JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2017.Fall

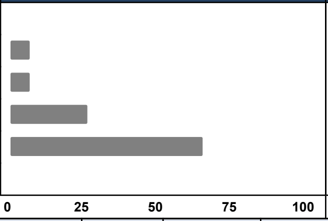
Course: EN.601.220.03.FA17: Intermediate Programming

Instructor: Yair Amir *

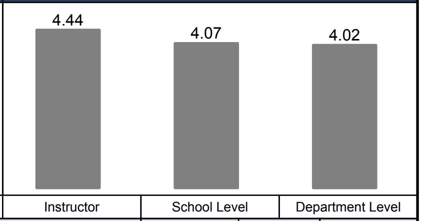
1 - The overall quality of this course is:

Response Option	Weight	Frequency	Percent
Poor	(1)	0	0%
Weak	(2)	2	6.25%
Satisfactory	(3)	2	6.25%
Good	(4)	8	25%
Excellent	(5)	20	62.5%
N/A	(0)	0	0%

Percent Responses



Means



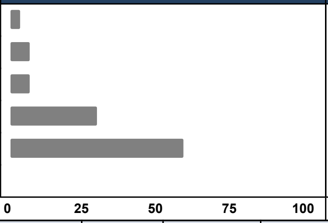
Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
32/33 (96.97%)	4.44	0.88	5.00	10,238	4.07	0.98	4.00	1,807	4.02	1.02	4.00

2 - The instructor's teaching effectiveness is:

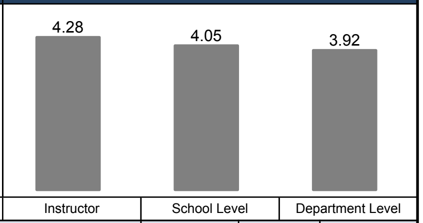
Yair Amir

Response Option	Weight	Frequency	Percent
Poor	(1)	1	3.13%
Weak	(2)	2	6.25%
Satisfactory	(3)	2	6.25%
Good	(4)	9	28.13%
Excellent	(5)	18	56.25%
N/A	(0)	0	0%

Percent Responses



Means

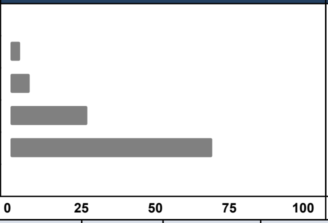


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
32/33 (96.97%)	4.28	1.05	5.00	10,146	4.05	1.05	4.00	1,785	3.92	1.14	4.00

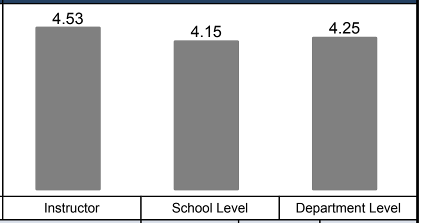
3 - The intellectual challenge of this course is:

Response Option	Weight	Frequency	Percent
Poor	(1)	0	0%
Weak	(2)	1	3.13%
Satisfactory	(3)	2	6.25%
Good	(4)	8	25%
Excellent	(5)	21	65.63%
N/A	(0)	0	0%

Percent Responses



Means

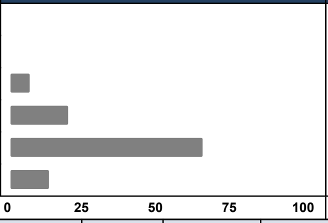


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
32/33 (96.97%)	4.53	0.76	5.00	10,136	4.15	0.90	4.00	1,781	4.25	0.88	4.00

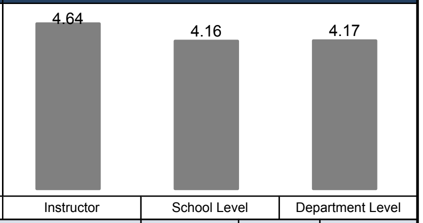
4 - The teaching assistant for this course is:

