JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2014.Spring

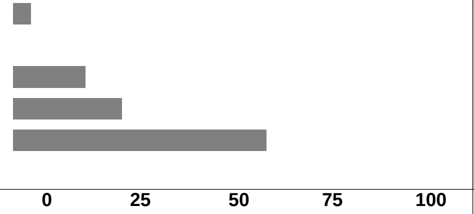
Course: EN.600.120.01.SP14 : Intermediate Programming

Instructor: Yair Amir *

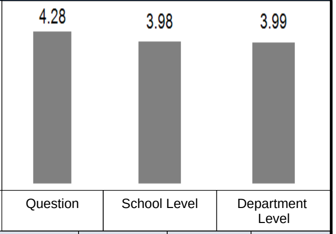
1 - The overall quality of this course is:

Response Option	Weight	Frequency	Percentage
Poor	(1)	1	4%
Weak	(2)	0	0%
Satisfactory	(3)	4	16%
Good	(4)	6	24%
Excellent	(5)	14	56%
N/A	(0)	0	0%

Percent Responses



Means

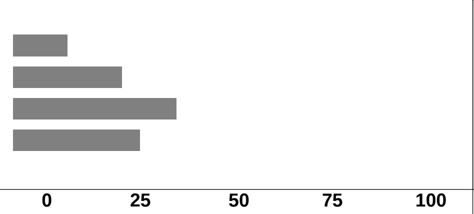


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/27 (92.59%)	4.28	1.02	5.00	7,431	3.98	0.95	4.00	887	3.99	0.90	4.00

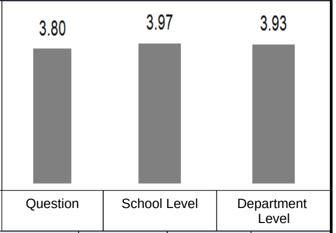
2 - The instructor's teaching effectiveness is:
Yair Amir

Response Option	Weight	Frequency	Percentage
Poor	(1)	0	0%
Weak	(2)	3	12%
Satisfactory	(3)	6	24%
Good	(4)	9	36%
Excellent	(5)	7	28%
N/A	(0)	0	0%

Percent Responses



Means

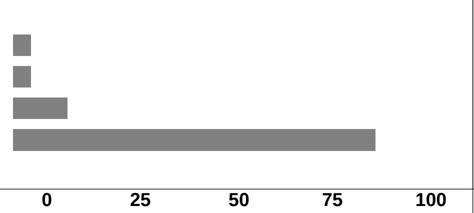


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/27 (92.59%)	3.80	1.00	4.00	8,146	3.97	1.05	4.00	956	3.93	0.96	4.00

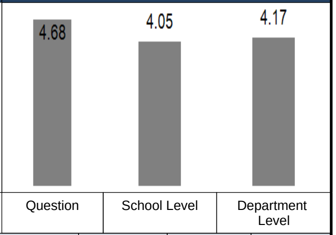
3 - The intellectual challenge of this course is:

Response Option	Weight	Frequency	Percentage
Poor	(1)	0	0%
Weak	(2)	1	4%
Satisfactory	(3)	1	4%
Good	(4)	3	12%
Excellent	(5)	20	80%
N/A	(0)	0	0%

Percent Responses



Means

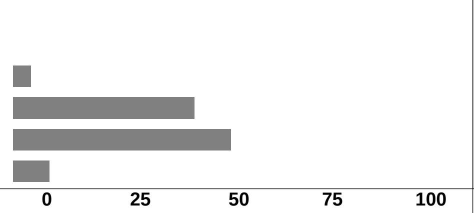


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/27 (92.59%)	4.68	0.75	5.00	7,379	4.05	0.91	4.00	877	4.17	0.86	4.00

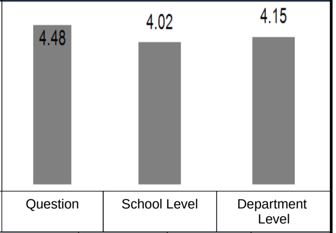
4 - The teaching assistant for this course is:

Response Option	Weight	Frequency	Percentage
Poor	(1)	0	0%
Weak	(2)	0	0%
Satisfactory	(3)	1	4%
Good	(4)	10	40%
Excellent	(5)	12	48%
N/A	(0)	2	8%

Percent Responses



Means



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/27 (92.59%)	4.48	0.59	5.00	7,376	4.02	1.04	4.00	878	4.15	1.04	4.00

Course: EN.600.120.01.SP14 : Intermediate Programming

Instructor: Yair Amir *

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

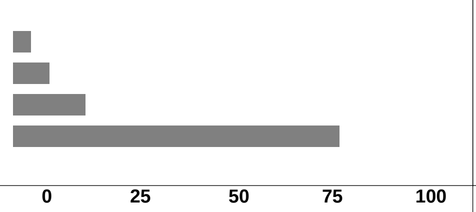
Return Rate	20/27 (74.07%)

- All of the TA's were great
- All of them
- All of them
- Amy Babay
- Amy Babay
- Amy Babay
- Amy Babay
- Amy Babay
- Diana Chen
- Jeff
- Jeff
- Jeff Dallatezza
- Jeff Dallatezza
- Jeff/Amy + all the TAs for the 1:30 Section!
- Jeffery Dallatezza, Amy Babay
- Jeffrey
- Jeffrey DallaTezza
- Jeffrey Dallatezza
- Jeffrey D'Allatezza
- Many different ones

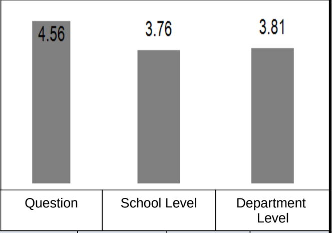
6 - Feedback on my work for this course is useful:

Response Option	Weight	Frequency	Percentage
Disagree strongly	(1)	0	0%
Disagree somewhat	(2)	1	4%
Neither agree nor disagree	(3)	2	8%
Agree somewhat	(4)	4	16%
Agree strongly	(5)	18	72%
N/A	(0)	0	0%

Percent Responses



Means



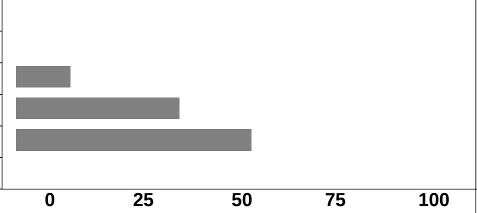
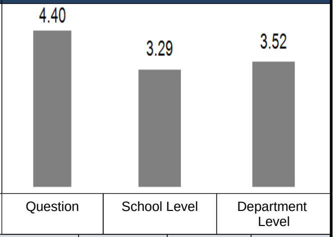
Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/27 (92.59%)	4.56	0.82	5.00	7,344	3.76	1.07	4.00	876	3.81	1.08	4.00

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2014.Spring

Course: EN.600.120.01.SP14 : Intermediate Programming

Instructor: Yair Amir *

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

Response Option	Weight	Frequency	Percentage	Percent Responses	Means
Much lighter	(1)	0	0%		
Somewhat lighter	(2)	0	0%		
Typical	(3)	3	12%		
Somewhat heavier	(4)	9	36%		
Much heavier	(5)	13	52%		
N/A	(0)	0	0%		

Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/27 (92.59%)	4.40	0.71	5.00	7,378	3.29	0.99	3.00	876	3.52	1.01	3.00

Course: EN.600.120.01.SP14 : Intermediate Programming

Instructor: Yair Amir *

8 - What are the best aspects of this course?

Return Rate	23/27 (85.19%)

- Challenging

- Challenging and interesting

- Every part was very challenging but very enlightening.

- Great course

- Hands on programming. You learn a lot.

- I like how easy it is to get help and instead of just telling us how to fix it, the instructors will go through and try and help us understand it ourselves. The instructors really seemed to care about how well we did in the course. I feel as if the only way to teach a programming language is through example code, and the instructors did a very good job in teaching it and making sure we understood.

I also liked how we got feedback for all of our projects. The instructors actually take the time to read and provide valuable feedback for each student and I found that very helpful as we moved on to more demanding projects.

The instructors also gave us extensions on our projects so we could turn in more quality projects instead of just rushing through and not submitting our best work.

- I love the tutorials, when you can work 1 on 1 with a TA looking at your code

- It challenges you to think much more deeply about your programs, and you become a much better programmer by the end of the semester.

- Learn a TON about programming, and the projects we do are very interesting

- Learning C and C++ in depth, along with some interesting data structures.

- Learning two languages that are very applicable and useful

- teacher was very helpful, amazing teaching style, great projects, really helped my coding ability improve

- The amount of help given by the TAs and CAs.

- The best aspect of this course is that you come out of it a much better programmer. The professor pushes you to do high quality work, and it is for the better.

- The course is challenging but not too much so. TAs and CAs are both incredibly helpful at finding bugs in programs and the schedule is very organized.

- The hands-on approach to learning the basics of programming

- The hands-on teaching style and the projects that test new concepts well.

- The intellectual challenges provided by the projects (especially the 3rd and the final ones) are easily the best aspects of this course!

