful as well giving on point feedback and helping whenever needed.
- The projects covered the material in the course very well. The TA's were excellent and were great at taking time to explain confusing concepts.
- The programs are very challenging but also very interesting.
- The CAs, TAs, and teachers are always helpful when you have a bug in your project., and have awesome personalities. When grading projects and the midterm, They take a lot of time and effort to find out what is wrong with your code and give you feedback on how to fix it, as well as improve its readability and efficiency.
- Teaches you how to program well
- The best aspect of the course were the projects, which trained us to develop larger and more complex programs, while also introducing us to new data structures and their practical applications.
- Challenging, learn a lot
- Yair and Amy really wanted us to learn and their enthusiasm made the class better
- Yair Amir
- The course provides an introduction to two very important programming languages and an introduction to some difficult aspects of programming.
- Professor Amir, Amy, and the rest of the CS120 staff are really excellent at providing help and knowledge whenever you may need it. If you're engaged and present, they will be too.
- Meaningful, helpful feedback
- Detailed feedback, intellectually demanding, focused on building from essentials. practice quickly implementing linked lists.
- I think the practice on programming to clean up code and create a design document really helped to tackle larger problems effectively as well as emphasis on proper time management.

9 - What are the worst aspects of this course?

Return Rate	20/26 (76.92%)

- The lectures sometimes get to the point of boredom by being too long and drawn out.
- The lack of curve in this class is a bit discouraging, too few people get As in this class.
- I think this course was very well put together—the only thing is that Dr. Amir can get a little impatient sometimes. He's an incredible professor and I've learned a lot from him, but I feel that many students are intimidated by his tone when he responds to questions that he might not think are worth his time.
- The knowledge level of the students in the beginning of the course is very varied, which may make those who are advanced feel that the content covered in the course is somewhat shallow.
- Instructor doesn't know C++ well Too much weight is given on solving problems the "right way" than on having valid solutions Assignments are only explained in class, and handouts basically say refer to what we did in class
- NA
- I did not dislike any aspect of this course; however, the worst part (although not terrible) was Yair's accent. At times his words would slur into other words, making "bits" sound like "beats" for example. Also sometimes the assignment handouts were written obscurely, meaning they were difficult to understand what exactly the assignment was asking from the student. In particular as well, the course asked students to use programs that weren't sufficiently explained. It took me weeks to figure out what gdb and valgrind were capable of, despite the one class devoted to them.
- When we initially learned concepts, it was at times difficult to understand. A lot of understanding came when we did the projects. At times, more at the beginning, it was difficult to understand instructions.
- Still don't understand destructors.
- They grade pretty harshly.
- The way class was taught. Spent too much time literally reading over code rather than doing smaller practice examples and then it culminating in a large project.
- I wish there were more projects of a higher difficulty, like Project 3 and the Final Project.
- CAs/office hours/emails are not always helpful
- I struggled to both learn the language syntax AND the technicalities of the project together in one project
- Class was hard to pay attention to
- There was a disconnect between what was taught in class and what one was expected to do in assignments. Grading seemed rather arbitrary and unnecessarily harsh. Assignments were extremely large and sometimes there wasn't a whole lot of time to complete them. The professor wasn't very understanding of those who struggled and seemed to deride those who had questions during lectures. The professor, TA, and course assistants frequently offered to answer questions and made themselves quite accessible, but when asking questions, they would generally turn the question back on you (like "I don't know, you tell me.") without giving any answer or assistance. Sometimes the work being done in assignments didn't seem entirely applicable or relevant.
- It's a fair amount of work and if not managed properly can become too much to handle.
- Time consuming projects
- discouraged from asking questions
- I felt that some of the in class lessons in the programming knowledge was a little bit repetitive on topics that we needed to know coming into the class (such as pointers and classes)

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2015.Fall

Course: EN.600.120.03.FA15: Intermediate Programming

Instructor: Yair Amir

10 - What would most improve this class?

Return Rate	20/26 (76.92%)

- More coding to learn concepts than listening to lectures.
- Clearing up what the rubric for the class is could be helpful to motivate students, also, the curve should probably increase.
- N/A; I think this class was already very well put together.
- A group task could be fun, although it would probably escape's Professor Amir's goal: to learn how to program by oneself by the end of the semester. However, if there is one thing that the course is missing is making the students open up and do some talking between themselves. Excluding a few people, I finished the semester without knowing the rest of my class. It could be a good idea to email students some (not all) fragments of code beforehand, so the class can be used to program and get help on current projects.
- For C++, explain the abstract concepts instead of just showing examples. List full assignment on handouts, rather than "implement the structure we went over in class"
- Nothing.
- clearer instructions on assignments.
- At the beginning of the course, spend more time covering project instructions. At the beginning of the course, since I did not have that much project experience, it was hard for me to follow what I was supposed to do.
- Sometimes being a little clearer with the logic.
- More in class exercises and talks about how to improve code cleanliness.
- Doing examples in class, smaller, more frequent homework exercises to complement the large projects
- More projects.
- Using something like piazza instead of a cs120 help email so that all questions/answers are open to students
- Personally, I think having more small projects/exercises that help learn the language would make the more complex exercises much more doable because I would understand the syntax much better
- Do more coding in class
- More helpful and more thorough instruction in lectures. Smaller, less overwhelming assignments. More understanding and accepting of questions and those who ask them, in addition to a better approach to answering questions.
- I think it is really well run.
- More explicitly explain the reasons behind using certain programming techniques
- more small modular exercises in class.
- I think something similar to the evaluation of the final project could have happened earlier in the semester so that we would know what exactly is expected and get a step by step walk through on the grading of each program.

11 - What should prospective students know about this course before enrolling? (You may comment on any aspect of this course such as assumed background, readings, grading systems, and so on.)

Return Rate	20/26 (76.92%)

- The course is challenging, but helps greatly in learning the languages of C and C++.
- It is a great class to learn computer science in, it teaches you all you need to know about how to become a great programmer.
- It's really important to think about your design and follow Dr. Amir's advice. Of course there's the odd student who doesn't need as much planning and still can get the assignment done in a decent amount of time, but don't assume that you're that student. Dr. Amir's advice is VERY helpful so you should always take it into account when possible!
- The course is challenging and very fun. It may seem overwhelming at first, but students quickly adapt to it. It is important not to procrastinate.
- Projects are graded on if your solution matches the reference solution
- It is probably best to have some knowledge of VIM so that students taking the course are not swamped with new editors and syntax of a new language.
- Great teacher. Great time. I loved this class.
- This course is excellent, but requires a lot of thinking (how to code in an efficient and effective way). Projects are manageable as long as you keep a good pace and reserve time to come in for help with debugging.
- You should definitely have taken AP CS and feel comfortable in programming abilities.
- Make sure you ask for help when needed, because everything is taken into account when projects are graded, such as code cleanliness and efficiency. It teaches you to think about different algorithms and great problem solving methods.
- Lots of time is spent on projects, work ahead.
- You need to be committed to putting in work outside of class to actually grasp the material.
- It is a lot of work
- It's tough but it's important
- The work isn't hard, but there's a lot of it
- The workload is extremely heavy, rather unreasonable. The grading is harsh. The professor forbids you from collaborating. You are required to do a lot of learning and working entirely alone. It is very difficult if you don't have a strong background in programming.
- Don't procrastinate on your programming projects.
- Projects will take a while. Be prepared to learn (open mind, new programming methods)
- time intensive. you will learn how to program.
- Students should already have decent experience in programming and should know that time management is important for these larger projects.