Response Option	Weight	Frequency	Percent
Poor	(1)	0	0%
Weak	(2)	0	0%
Satisfactory	(3)	2	6.25%
Good	(4)	6	18.75%
Excellent	(5)	20	62.5%
N/A	(0)	4	12.5%

Percent Responses



Means



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
32/33 (96.97%)	4.64	0.62	5.00	10,133	4.16	0.99	4.00	1,783	4.17	1.00	4.00

Course: EN.601.220.03.FA17: Intermediate Programming

Instructor: Yair Amir *

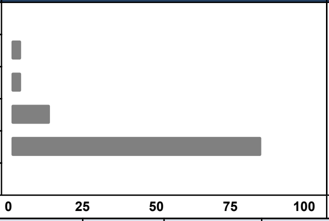
5 - Please enter the name of the TA you evaluated in question 4:

- Amy
- I don't have a specific TA, so that goes for all of them, they're all pretty good.
- Ryenne Dietrick, Billy Carrington, Suyi Liu, Eric Tsai
- There was a group of TAs and they were all pretty good.
- Su Yi
- Billy, Ryenne, Suyi, Eric
- Amy Babay
- Amy Babay
- Amy Babay
- Amy Babay
- Reynne
- Amy
- N/A -- multiple TAs
- There are many TAs.
- Ryenne Dietrick
- Amy Babay
- Not really a TA, but all the CAs and PhD students were awesome.
- N/A, 4 general Course Assistants
- Eric, Amy, Billy, Ryenne
- Amy
- Amy
- Billy, Ryenne, Suyi, and Eric
- N/A
- Amy
- Amy Babay
- Amy Babay

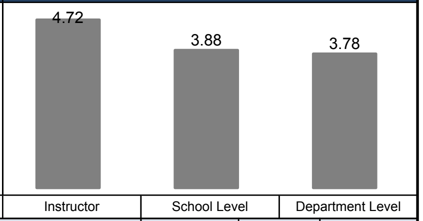
6 - Feedback on my work for this course is useful:

Response Option	Weight	Frequency	Percent
Disagree strongly	(1)	0	0%
Disagree somewhat	(2)	1	3.13%
Neither agree nor disagree	(3)	1	3.13%
Agree somewhat	(4)	4	12.5%
Agree strongly	(5)	26	81.25%
N/A	(0)	0	0%

Percent Responses



Means

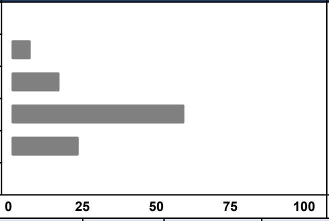


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
32/33 (96.97%)	4.72	0.68	5.00	10,091	3.88	1.08	4.00	1,767	3.78	1.14	4.00

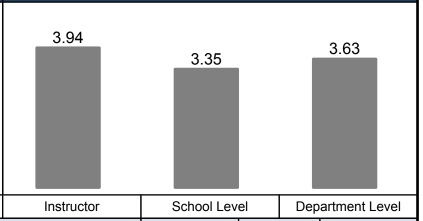
7 - Compared to other Hopkins courses at this level, the workload for this course is:

Response Option	Weight	Frequency	Percent
Much lighter	(1)	0	0%
Somewhat lighter	(2)	2	6.25%
Typical	(3)	5	15.63%
Somewhat heavier	(4)	18	56.25%
Much heavier	(5)	7	21.88%
N/A	(0)	0	0%

Percent Responses



Means



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
32/33 (96.97%)	3.94	0.80	4.00	10,120	3.35	1.03	3.00	1,773	3.63	1.00	4.00

Course: EN.601.220.03.FA17: Intermediate Programming

Instructor: Yair Amir *

8 - What are the best aspects of this course?

• The projects. They are very well designed so that they challenge you but at the same time greatly improve your programming skills.