- The staff that run this course are the most hardworking I have ever seen. They are a huge support system. They know we struggle, they know it's hard, but they know it's worth it, and they're there every step of the way. I think the in-class tutorials are great and extremely useful. I wish we did a few more in-class activities other than lectures (some group assignments would be interesting). The projects, though overwhelming at times, were very well put together, however I think it would be useful to spend a day or so on basic things such as good design techniques, efficiency, and standard libraries, etc. Overall, I feel that the course is very well run and part of me thoroughly enjoyed it. I can't imagine someone leaving this class without becoming 300x the programmer they were before taking the course.

- There was a lot of one on one time with the TA's. Lot's of time and ways to get help.

- This class begins as a huge challenge. It seems like it is too much to handle, however, this only demonstrates to students the challenge a problem can represent when you approach it without emphasis on the design. As the course proceeds, the complexity of programs increase to a level at which it is impossible to just hack away and code.

This class more than anything teaches problem solving in an efficient way. Thus, this class teaches far more than coding and other topics for computer science. I would think this class should be required for every engineer.

- This is a very challenging course, but there is a lot of help. The course assistants are plentiful and they are all really good. They are there to help at all hours of the day and they really know what they're doing.
Yair is so smart. His research is incredibly interesting, life changing stuff. And he is adorable.

- You really learn how to code, and most of your grade is project-based.

9 - What are the worst aspects of this course?

Return Rate	23/27 (85.19%)

Course: EN.600.120.01.SP14 : Intermediate Programming

Instructor: Yair Amir *

- Challenging

- Code learning in class is not very engaging. Reviews are often necessary for grasping new syntax.

- definitely very challenging, you have to schedule your work well and start early

- Despite the number of TAs, it is difficult to get help. No focus on teamwork, which is integral to programming.

- Great course

- I'd say that the workload is a lot heavier than most classes and the grading is strict (but fair).

- It is a ton of work, and when you complete a project but have a small bug in it (regardless of whether the design is solid or not), you often get penalized for the resulting mistakes (like memory management) that all stem from one issue.

- It is sometimes difficult to comprehend the material just by going over code. It is sometimes too dry to hold my attention. Doing practice with the complex class and the package class helped in the C++ portion.
Also the midterm was not helpful. We do not get enough time to complete the assignment and it is worth a large portion of our final grade. At least for me, the midterm was not an indication of my grasp of the material but how fast I could type. All it did was hurt my grade. (That being said, the dry-run was absolutely necessary).

- It was extremely difficult to keep up with the workload. You were thrown a variety of topics and you had to become familiar with them in no time so that you can solve and complete the projects that were given to you.

- Long assignments. That's the only way to learn though.

- Lots of material in a short amount of time and large projects. Most students I've seen can't finish the projects without asking for help from other students/TA's. Teaching style of code on large screen can be sometimes ineffective, would rather have a more do-it-yourself approach.

- Most of the time when i had a question in this course, it was either because I could barely understand what the assignment was about because it was very poorly explained by the instructor or it was about a bug that I have been working on for hours. Although there were lots of opportunities to get help, if I asked the instructor ANYTHING, he would be extremely disrespectful and talk to me like I'm the dumbest kid on the planet. I think many people in the class would agree that the professor had a temper problem when dealing with our questions. The TA's were definitely better with this, however, if I asked for help with a bug that I was stuck on for hours, rather then giving me efficient help and pointing out the mistake so i could fix it myself, they would give me an answer like " Why don't you try to think of this, or try to error check in this way, or try to redesign this". But the problem is, IVE ALREADY THOUGHT OF ALL THESE, THAT'S WHY IM ASKING FOR YOUR HELP IN THE FIRST PLACE. I always end up rewriting my code over and over again because of this kind of answer from the TAs. The grading for this course was also a horrible aspect of the course. Rather then getting graded on my code, most of the time, I was losing points because of my design file. I would also lose points in one area of the rubric such as "code efficiency", then lose points for the same error somewhere lower in the rubric such as "overall code quality" or "runtime error", which would add up in the end giving me a low score. The midterm was definitely unfair as well. It did not test whether you knew how to code or how well you knew your material, it tests how fast you can code. Not an accurate measurement of our skills in the class AT ALL.

- none

- Other than the workload (especially the final project that I felt was a little too much), I felt that the midterm was a bit unfair, not in regards to us not knowing the material or it being a surprise (the dry run is a great idea), but I am just not a big fan of having to code under such a time restraint when I was taught to do the opposite.

- Sometimes program instructions can be a little vague. I've implemented a few of my programs in ways that have had points taken off for not doing things that the paper implied but did not explicitly state. For example, the midterm instructed to create a list that could work with a provided driver program. I noticed that the driver inserted numbers in ascending order so I did no account for any other possibilities. I then got points taken off for not accounting for the other possibilities.

- Sometimes the professor and a few of the TAs could be abrasive when you asked them questions. If they covered a topic in lecture, they didn't see it as an option that you might not understand it the first time, and often made me feel dumb for asking questions.

Also, at the end, I received a B+ in the class, but was told that I shouldn't have taken the course if I couldn't do better than that. Ridiculous.

- Sometimes Yair would lose his temper on students, and not be very helpful if they don't understand

- The course is mostly taught by example, but not many CS concepts are touched upon, I feel.

- The grading is harsh considering this is the second cs course many students take. The projects are time-consuming and require your full attention and focus. The instructor can be intimidating, making it hard to ask for help. Lectures are not hands-on, and I felt I didn't learn anything on those days. You learn the most from actually doing the projects and having the TAs available to help you.

- The intellectual challenges are called challenges for a reason

- The worst aspect of this course is that it is extremely time consuming.

- There was not enough time given to take the midterm, and the projects were too complex.

- This class teaches quite a bit about data structures, which is incredibly helpful and educational. However I wish it had projects that were based more on programs that we could use in the terminal ourselves.

Course: EN.600.120.01.SP14 : Intermediate Programming

Instructor: Yair Amir *

10 - What would most improve this class?	
Return Rate	23/27 (85.19%)

- A different teaching structure. Examples were overwhelming and generally printing out long codes doesn't help students, too hard to read code in black/white and not effectively documented to help students understand.

- A more studyable midterm

- Give small exercises in class so that students can be more engaged when learning code. Teach the theoretical part of the program more in depth.

- Having some more TAs positioned in the CS labs would probably be beneficial.

- I think it would be better to have more frequent smaller projects throughout the year instead of four large ones and a final project.

- I wish we did a few more in-class activities other than lectures (some group assignments would be interesting). The projects, though overwhelming at times, were very well put together, however I think it would be useful to spend a day or so on basic things such as good design techniques, efficiency, and standard libraries, etc. At the beginning of the class, I remember getting my first few grades back and seeing negative marks regarding some of those things and wishing that I knew more about them before having to do the assignment. I also felt that the midterm was slightly unfair. Some people are fast, quick-thinking programmers that could handle a situation like the midterm well, but I was slightly upset because my coding technique was a lot more careful, toned back, and slower-paced, because that's how I'm used to programming and that's how I was taught to do so, so I ended up doing poorly when I feel that I knew the material well. I know I'm not the only person who felt that way about it, especially since it's a huge part of our grade. I think it would be useful and interesting to do a qualitative exam rather that quantitative since it would level out the playing field more and encourage interest and engagement in other aspects of programming and the languages (like efficiency, design, etc.).

- It would be better to divide people into teams. That way team members can teach each other the basics, and keep each other accountable to deadlines. TAs would then be able to service the entire class without much help from CAs

- More effective teaching for the concepts - I had to figure a lot out myself (which isn't really a problem), but if the teaching was done more methodically (maybe sheets with syntax or overview to make it easier to follow the code walkthroughs on the projector), then I feel like the projects could have been easier.

- More smaller assigments between the big projects

- more time to complete the assignments

- More ungraded in-class programs. The only way to learn programming skills is to practice, and it seemed as though we would go for several weeks learning programming concepts that were never put to use until we had a project to do.

- none

- nothing

- Nothing much I can think of, maybe changing the lecture style to not just be talking through code examples

- Sometimes lectures on how to code are very boring.

- The class should involve writing more sample programs before jumping into the projects because going over programs doesn't really teach you anything.

- The course focuses on showing code, which is great, but I feel like there should be more aspects on topics like "What is efficiency?" and "What's considered concise code?" and other related topics.

- The feedback on projects is not consistent throughout the course. I had been doing a specific syntactical thing on the first four projects and received no feedback that I shouldn't be doing it, but then received negative comments about it on the final project. Just standardize the way these things are graded, because it seems really arbitrary right now, depending on who is grading it.

- The projects shouldn't be as large but there should be more than 5 projects. By increasing the number of projects and reducing the amount for each project, I feel that I can more effectively learn the material in a time efficient manner.

- The way the assignments get graded. Also, the difficulty of the assignments should be reduced. Should not have homework assignments that were previous PhD projects. Also, remove the midterm.

- This class is actually set up in a really great way, and I don't think there is anything that can improve it.

- To work on code in class. Not neccesarily big chunks of code, but it would help a lot if the basis for a new idea was developed in class (by writting a small program together with the students, so that one can have an idea of what the new topic acutally is)

- We should do more ungraded practice coding. When it comes time for the projects, we are just thrown in without an idea of where to start.
Also, sometimes instructions on the projects are a little confusing. The minor details could use clarification.