• You get to use programming knowledge to do a lot of cool stuff. By the end you've learned how to think in algorithms, which helps you in every programming scenario.

• Great, informative lecturing about C/C++ concepts, followed by projects to use the material. Professor and TAs really care about your success and are extremely helpful in advnacing your skills.

• The projects are extremely involved, but it forces you to learn excellent programming habits by necessity. You spend a huge amount of time debugging your own code, which means that you really learn about your own style and see where it could be improved, which makes you better. The feedback is incredibly detailed for the size and number of the projects, which adds another several sets of eyes looking at your code and telling you where and how you can be better. And the professors are EXTREMELY accessible via email (all hours of the day and night) for help with bugs.

• Yair and Amy are a good teaching pair and both are very invested in their work and their students' work.

• C and C++ are interesting and fundamental languages, and learning how they work in general is important. Of the projects, most were relatively interesting data structure assignments (which used unorthodox structures to be more challenging, I presume).

• -Projects are good and you will really learn how to program and feedback on projects is very detailed.

• The professor and TA were very good teachers. The feedback on assignments was accurate and helpful.

• Great TAs

• This course adequately prepares anyone to be a better programmer, and to think about computing concepts more effectively and efficiently. In addition, the help that we got from the TA and the CAs was frequent and effective, and I felt that I could get the support I needed to get projects done.

• I really enjoyed each of the projects, but especially Project 3 and the final project, as they really put my logical reasoning and programming ability to the test. I enjoyed working through the design and code of these complex, data structure centered projects.

• Yair does a very good job teaching the material and giving assignments that challenge you.

• Learning the complex topics of programming.

• The written feedback on our programs, as well as one-on-one discussion during class was extremely useful.

• Comprehensive overview of C and C++, forces us to use good programming techniques

• Yair and Amy spend a lot of time making sure everyone in the class understands the material and is up to speed with projects. They try to arrange office hours almost every day and let you stop by their lab whenever you have a question.

• Sharply improves programming ability Exposure to ideas (e.g. pointers, memory management) not found in high-level languages (Python/Java especially) Learn to think like a programmer (designing and thinking about code before writing it)

• Taught me how to properly think about making a program from design to finish. Taught me all intermediate concepts needed for upper level courses (very encompassing). Somewhat challenging, but the challenge teaches us to think. Email feedback is extremely quick and useful.

• Yair, Amy, Tom, and the CAs. Having so many people looking at our code in such a small class allowed for a lot of progress. I was able to absorb conventions and design patterns through Yair and Amy's code samples and project feedback.

• The willingness of the CA's, Amy, and Tom to help you with any questions you might have. The projects were very interesting and difficult.

• Projects were all relevant

• Professor taught very well and the class size was small so he was able to talk to each student individually to give the best feedback possible.

• The instructors really care about the students. Although this is a tough course, they have office hours all the time and are always available if you need any help. The projects are very creative and really make you think! The feedback on the projects is also very detailed and useful, and you can tell that a lot of time and effort was put into grading the projects. By the end of the course you really are a good programmer.

• The professor is very animated and enthusiastic about the subject and you can tell he cares a lot for the students. Though his English isn't the best, it is leveraged by the fact that he brings his teaching assistant to the course to help him communicate ideas to the students. Additionally, the code feedback really helped with the projects.

• The projects and midterm helped me better my programming skills. I am better at both designing and implementing my code. Also, each assignment was followed by an extremely thorough evaluation which was also helpful in improving my skills.

• The best aspects of the course are how it teaches you to think - our in-depth projects were very useful and thought-provoking (you really had to think to see how a problem was solved), and we also learned two incredibly useful languages in computer science. This is one of my favorite courses that I signed up for in the 2017-2018 school year! Also, the TAs and Professor Amir and Amy were all incredibly, incredibly helpful!

• The best aspects of this course were the professor and TA's help on assignments which was very thorough and helpful. However, assignments were often given with little notice and only a week to complete them, and this feedback was most useful to students who happened to have time to complete assignments right away to receive the very helpful early feedback. Students who did not get as much help were at a disadvantage.

• The final project accurately represents what we learned throughout the course. The content is all related and flows together really well. It was super helpful being able to go to office hours or just stop by the lab whenever I needed help and email responses were always quick and helpful. It was obvious that Yair and all his assistants really knew what they were doing and cared about the wellbeing of everyone in the course.

Course: EN.601.220.03.FA17: Intermediate Programming

Instructor: Yair Amir *

9 - What are the worst aspects of this course?

- The lectures...they are too long and dry.
- It's really tough. There's a lot of work when projects come out, and it's all individual. The course moves at a breakneck speed as well, be careful of falling behind.
- Heavy workload at times, not as many examples of some concepts as would be helpful
- It's easy to get lost in learning the syntax of C, especially in the first several weeks of the course, where there aren't any assignments. Once you do get an assignment, it feels like there isn't nearly enough time, which is followed by a break of maybe a week with no homework, so the workload feels very start-and-stop. Also, the expectations are very high.
- Sometimes the feedback on the projects was a bit silly...
- Yair teaches C/++ as if it's still the 1980s or 1990s, and ignores advances in coding style which have been made since then, even enforcing his antiquated style somewhat in grading (though mild, this is very annoying). Sometimes I feel as though, having already studied C++ a bit myself, the explanations failed to fully flesh out a topic and its edge cases, only satisfying the necessary knowledge to complete the projects at the most basic level. The projects involving data structures were interesting, but the other parts of projects usually involved a large amount of boilerplate code to retrieve input from the user in an interactive application, which is simply repetitive and tedious. The data structures alone fulfilled the role of programming logic, class (hierarchy) design, etc. without needing other accompanying factors, I felt.
- -Grading seems unnecessarily harsh, and it's somewhat vague exactly how projects are graded. Also, I would much rather have a written midterm.
- Inconsistent workload. Sometimes would have a lot of work, sometimes none.
- Not in sync with other sections. Focused so hard on Data structure I didn't learn enough about using C/C++.
- The comments on grading for some of the projects were sometimes vague, and it would be confusing as to how to improve from one project to the next.
- I wasn't the biggest fan of the lecture style classes that we had, where we were shown pre-made code and had to listen to a lecture about how they worked. I also thought that project 4 and part 2 of the final project seemed a bit arbitrary, since
- The assignments are difficult making the few days before it is due very stressful.
- The time it takes to do some projects.
- Some projects were challenging but boring -- maybe assignments that implement a solution to a real world problem would be more interesting. For example, programming a sort of simulation of grading and tests and school is literally the last sort of thing I want to program...
- Some of the later projects are too open ended, especially the ones where you need to write a user interphase. (Project 4, Project 5 part 2)
- This course can consume a lot of your time, and projects have kind of short deadlines, so on weeks when a project is due this class can take over your life.
- Feedback is slow Not much class time is spent practicing code, only reading it off screen Workload can be painful at some times (when we have projects), nonexistent at others (when we don't)
- Although Yair attempts to attract questions, he is sometimes intimidating to approach (though I like this aspect about him, many don't like it).
- The course was taught through lectures in class which were helpful, but this was a difficult way to try to learn the nuances of C and C++.
- I really cannot think of anything. They prepared us very well and gave a lot of help and support when we needed it.
- The worst part of this course was getting back the project 4 review only about a week before the final project was due. At that point, I had already written my design and was well into implementing the final project and it was really tough to implement the suggestions from project 4.
- At times, I found the parts of the course where we view sample code as not engaging. I realize the importance of knowing the programming language so I would still pay attention but I did not learn as actively as I would have wanted to sometimes.
- It was very easy to get left behind a couple times for the project, particularly if you were slammed with work on a particular week. However, with Professor Amir giving us more time on a couple of projects, and simply with being willing to ask for help (i.e. coming to the CS lab, visiting during office hours, etc), it wasn't too bad to remedy this!
- The worst aspects were the professor's teaching style which was often scattered and used example programs to teach code instead of presentations to present the concepts. This made the material hard to follow and required students to do a lot of research out of class to figure out what was going on in the professor's examples.
- The projects were really repetitive and got kind of boring by the end.