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2014.Spring

Course: EN.600.120.01.SP14 : 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.)	
Return Rate	22/27 (81.48%)

- a lot of homework

- Be prepared for the workload. I was constantly overwhelmed by this course halfway through the semester--it was much more work intensive than I expected coming into it. Though it was the most difficult class I have ever taken, I can wholeheartedly say that I am a better programmer because of it. I can't imagine it any other way; the struggle was worth the reward. The feeling of accomplishment when completing this class (or an assignment) is incredible (and the professors are incredible and engaging as well). The staff that run this course are the most hardworking I have ever seen. They are a huge support system. They know we struggle, they know it's hard, but they know it's worth it, and they're there every step of the way.

- Be prepared to put in a lot of work

- have a strong grasp on the basics of programming, you will need it

- If you like programming, it is a very enjoyable course. Yair knows what he is talking about.

- It is a fun course and Yair is a fantastic instructor. You'll come out of this course a better programmer.

- It is hard, but don't be afraid to ask for help. There is plenty of help, you just need to ask for it. Amy does not have a background in computer science and now she is teaching the course! Don't be afraid to take computer science courses. Yes, it will be difficult, but it is rewarding (even if it hurts your GPA).

- It requires quite a bit of work. Be prepared to be committed to it or do poorly. If you are committed and do your work ahead of time, not pushing it off until the end, you will be fine.

- Learn to type your ideas

- Lots of time

- N/A

- Required for CS majors. Very useful to learn how to program in C/C++. You may want to take data structure along or before taking this course. There are only 5 projects for the entire course, and they determine most of your grade. These projects are usually long and complex, so starting early is almost necessary. Learn how to write design document BEFORE you start coding. C/C++ syntax is trickier than that of Java, but it's manageable.

- Students should be prepared to work hard on their projects but have a great time challenging themselves in doing so! The projects are very reasonable, and the course is generously structured to help you manage your time!

- TAKE THIS COURSE. It is VERY hard, and you will work your ass off, but you learn a ton, and it is very worth it in the long run.

- The workload is fair, but you should be aware that the material and grading is much more than intro programming classes.

- This class is a lot of work.

- This course is a very heavy load. Weeks where there are projects due can be really intense. However, this course also teaches you to be a much better programmer and allows you to see if you are cut out for computer science.

- This course is very time consuming but VERY rewarding. You have to work hard to earn your grade. To get the most out of the course, I recommend reading the book before class and asking for help when you need it. The instructors are always nice and willing to help.

- This is a 100 level course but seems much higher. You must be prepared for a large workload that includes mostly of yourself working on code.

- This is not Intro to Java, the class is much harder. Start your projects early because problems will arise and there isn't as much time as you think there is. Don't be afraid to ask for help because TAs can fix your problems much faster than you probably can. Take advantage of the tutorial days where you just work on your projects in class. Don't be discouraged by the harsh grading.

- VERY difficult workload

- You will spent endless upon endless hours working for this class.

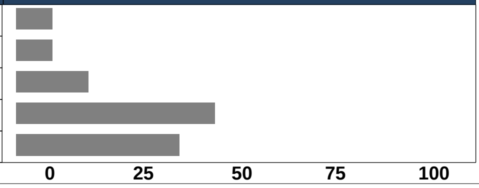
JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2014.Spring

Course: EN.600.120.01.SP14 : Intermediate Programming

Instructor: Yair Amir *

12 - This course improved my appreciation for and/or ability to engage in life-long learning.

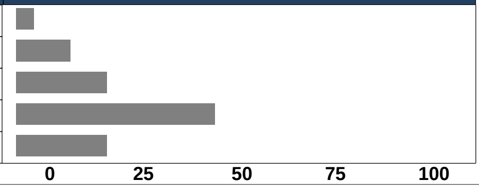
Response Option	Weight	Frequency	Percentage
Disagree strongly	(1)	2	8%
Disagree somewhat	(2)	2	8%
Neither agree nor disagree	(3)	4	16%
Agree somewhat	(4)	11	44%
Agree strongly	(5)	9	36%



Return Rate	25/27 (92.59%)

13 - This course deepened my understanding of the social impact of computing.

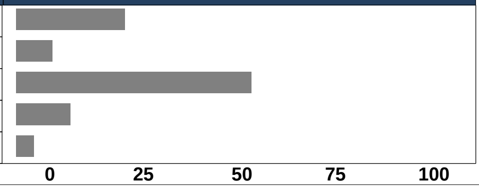
Response Option	Weight	Frequency	Percentage
Disagree strongly	(1)	1	4%
Disagree somewhat	(2)	3	12%
Neither agree nor disagree	(3)	5	20%
Agree somewhat	(4)	11	44%
Agree strongly	(5)	5	20%



Return Rate	25/27 (92.59%)

14 - This course enhanced my ability to work effectively in a team.

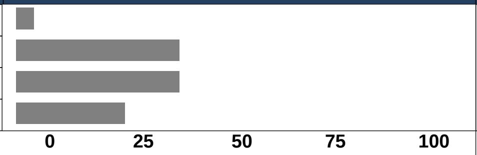
Response Option	Weight	Frequency	Percentage
Disagree strongly	(1)	6	24%
Disagree somewhat	(2)	2	8%
Neither agree nor disagree	(3)	13	52%
Agree somewhat	(4)	3	12%
Agree strongly	(5)	1	4%

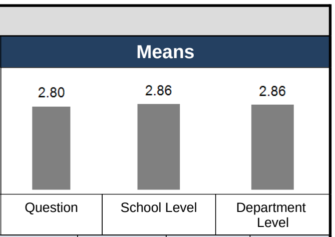


Return Rate	25/27 (92.59%)

15 - How effective were the in-class activities for learning the material?

Response Option	Weight	Frequency	Percentage
Not at all effective	(1)	1	4%
A little effective	(2)	9	36%
Somewhat effective	(3)	9	36%
Very effective	(4)	6	24%

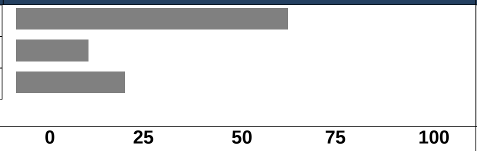


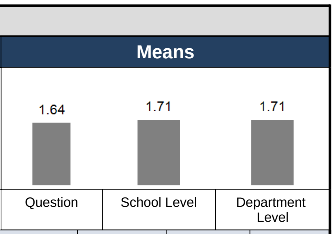


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/27 (92.59%)	2.80	0.87	3.00	156	2.86	0.83	3.00	156	2.86	0.83	3.00

16 - When doing in-class exercises I prefer to work

Response Option	Weight	Frequency	Percentage
by myself.	(1)	15	60%
with one partner.	(2)	4	16%
in a small group.	(3)	6	24%





Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
25/27 (92.59%)	1.64	0.86	1.00	156	1.71	0.80	1.00	156	1.71	0.80	1.00

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2014.Spring

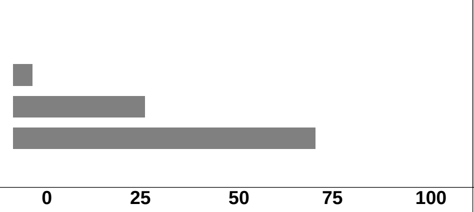
Course: EN.600.120.02.SP14 : Intermediate Programming

Instructor: Yair Amir *

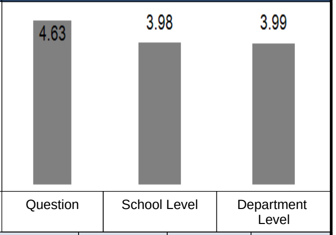
1 - The overall quality of this course is:

Response Option	Weight	Frequency	Percentage
Poor	(1)	0	0%
Weak	(2)	0	0%
Satisfactory	(3)	1	4.17%
Good	(4)	7	29.17%
Excellent	(5)	16	66.67%
N/A	(0)	0	0%

Percent Responses



Means

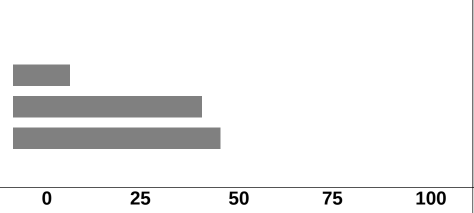


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
24/25 (96%)	4.63	0.58	5.00	7,431	3.98	0.95	4.00	887	3.99	0.90	4.00

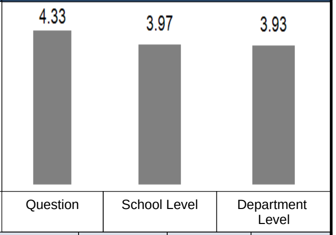
2 - The instructor's teaching effectiveness is:
Yair Amir

Response Option	Weight	Frequency	Percentage
Poor	(1)	0	0%
Weak	(2)	0	0%
Satisfactory	(3)	3	12.5%
Good	(4)	10	41.67%
Excellent	(5)	11	45.83%
N/A	(0)	0	0%