Course: EN.601.220.03.FA17: Intermediate Programming

Instructor: Yair Amir *

10 - What would most improve this class?

- Instead of lectures, give short in-class programming assignments to complete and learn new topics. We did two of these(complex numbers and Package Inheritance) which were very helpful.
- That's a tough question! Maybe building a stronger foundation with pointers, const, debugging compilation errors, and language intricacies, because I'm still having trouble with those.
- Prof. Amir and Amy teaching more sections!
- More time on the first project. I think we could have gotten it earlier, and then maybe the semester wouldn't have started out on such a terrified foot (although I think that was the point).
- Nothing it is meant to be hard in order to create better programmers.
- Remove most or all of the "interactive application" parts of the course. They are boring, hard to test for the CAs almost certainly, and noninformative. Stop enforcing a long-dead C/C++ coding style which is not relevant. Modern programming has moved on, and there are good reasons to adopt its habits as well, beyond modernity.
- -Swtich to a written midterm and be more clear about what is expected from the projects (maybe provide a rubric beforehand)
- Clearer instructions on assignments.
- More lecture style slides or board-writing. Less "show and understand"-ing of code concepts through quickly explained examples
- I think giving more information on how to navigate Linux and Vim could have helped us write and edit our code more quickly.
- I think that going through the main concepts of each week first, before looking at code, would have been helpful. Sometimes, I would find out that I missed a concept that was mentioned in class, and had to read through a lot of the code for that week to figure out how it worked. I think that maybe having a presentation with a list of each concept would have been nice. I also think that the project description documents (especially for projects 3,4,5) could have been more clear. There were many times where I had questions about the project because something wasn't explained clearly in the description, and I had to make sure to clear it up before working on that particular section.
- Give more time for each project.
- More interesting projects. Perhaps adding some competitiveness -- who can write the most efficient code given this metric -- would be fun. (This of course in addition to the code reviews that are so helpful)
- Feedback from one project before midway point of the next More short in-class programming assignments
- I noticed there was a section of the class that was falling behind. Be more attentive to those students.
- It would be helpful to do some mini, ungraded assignments in class, like we did with ComplexNumber and Package. I think it is much easier to learn and retain information by actually writing code instead of just listening to lectures.
- This class is already structured in a way that is beneficial to the students, so I would say keep things the same.
- Receiving suggestions for previous projects before I get too far into the next project.
- Not all sample programs were dry, a lot were engaging, but for the others it would help if maybe there could be a more active way to learn the material.
- Possibly going over the general idea of a project (not just the algorithms associated with them) more than once in class - I accidentally came late to one of the classes, and I'd already missed part of the explanation, so from then on, I had a totally wrong idea of the project, and by the time I found out and had to make a complete 180, I was far behind everybody else, and that left me floundering. If I had discovered it just a couple days earlier, it probably would've turned out better, but overall, it wasn't terrible.
- Clear presentations and assignments that were not literal data structures assignments done in another language earlier in the curriculum.
- I wish there were more smaller homework assignments so we could practice the big themes like inheritance and dynamic memory allocation in small chunks rather than just seeing an example in class and then having to produce a fully functioning project. For example, what we did with the complex number practice in class was really helpful because I was able to make sure I knew the syntax and logic behind what I was doing before I started a big project. It made the project more manageable once it came to that because I could go back and look at the code I wrote if I needed guidance. I think more time in class practicing for ourselves rather than just listening to lecture would be beneficial.