Percent Responses



Means

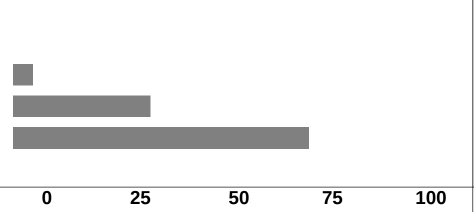


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
24/25 (96%)	4.33	0.70	4.00	8,146	3.97	1.05	4.00	956	3.93	0.96	4.00

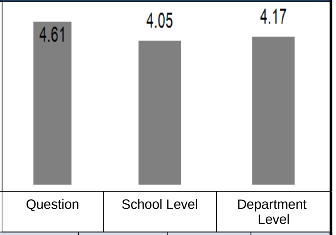
3 - The intellectual challenge of this course is:

Response Option	Weight	Frequency	Percentage
Poor	(1)	0	0%
Weak	(2)	0	0%
Satisfactory	(3)	1	4.35%
Good	(4)	7	30.43%
Excellent	(5)	15	65.22%
N/A	(0)	0	0%

Percent Responses



Means

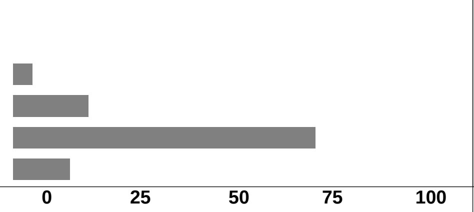


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
23/25 (92%)	4.61	0.58	5.00	7,379	4.05	0.91	4.00	877	4.17	0.86	4.00

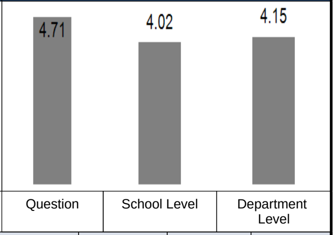
4 - The teaching assistant for this course is:

Response Option	Weight	Frequency	Percentage
Poor	(1)	0	0%
Weak	(2)	0	0%
Satisfactory	(3)	1	4.17%
Good	(4)	4	16.67%
Excellent	(5)	16	66.67%
N/A	(0)	3	12.5%

Percent Responses



Means



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
24/25 (96%)	4.71	0.56	5.00	7,376	4.02	1.04	4.00	878	4.15	1.04	4.00

Course: EN.600.120.02.SP14 : Intermediate Programming

Instructor: Yair Amir *

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

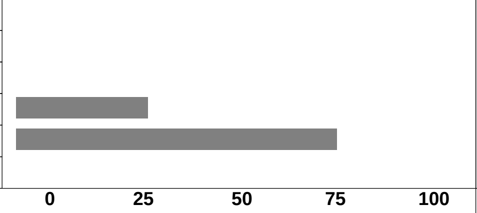
Return Rate	18/25 (72%)

- Abby
- All of them are amazing! (4:30-5:45 section)
- Amy
- Amy
- Amy
- Amy Babay
- Amy Babay
- Amy Babay
- Amy. Jeff, Meg and Tom
- Farhan Damani
- Jeff
- Jeff
- Jeff DallaTezza
- Jeff DallaTezza
- Jeffrey DallaTezza
- Jeffrey, Amy, etc.
- Jeffrey, Diana, Amy, Tom
- Papa Jeff and Mama Amy

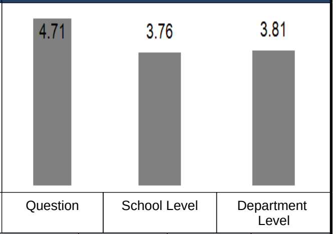
6 - Feedback on my work for this course is useful:

Response Option	Weight	Frequency	Percentage
Disagree strongly	(1)	0	0%
Disagree somewhat	(2)	0	0%
Neither agree nor disagree	(3)	0	0%
Agree somewhat	(4)	7	29.17%
Agree strongly	(5)	17	70.83%
N/A	(0)	0	0%

Percent Responses



Means



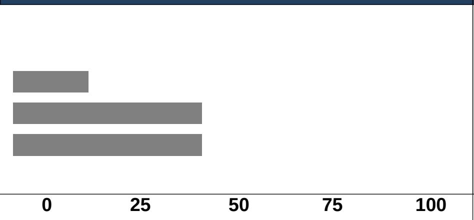
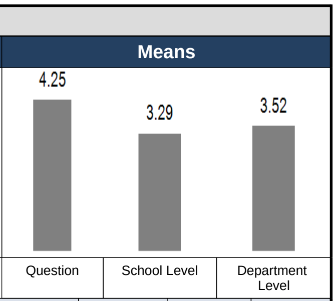
Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
24/25 (96%)	4.71	0.46	5.00	7,344	3.76	1.07	4.00	876	3.81	1.08	4.00

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2014.Spring

Course: EN.600.120.02.SP14 : Intermediate Programming

Instructor: Yair Amir *

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

Response Option	Weight	Frequency	Percentage	Percent Responses	Means
Much lighter	(1)	0	0%		
Somewhat lighter	(2)	0	0%		
Typical	(3)	4	16.67%		
Somewhat heavier	(4)	10	41.67%		
Much heavier	(5)	10	41.67%		
N/A	(0)	0	0%		

Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
24/25 (96%)	4.25	0.74	4.00	7,378	3.29	0.99	3.00	876	3.52	1.01	3.00

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2014.Spring

Course: EN.600.120.02.SP14 : Intermediate Programming

Instructor: Yair Amir *

8 - What are the best aspects of this course?	
Return Rate	22/25 (88%)

- A very good programming course that will increase your programming skills significantly.

- Assignments are worth while

- Best course I've ever taken at Hopkins. The professor is extremely dedicated to helping students learn. TAS offer alot of help and expertise. The course is very challenging, but worthwhile.

- Cool and valuable introduction to C and C++. I feel confident about my knowledge in both languages.
The projects are fun. I really liked how all the projects kind of fit together.

- Excellent homework assignments-- even though the lectures were helpful, I feel like the majority of what I learned in this class came from the programming assignments. The CAs were also extremely helpful and significantly improved my course experience. I also found the tutorials on Fridays to be very useful, especially the "Complex.cpp" class which solidified my understanding of overriding operator<<.

I thought the MLH and skiplist assignments were excellent and where I learned the most.

Lastly, I felt that Yair genuinely cared about the performance of his students and did a lot to help us do our best as students.

- Gives you a very fundamental introduction to programming in C and C++.

- Great introduction to C and C++. I really learned a lot. The professors were very helpful both in class and during office hours. The feedback on my programs were great. They were thorough and I really learned a lot from them. They were thoughtful as well. I really enjoyed this class.

- I feel projects were what really allowed me to learn both C and C++.

- I think the most helpful thing about this course was the cs120help email. Everyone is always so fast to help and it comes in really handy when you've been stuck on a part of code for a while (especially in debugging).

- Interesting projects
Good course assistants

- Learn a lot

- learn c, c++

- No paper exams. Practicals and projects only.

- Solid foundations for any programming needs in the future

- The best aspects of this course is the instructor (Yair) and his enthusiasm when teaching the course. I was nervous about learning a new language (C/C++), but the instructors make it really easy to understand and give you plenty of sample code to start out with each time they introduce something new. The projects are also very well spaced out, and they are really flexible when it comes to deadlines. If they hear that students are taking longer than they expected to complete the project, they extend the deadline just a little so you can submit the best work possible. It really feels like they want you to do well and have to opportunity to submit something you can be proud of.

- The C section of this course and the fact that it assumes no one knows anything about C

- The feedback was very useful to figure out what was done wrong.

- The instructors and teaching assistants were helpful and the feedback was helpful in improving preceding work. The lectures were also helpful. I also liked how the last project was cumulative of everything learned.

- There is plenty of one-on-one attention, and all the instructors/TA/CAs really care about helping you understand C/C++ and do well on the projects. There is a lot of feedback on projects, and overall, I thought I grasped the concepts they taught in class.

- This class is very challenging but definitely teaches students to program at a higher level. There are work days designed into the schedule that allows for students to work on their code with CAs around to help with any problems they may encounter.

- Yair is great, pretty funny. The TA's and the CA's were great at teaching/helping.

- You learn a lot and I am now confident in my programming.

Course: EN.600.120.02.SP14 : Intermediate Programming

Instructor: Yair Amir *

9 - What are the worst aspects of this course?

Return Rate	20/25 (80%)

- data structures and long difficult assignments

- Difficult and very time consuming

- Extremely heavy workload, the projects are ridiculous.

- heavy courseload

- Lectures can get boring. A lot of stuff is thrown at you at once

- Must have a lot of time to be able to complete everything for each assignment

- Nothing?

- Often very fast paced.

- Sometimes I felt the grading was a bit inconsistent; depending on the grader, I would get points marked off for my 'design' document and the criteria (i.e. the different areas I got points off on) wasn't the same across the board. For example, I got points off on something in my second project formatting that hadn't been mentioned in my first project's design document, and I would obviously liked to have not lost those points. Also, Yair, while very enthusiastic and passionate about what he's teaching, can be a bit difficult to get help from because he sometimes seems agitated when we have project 'help' sessions in class.

Most of the projects I thought were interesting and great problems to work with, but I found project 4 frustrating. I thought the assignment itself was incredibly boring, and I think they could have taught the same concepts in two smaller projects.