Course: EN.601.220.03.FA17: Intermediate Programming

Instructor: Yair Amir *

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.)

- Learning new programming languages is frustrating at first(you will get a lot of compiler errors in this class) but by the final you'll be a much better programmer.
- It's mostly based on how well you do on projects, and the projects are hard. Be prepared to relearn how to write code in a better, more organized way.
- Prof. Amir teaches differently than the other Intermediate Programming sections. He goes slower, gives better projects and concentrates more on your individual learning. If you can get into this section, you should.
- Again the expectations are VERY high. If you are new to programming data structures, you will be confused and scared. However, the professors and to a lesser extent the CAs are extremely accessible and helpful when you have issues, and you will improve by leaps and bounds if you respond to feedback and the pressure of the assignments. If you stick with it and give it your all, you will be a much better programmer and probably very proud of yourself by the end of the course.
- Start your projects quickly and don't wait. Also try to listen to their feedback on the projects although they will probably find more feedback to give you on the next one regardless which is fine...!
- This course seems significantly easier (project-wise) and perhaps slower-paced than More's course. I would honestly recommend taking it with More instead of Amir, simply because I think what you will take away would be deeper and more contemporary. However both offer a fair view of programming to an absolute newcomer (which I was not).
- -Have a strong understanding of programming fundamentals (eg loops, conditions, subroutines, recursion, OOP concepts, etc.).
- Heavy workload, but worth it. The professor tries very hard to make sure that every student is a great programmer by the end of the course.
- Pay attention as much as physically possible. Update your design
- This course is very time-consuming; be prepared to spend a lot of time compiling and editing your projects. Also, don't be afraid to ask for help! Yair, Amy and the CAs are more than willing to help with any questions.
- Students taking this course definitely need a strong background in the fundamentals of programming (including variables, functions, loops, classes/objects, logic etc). You do not, however, need any experience with C or C++, as we start from scratch when learning the syntax of the languages. The class is heavily project-based: almost all of the homework assigned is simply working on a graded project. Each project takes about 2-3 weeks to complete, and there are 5 total projects assigned throughout the semester. The rest of the class is lecture based, where the professor shows the class some premade example code in order to demonstrate certain concepts in programming.
- This is a very good course to learn about programming nomatter what your level.
- Spend time on the projects.
- Good introduction to C/C++. Professor is extremely nice and seems to care about every student. A decent amount of work.
- Easy if you have a strong background in programming, difficult if you have taken very little.
- You learn a lot in this class and Yair is great, but you have to be prepared to dedicate a lot of time and hard work.
- Workload can be heavy
- Highly recommended course to take as computer science fundamentals are very important to have in any field. Better if you're already familiar with basic programming concepts.
- It is a lot of work, but it is definitely worth it. Take Intermediate with Yair if you can!
- This is a great intro class with lots of support to help you succeed. However, the workload is considerable.
- The first project may not go well for you because you haven’t learned to code effectively yet, but stuck with the course because you will learn a lot and will do better on the next four projects!
- This course doesn't just teach you C/C++, it also teaches you how to become a better programmer e.g: code quality, design, efficiency. These things makeup a significant portion of your grade. Because of this, don't expect to get a good grade on the first project because you probably didn't pay enough attention or don't know how to do these things so the first project is meant to be a wake up call. Don't worry because you have more than enough opportunities later in the semester to bring your grades back up.
- If you complete the course in good standing, you are sure to have good programming skills (including design and implementation).
- They should be prepared to learn, think, and have fun at the same time. Overall, this was definitely a great course!
- Students should know that compared to the course number of 2xx, the workload is very high.
- Do not procrastinate. AT ALL
- Prospective students should know to get started early and not be afraid to go into the lab and ask questions and get help. I felt like I had to figure everything out on my own but that's not the way the course is designed, the CAs and Amy and Yair really just want to help you as much as they can. Also, the grading system is very fair and the comments you get back on projects are usually detailed and help to improve future assignments.