- the later half of the class is very fast paced

- The lectures are helpful because the instructors go through different codes and structures in detail. However, since this is a longer class it is difficult to keep focused for the entire hour and fifteen minutes without a break from looking at code and listening to the instructor.

- The midterm. I'm just not good at programming in a vacuum.

- The only thing I would think to complain about is the wording for some assignments. Often times, the assignments are a little vague when it comes to little things--like counting steps and some small amount of functionality. I understand that a lot of this is to give the student freedom when doing their projects, but sometimes it is a little confusing.

- The second half of the course (C++) was not as clear to me as the first part. I found myself feeling unprepared for Project 4 and actually found it to be more challenging than any of the other homeworks even though I got a very high (>95) grade on it.
I also thought the lectures were fairly dry sometimes and could be more interactive.
Lastly, I did not enjoy the fact that we had an email list instead of a Piazza. I would much prefer Piazza (or some other discussion/announcements board) to the list serve.

- The way the lessons are structured are really ineffective for learning. For most part of the course I felt I had work on my own to get the concepts mentioned in class, mainly because of the fast pace.

- This class was a lot of work but if you utilized all the available resources it was very manageable.

- Very big projects.

- Very difficult assignments

- We are just handed a packet of code in class and we just go over them. This is pretty hard to follow at times.
The final project took a very long time and is a little convoluted. The midterm is not a bad idea, but the task was too much in too little time.

- While we learn object oriented concepts, it doesn't quite touch upon grander ideas of actual software development. Still looking at things under a microscope.

Course: EN.600.120.02.SP14 : Intermediate Programming

Instructor: Yair Amir *

10 - What would most improve this class?	
Return Rate	19/25 (76%)

- A better explanation of what they were looking for in the design documents

- cut out some of the less important topics towards the end of the course in favor of going more in depth on topics such as inheritance

- Doing smaller assignments along with the projects and less projects.

- I feel like focusing on one of the two languages would make students more comfortable with writing code, sooner in the course. Personally I think we should have spent more time on C++.

- I think adding more in-class exercises like the "Complex.cpp" one would really make the lectures better. The lectures were good but could be a bit dry, and having more in-class exercises would make attending even more worthwhile.
  Furthermore, I think Piazza would help resolve the issue of too many emails that not everyone reads.

- I think less emphasis on formatting (or at least, clearer requirements like no text past 80 characters) in our grades.

For project 4, which I found to be the weakest and most boring, I think it could be split into two projects. The first half would be one where we create some classes (which hopefully would be explained in greater detail) and be provided with some basic program to that runs these classes. The second half could be creating our own 'driver' program. The reason I think this might be better is because the main point (from my POV) was to learn about the new C++ class concept/to learn C++ syntax. However, I felt like I spent so much time just trying to 'detangle' the project to understand exactly what we were supposed to be doing with everything that I didn't invest time into actually designing my program. Splitting it into 2 1-week-long projects might make everything clearer.

- It was helpful having the cs120 help email but I feel like it would be better for the class to just use piazza so students could also interact with one another and see what kind of problems other students are running into and how to fix them.

- Maybe specifying a little more in the project description would help clear things up. Other than that, the course is really good! But programming is always really fun.

- More consistent pacing. The final project is allotted less time than the 4th project, while the 4th project is arguably much easier to complete.

- More one on one meetings.

- More smaller assignments

- more smaller assignments to build up fundamentals

- More TAs and more reference material for projects.

- n/a

- Not possible, but more time in the day

- Nothing much. The class was enjoyable. The professors, TAs, and CAs were helpful and I learned a ton.

- Perhaps instead of a programming based midterm, the midterm could of been about theoretical applications of C programming. For example, questions about whether something should be passed by reference or by value or what something will equal after a block of code.

- Sort of felt like, the last minute extensions/helpful tips encouraged people to do their assignments late.

- This class could be improved by perhaps doing half work and half lecture days rather than full lecture and full work days.

Course: EN.600.120.02.SP14 : 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.)	
Return Rate	21/25 (84%)

- be prepared to spend a lot of time

- be ready to devote time and effort to every assignment

- Be ready to put in time to learn outside of class

- Being good at programming helps.

- Do not procrastinate, the projects take a lot of time. The material is not too complex, but there is a lot of it. Attending lecture is only necessary for attendance points.

- Great course

- Lots of work but worth it

- Projects are fairly difficult but do-able.
Writing up the project design is important

- Start your programs early

- Students should be very responsible with their time when working on the projects or they will run out of time.

- take data structures first, it's a hard but doesn't require prereqs although they could help

- The instructors give you plenty of material and time to do well in this course. There is a lot of enthusiasm in the classroom, and it has been a lot of fun to learn to be a better programmer with all of them. Take the course!

- The Projects are really tough and you have to spent a lot of time on designing even more then coding.

- The workload is not too heavy as long as you pace yourself and always go to class. Yair and Amy are both really great instructors and want to help people do well, and the TA/CAs are all really helpful, too. Assumed background is some intro programming class, and while not required, I think some background in data structures is helpful.

- This class is a lot of work even though it doesn't seem like it

- This class is extremely helpful for programming, but it is a lot of work.

- This class is very challenging and requires a lot of time outside of class to not only learn the material well, but also to work on the programming assignments. There are many helpful resources, such as the work days, the help e-mail, and office hours. Students should take advantage of all of these helpful tools.

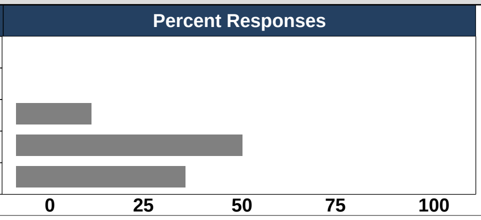
- This is a hard class and a lot of work but you will learn a ton and it is actually useful information for your future. Definitely recommend it.

- Yair's extremely fair, just don't screw around.

- You have to dedicate a lot of time to this class if you want to do well. However, if you are scared that you shouldn't take it because you didn't do well in a previous programming class and you feel like you don't really understand it, don't be. They really do start from the basics and teach you everything from the ground up.

- You will work hard for this class, but will reap huge rewards for the work you put in.

12 - This course improved my appreciation for and/or ability to engage in life-long learning.

Response Option	Weight	Frequency	Percentage	Percent Responses
Disagree strongly	(1)	0	0%	
Disagree somewhat	(2)	0	0%	
Neither agree nor disagree	(3)	4	16.67%	
Agree somewhat	(4)	12	50%	
Agree strongly	(5)	9	37.5%	

Return Rate	24/25 (96%)

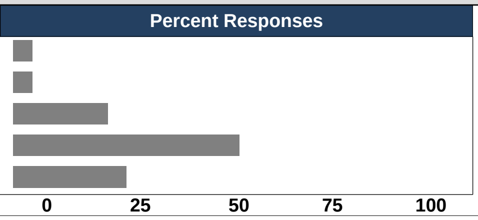
Course: EN.600.120.02.SP14 : Intermediate Programming

Instructor: Yair Amir *

13 - This course deepened my understanding of the social impact of computing.

Response Option	Weight	Frequency	Percentage
Disagree strongly	(1)	1	4.17%
Disagree somewhat	(2)	1	4.17%
Neither agree nor disagree	(3)	5	20.83%
Agree somewhat	(4)	12	50%
Agree strongly	(5)	6	25%

Percent Responses

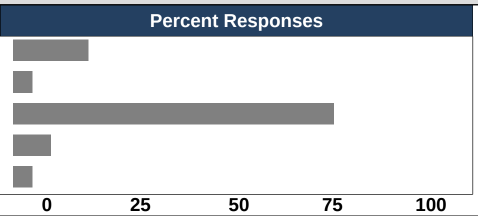


Return Rate	24/25 (96%)

14 - This course enhanced my ability to work effectively in a team.

Response Option	Weight	Frequency	Percentage
Disagree strongly	(1)	4	16.67%
Disagree somewhat	(2)	1	4.17%
Neither agree nor disagree	(3)	17	70.83%
Agree somewhat	(4)	2	8.33%
Agree strongly	(5)	1	4.17%

Percent Responses

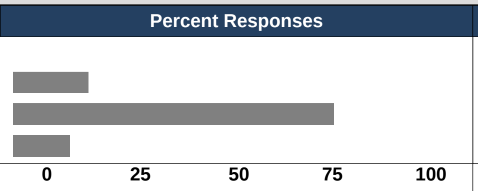


Return Rate	24/25 (96%)

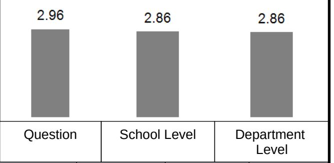
15 - How effective were the in-class activities for learning the material?

Response Option	Weight	Frequency	Percentage
Not at all effective	(1)	0	0%
A little effective	(2)	4	16.67%
Somewhat effective	(3)	17	70.83%
Very effective	(4)	3	12.5%

Percent Responses



Means

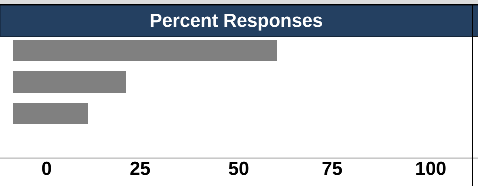


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
24/25 (96%)	2.96	0.55	3.00	156	2.86	0.83	3.00	156	2.86	0.83	3.00

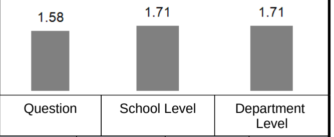
16 - When doing in-class exercises I prefer to work

Response Option	Weight	Frequency	Percentage
by myself.	(1)	14	58.33%
with one partner.	(2)	6	25%
in a small group.	(3)	4	16.67%

Percent Responses



Means



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
24/25 (96%)	1.58	0.78	1.00	156	1.71	0.80	1.00	156	1.71	0.80	1.00

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2014.Spring

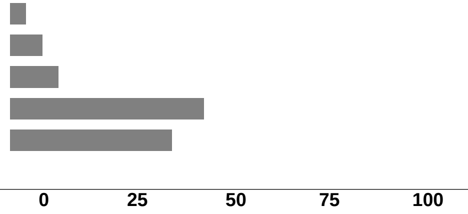
Course: EN.600.120.03.SP14 : Intermediate Programming

Instructor: Yair Amir * ,ROBERT DIPIETRO

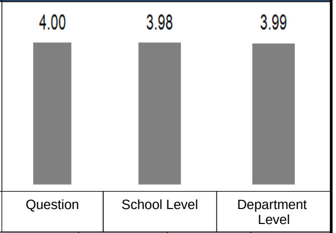
1 - The overall quality of this course is:

Response Option	Weight	Frequency	Percentage
Poor	(1)	1	3.57%
Weak	(2)	2	7.14%
Satisfactory	(3)	3	10.71%
Good	(4)	12	42.86%
Excellent	(5)	10	35.71%
N/A	(0)	0	0%

Percent Responses



Means

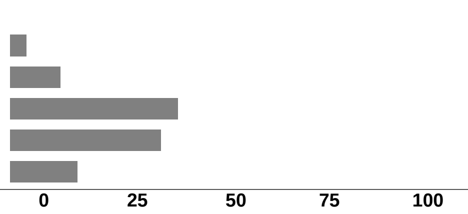


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
28/30 (93.33%)	4.00	1.05	4.00	7,431	3.98	0.95	4.00	887	3.99	0.90	4.00

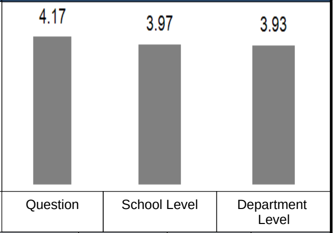
**2 - The instructor's teaching effectiveness is:
Yair Amir**

Response Option	Weight	Frequency	Percentage
Poor	(1)	0	0%
Weak	(2)	1	3.7%
Satisfactory	(3)	3	11.11%
Good	(4)	10	37.04%
Excellent	(5)	9	33.33%
N/A	(0)	4	14.81%

Percent Responses



Means

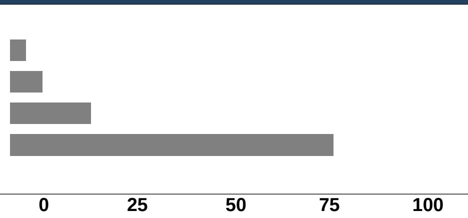


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
27/30 (90%)	4.17	0.83	4.00	8,146	3.97	1.05	4.00	956	3.93	0.96	4.00

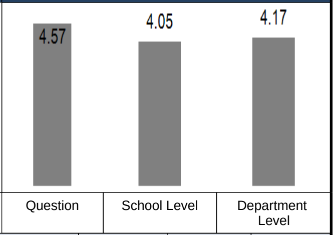
3 - The intellectual challenge of this course is:

Response Option	Weight	Frequency	Percentage
Poor	(1)	0	0%
Weak	(2)	1	3.57%
Satisfactory	(3)	2	7.14%
Good	(4)	5	17.86%
Excellent	(5)	20	71.43%
N/A	(0)	0	0%

Percent Responses



Means

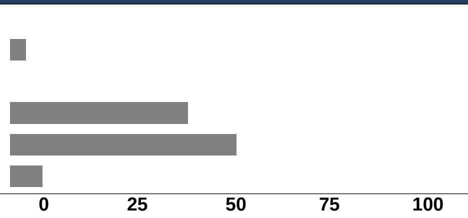


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
28/30 (93.33%)	4.57	0.79	5.00	7,379	4.05	0.91	4.00	877	4.17	0.86	4.00

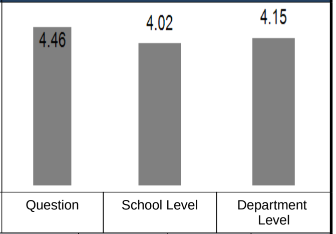
4 - The teaching assistant for this course is:

Response Option	Weight	Frequency	Percentage
Poor	(1)	0	0%
Weak	(2)	1	3.57%
Satisfactory	(3)	0	0%
Good	(4)	11	39.29%
Excellent	(5)	14	50%
N/A	(0)	2	7.14%

Percent Responses



Means



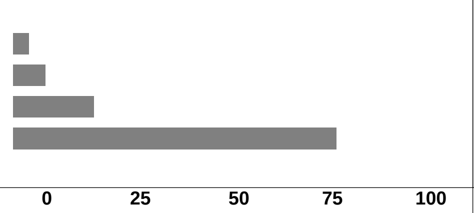
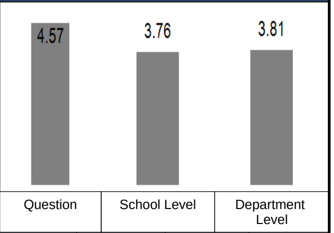
Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
28/30 (93.33%)	4.46	0.71	5.00	7,376	4.02	1.04	4.00	878	4.15	1.04	4.00

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

Return Rate	23/30 (76.67%)

- All of them
- amy
- Amy, Jeff
- Cameron Davis
- Diana
- Diana; Amy; Cameron; Andrew
- Jeff
- Jeff Dalla Tezza
- Jeff Dallatezza
- Jeff DallaTezza
- Jeff DallaTezza
- Jeff Datalllezza
- Jeff Delatezza
- Jeff Della Tezza
- Jeff DellaTazza
- Jeff, Amy, Emily
- Jeff, Cameron
- Jeffrey DallaTezza
- Jeffrey DallaTezza
- Jeffrey DallaTezza
- Jeffrey Dallatezza
- Jeffrey Dallatezza
- multiple

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

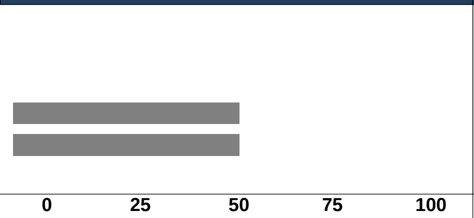
Response Option	Weight	Frequency	Percentage	Percent Responses	Means
Disagree strongly	(1)	0	0%		
Disagree somewhat	(2)	1	3.57%		
Neither agree nor disagree	(3)	2	7.14%		
Agree somewhat	(4)	5	17.86%		
Agree strongly	(5)	20	71.43%		
N/A	(0)	0	0%		

Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
28/30 (93.33%)	4.57	0.79	5.00	7,344	3.76	1.07	4.00	876	3.81	1.08	4.00

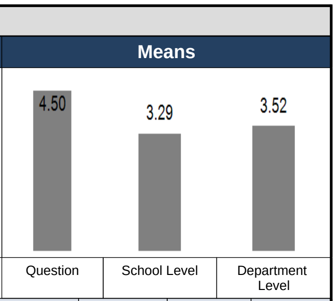
7 - Compared to other Hopkins courses at this level, the workload for this course is:

Response Option	Weight	Frequency	Percentage
Much lighter	(1)	0	0%
Somewhat lighter	(2)	0	0%
Typical	(3)	0	0%
Somewhat heavier	(4)	14	50%
Much heavier	(5)	14	50%
N/A	(0)	0	0%

Percent Responses



Means



Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
28/30 (93.33%)	4.50	0.51	4.50	7,378	3.29	0.99	3.00	876	3.52	1.01	3.00

8 - What are the best aspects of this course?

Return Rate	26/30 (86.67%)

- Absolutely loved this course; one of my favorite classes so far at Hopkins. Coming into this class, I had very little programming experience (only Intro to Java) and found myself having to try very hard to keep up with the material, but the professors, TA, and CAs are excellent and really helped me. The projects really increased my ability to think through the logic of problems. I know professors don't like project 4, but I found project 4 to be fun and good practice with object oriented programming (I haven't had too much experience with oop, I took Java over the summer and didn't have much time to devote to oop). People were also concerned about the midterm, but I think that it's a good way to test our ability to write code and think through logic without outside help. When we are doing projects, I know everyone uses professor/CA/CA help to different degrees, and it's good that we have so many resources to get help, but I think the midterm is a good way to test who actually knows the material and can think through the logic by themselves. The professors, TA, and CAs are all EXTREMELY helpful. Rob DiPietro is definitely my professor crush (I hope I don't get in trouble for saying that!). It's very obvious that all the professors are passionate about programming and it's very interesting to see how all of them have different backgrounds. Definitely gained greater perspective into the world of computer science! Yair talked about how he got a really bad evaluation last semester and I really don't understand how anybody could have such negative things to say about Yair. He is so helpful and passionate and really tries to help and teach his students as effectively as possible. Don't let the hate get to you Yair!

- All of the assistants for this course taught me something that will help me continue into the professional world.

- Challenging and you really learn from it.

- Clearly passionate teaching staff, plenty of material to digest

- Extensive curriculum

- final project

- Forces you to constantly think at a higher level. Focus on the design emphasizes this

- I gained a lot of programming experience and I have a newfound respect for good designs. I also feel like this made me think much more deeply and conceptually about the aspects of good code than I ever have before since I had to worry so much about efficiency and memory management. It was also really great that we had time to work on things in class and ask questions of TAs, CAs or professors/instructors so often.

- I learned c++ and c.

- I. don't think I could've learned much more from programming in 13 weeks. My amount of personal growth was awesome

- It is a crash course in C and C++. It moves very quickly, but covers a lot of what you need to know about these languages

- It really challenges the way think. It is hands down the best class I've taken at Hopkins. It teaches you how to think and how to approach a problem. As a computer science major, this is strongly recommended. It has a hefty workload but at the end, it is all worth it. It is so rewarding. I cannot express how much I love this class. It is the best class at Hopkins.

- learning syntax

oh Yair talks about his research, that's pretty cool. his class distributed systems also sounds cool. But this class doesn't have much to offer to anyone who already knows how to code.

- The availability of TAs and CAs.

- The best thing about this course is it allows students to become real world programmers, able to actually take a task and make what someone else describes into reality.

- The course will make you a better programmer. The assignments challenge you to think, and the thorough feedback on the assignments let you know exactly what you could have improved on, which is helpful for future assignments. All of those who teach the course truly care about you learning the material.

- The in-class work sessions were a great time to get pointers on the projects. The large number of CAs and TAs meant that it was typically pretty easy to talk to someone for help.

- The instruction is fantastic, you can really see how much effort has been put into the course over the years.

- The learning

- the projects make you really learn and are very well structured. Rob was incredibly friendly, cool, and approachable and was always willing to help. The idea of the tutorial sessions where everyone theoretically would be able to be helped. When Yair did come in to teach our class, he did a really good job and was very enthusiastic about the material. I like Yair but I know many who are scared of him...

- The TAs are the best I've had in any course hands down. This makes sense since they are handpicked by Yair but there is almost never a time where I see them incapable of helping me or others in the class.

- The tutorials are really helpful, and Rob did a really great job in lecture teaching the different data structures and functions.

- This course gives good examples to learn from and provides challenging projects that teach programming concepts well. Amy, Jeff, and the other TAs are fantastic helpers whose help was invaluable in developing my programs and fixing my bugs. Working with them during each class and talking with them 1 on 1 was great and is something I wish I had in other classes.

- Topic is interesting. Teachers and course assistants put a lot of effort into making sure student's questions are answered.

- When your code compiles. It feels amazing seeing zero memory leaks and when it functions properly.

- you learn a lot

9 - What are the worst aspects of this course?

Return Rate	25/30 (83.33%)

- A stack of error messages always seem like a great deterrent

- Difficulty of project assignments. Very hard to finish projects only with what is taught in class, much self studying is required. Assignments is very time consuming.

- Extreme difficulty

- I really did not like this class. It should be called intro to data structures that aren't useful. I learned silly data structures that were just hard to code. The class is boring and you sit there learning syntax that you won't remember all class. Very dissatisfied.

- I would say the most annoying, not necessarily worst, thing about this course was some of the way the projects were specified in the design documents. It was particularly annoying when we might get halfway through a project, already done with our designs, and suddenly they told us that something was supposed to behave in a certain way. For example: the Final Project we are told, about a week in, that delete should return a pointer and the data structure not deal with memory allocation. So this is fine and dandy, but it should have been told to us in the project description. Now why do I feel so strongly about this in particular? Imagine we get to the real world and we're given a project by a client, with certain specifications. Anything they do not specify is then left up to us. If they come back and say that the product isn't acting in a certain way that they didn't specify, it's their own fault, not ours as the programmer. This is why the projects should be a bit more tightly specified for exactly what you want, not just for some general parts of it. I understand that you want us to be able to design from high level to near pseudo code, but if you want someone to be able to design you need to give them every last specification.

- Ironically, the idea of this class is great, but the implementation is poor. The lectures were completely useless but you had to go since attendance is taken. The size of the course was too large to actually have Yair's idea of everyone being able to get help. To a lot of people I knew, Yair and Jeff often seemed over-critical and unapproachable, which is a problem since they want the course to be personal (they would help when you asked them, but they'd insult you while they solved your problems). Rob did a really good job with this being the first time he's taught the course but it was questionable at times whether he knew what was going on. The data structure for the final project is simply disgusting.

Also, the feedback that we'd get back on projects was kinda shady and hard to understand, and it was pretty unclear what they want for things like the design. Perhaps showing what a good design is would help?

- It isn't easy. that just comes with the territory.

- It often moves too fast, and the work is almost impossible to do by yourself at times

- Its a lot of work.

- Learning concepts is kind of hard. You essentially have to do it yourself on your own time. Since the classes tend to be watch the instructor, you kind of have to figure it out later when you do your own project, but the tutorial sessions are a great way to get things done.

- Most of the instruction was simply going over example programs, but they weren't always clear. It's necessary to go over the examples outside of class to really understand them.

- None

- Professor Yair should not teach this course. He has an arrogant approach to education and, while he certainly cares for his students and has implemented policies meant to benefit them, his presence is detrimental to the environment in the classroom. Seemingly incapable of listening, he cuts students off and seldom explains their questions, often lecturing them instead on personal faults that have nothing to do with their programming questions. From the course description I was interested in taking Distributed Systems but, based on my experience in this class, I will *certainly* not. I don't question Professor Yair's knowledge in his field, and I'm sure experienced programmers can work well with him, but putting him in the position of teaching beginning students is a mistake.

- project 3

- Really can't think of anything!

- So much time. I spent many late nights coding these projects especially since I tried to do extracurricular activities. If future courses are like this, I don't think I'll be doing any more performance.

- Spending most of the class time looking at blocks of code projected onto a screen gets boring after an hour and 15 minutes. Also, the CAs (excluding Diana) rarely showed up to the CS ugrad lab and at times, the help email did not give replies fast enough.

- the lectures just bombard you with pages and pages of code. it's completely useless. the examples should be a few lines each, not entire programs. make the examples more concise.

the grading is also inconsistent. the CA's are not on the same page because you will lose points on, for example, not including enough detail in your design, but then on the next project you will get points taken off because you devoted too much space to low level details and not high level description. I didn't know this was a writing intensive course.

- The projects themselves are logically challenging and when an error is made, it takes a long time to find, even with assistants.

- The very first assignment was rough. It kind of felt like jumping into the deep end before learning how to float for someone that is learning how to think about memory management and efficiency on top of a brand new language and syntax all for the first time. It was also unclear how exactly different aspects of the projects would be weighted. The packets with the code each week were also a bit confusing once we got into C++ since many programs were named after the figure numbers in the text book. It made it kind of hard to search for a specific example of code quickly, even with the descriptions in the front.

- The work

- The workload. It is a beast but it is definitely to be expected

- The worst aspects of this load is the work load. Also I did not feel prepared for the first project, I learned how to improve for the rest of the projects, but I feel that more advice about the first project would be very good. I also found the midterm particularly challenging. I take hours to code, and have to create a data structure that I had not seen before in a very short period of time, was very challenging. I would have liked to be able to learn more how to prepare for this. Because I felt the dry run didnt give me enough feedback for what I needed to do to prepare for the actual midterm.

- There's not very much time to digest the new information we're given. Sometimes we'll learn something one day and then start working on our next project (which applies that new material) the next day.

- very difficult course

Course: EN.600.120.03.SP14 : Intermediate Programming

Instructor: Yair Amir * ,ROBERT DIPIETRO

10 - What would most improve this class?

Return Rate	23/30 (76.67%)

- class is great as it is

- Clearer instructions regarding projects

- Decreasing the size, having less lectures or more small projects that people could work on in class together instead of lecture. I felt like we were just thrown into big projects without learning anything before.

- easier assignments

- Have Peter Frohlich teach in Yair's place.

- I mentioned this during my meeting , but I think some way of active learning would be really beneficial. When I was taught Python during the summer in an intensive 5 day informal course, we had to follow the instructor and that activeness paired with later practice allowed for much better learning of the topics. But in this class the project was when I first learned how to do many things. I know there is the obvious issue of this being a problem because the classes are getting so large but if there were some other way like making extremely short and easy to follow examples for each major concept a la the complex number thing for teaching making classes and overloading but even simpler.

- I think there should be more advice given during the first project, and more advice about preparation for the midterm.

- I think this class was fine as it is. However I feel like the way the syntax was taught was pretty dull, especially for those who have done this kind of thing before.

- If anything, maybe try to make the assignments more clear from the get-go. i feel like with almost every one, we were getting emails later to clarify or alter what the assignment sheet said. The lack of total clarity sometimes made things difficult.

- In C++ the packets could have labels at the top of the pages where the code actually starts to show what the program is supposed to be demonstrating. Just little things like "virtual inheritance" or "template" would be a huge help when a student is programming outside of class and needs to search through a bunch of packets to find on little block of code to figure out syntax. It would also help if you guys provided a rubric or something so that it is extremely clear that the design and code quality are major factors in grading.

- It is awesome as it is

- It is more of a do it yourself type of class, which in the world of computing you tend to do things as a team. So maybe having one team work assignment to prepare for those type of scenarios would be helpful. Also maybe restrict the time TA's spend with a particular student. Being so few in number per a section they can become caught up in one person's code for too long,so that they will not reach all of the students.

- It was a bit fast throughout semester. I believe that most students did not understand at that moment, but instructor already proceeds to next steps.

- More CAs in the ugrad lab, and faster replies from the help email.

- More interactive lectures. Many times I felt as though the code was just being explained to me. Even though we had tutorial sessions, I feel that doing some sort of example on the computer along with the professor would help solidify knowledge of syntax and implementation in languages that (for the most part) are new to us students.

- More time for assignments

- More time for the midterm.

- More time!

- not learning obsolete/inefficient data structures would be a start, since people will think that the data structures taught in the class are actually used in the real world.

more professional graders who don't take off points for petty things even when the code works accurately and efficiently

- Our homework consists of only four projects spread out over the semester. This is good in that it allows us to work at our own pace, but bad in that we get feedback on our work less frequently. There are in-class "tutorial sessions" that seem to try to mitigate this, but feedback when I'm in the middle of the project is less helpful than feedback for my finished product. So I think it'd help if they got rid of one of the projects and instead of it had mini-assignments (maybe a week/a couple days long) due a few times over the semester.

- Skip learning so much syntax and instead teach us useful things.

- Specify the projects more thoroughly, detail everything that you want done, not just some of it. Also redo project 4, which I know is already on the agenda. One idea I had for redoing project 4 was to potentially say that it's merely another issue of specifying what you want. If you say directly that you want a menu driven program than suddenly students know a bit more what you're talking about. Also the current ordering of information in it is confusing. I don't think it's an issue with the general idea, I just think the project needs to be tightened up.

- This class NEEDS to stay small and personal. This can't turn into a lecture style class. The tutorial sessions are terrific. To keep this the class needs to be advertised for what it is. A 13 week crash course in how to think as a computer programmer. This isn't intermediate programming where we just learn the syntax of a new language and a few new concepts. This class must find a way to stay the exact same size.

Course: EN.600.120.03.SP14 : Intermediate Programming

Instructor: Yair Amir * ,ROBERT DIPIETRO

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	23/30 (76.67%)

- Be familiar with C and C++. Don't try to do too much during the semester, be prepared to spend a lot of time on these projects.

- Be prepared to put in long hours for the projects.

- Be prepared to work hard, you are going to learn a lot of new things very fast.

- Definitely get started on the projects early

- Don't procrastinate on the projects. Finish them as quickly as possible, then check their success.

- Don't procrastinate.

- dont take the class, it pales in comparison to data structures. Except you have to take the class, so enjoy!

- It is pretty work intensive.

- It is super time consuming but it really is worth it, if you are interested in programming

- Know that this course requires a lot more effort than you think. It'll be a lot of late nights and head scratching.

- Pay attention in class

- Prospective students should realize that this course is pretty intense. You will learn enough to be proficient in C and C++ but you should be prepared to do a lot of in depth, conceptual thinking about what your program needs to do and how you can get the program to do it in the most efficient/effective way possible. This is not a class that should just be taken on a whim, you must put a good amount of time and effort into projects outside of class to do well. Students should have previous experience with a computer language, not necessarily C or C++.

- Really take advantage of the professors, TA, and CAs. They are great resources for help and are so helpful. Be prepared to work hard!

- start the projects early and dont procrastinate

- Start your projects early! Some of the later projects take a lot of time, especially the last two. It can be hard to recover if you don't start early.

- Take a programming class before this one. Also work in = results out

- This is a difficult course, if you don't work on projects early you will fall behind and get into lots of trouble. You have to stay one step ahead of the game in this course.

- This is a good first class in programming. Skip the Java course. Take the amount of time you think you'll work on the program, and multiply it by 3 - that's the amount of time you'll spend on it.

- This is a tough course. I wouldn't advise taking it unless you actually like to program and are already have a somewhat strong programming background. The textbook isn't needed at all, most things you need to know can be easily accessed online.

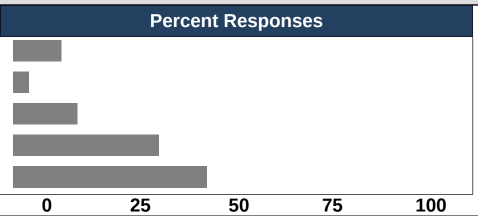
- This is a very challenging course and there is a big time commitment. It is also really important to show your code to any of the CAs, or TA, or professors before submitting because they are very helpful.

- This is much more than learning a new language and it is absolutely do-able for anyone who has the correct motivations.

- you will need to spend lots of time for this class

- You won't learn anything in class. Only on your own.

12 - This course improved my appreciation for and/or ability to engage in life-long learning.

Response Option	Weight	Frequency	Percentage	Percent Responses
Disagree strongly	(1)	3	10.71%	
Disagree somewhat	(2)	1	3.57%	
Neither agree nor disagree	(3)	4	14.29%	
Agree somewhat	(4)	9	32.14%	
Agree strongly	(5)	12	42.86%	

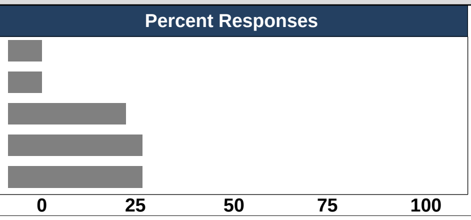
Return Rate	28/30 (93.33%)

Course: EN.600.120.03.SP14 : Intermediate Programming

Instructor: Yair Amir * ,ROBERT DIPIETRO

13 - This course deepened my understanding of the social impact of computing.

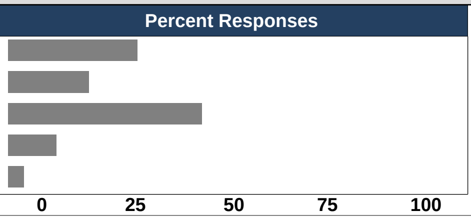
Response Option	Weight	Frequency	Percentage
Disagree strongly	(1)	2	7.41%
Disagree somewhat	(2)	2	7.41%
Neither agree nor disagree	(3)	7	25.93%
Agree somewhat	(4)	8	29.63%
Agree strongly	(5)	8	29.63%



Return Rate	27/30 (90%)

14 - This course enhanced my ability to work effectively in a team.

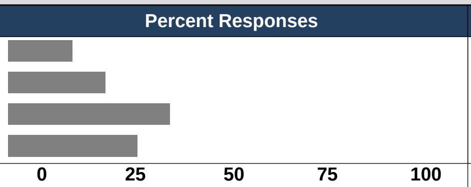
Response Option	Weight	Frequency	Percentage
Disagree strongly	(1)	8	28.57%
Disagree somewhat	(2)	5	17.86%
Neither agree nor disagree	(3)	12	42.86%
Agree somewhat	(4)	3	10.71%
Agree strongly	(5)	1	3.57%

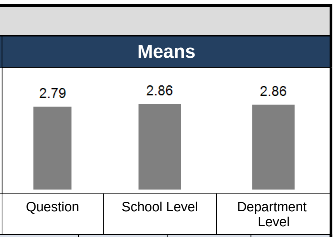


Return Rate	28/30 (93.33%)

15 - How effective were the in-class activities for learning the material?

Response Option	Weight	Frequency	Percentage
Not at all effective	(1)	4	14.29%
A little effective	(2)	6	21.43%
Somewhat effective	(3)	10	35.71%
Very effective	(4)	8	28.57%

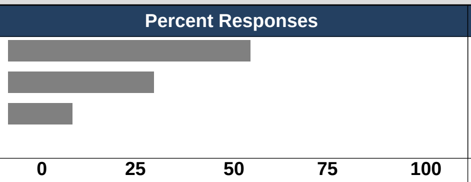


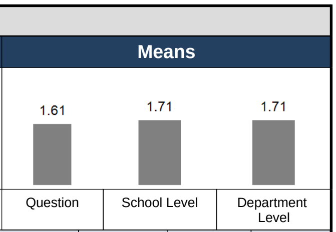


Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
28/30 (93.33%)	2.79	1.03	3.00	156	2.86	0.83	3.00	156	2.86	0.83	3.00

16 - When doing in-class exercises I prefer to work

Response Option	Weight	Frequency	Percentage
by myself.	(1)	15	53.57%
with one partner.	(2)	9	32.14%
in a small group.	(3)	4	14.29%





Return Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
28/30 (93.33%)	1.61	0.74	1.00	156	1.71	0.80	1.00	156	1.71	0.80	1.00