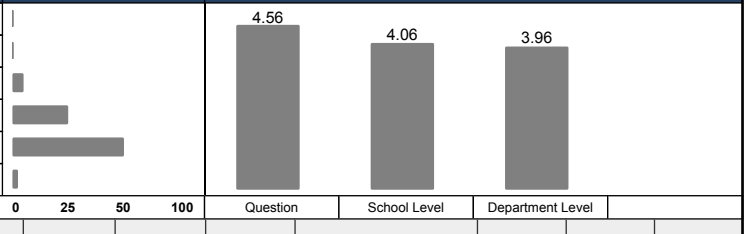
JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2018.Fall

Course: EN.601.618.01.FA18: Operating Systems
Instructor: Peng Huang *
Response Rate: 33/33 (100.00 %)

1 - The overall quality of this course is:

Response Option	Weight	Frequency	Percent
Poor	(1)	0	0.00%
Weak	(2)	0	0.00%
Satisfactory	(3)	2	6.06%
Good	(4)	10	30.30%
Excellent	(5)	20	60.61%
N/A	(0)	1	3.03%

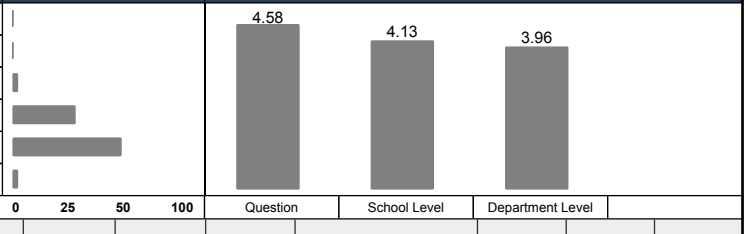


Response Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
33/33 (100.00%)	4.56	0.62	5.00	10082	4.06	1.01	4.00	1626	3.96	1.11	4.00

2 - The instructor's teaching effectiveness is:

Peng Huang

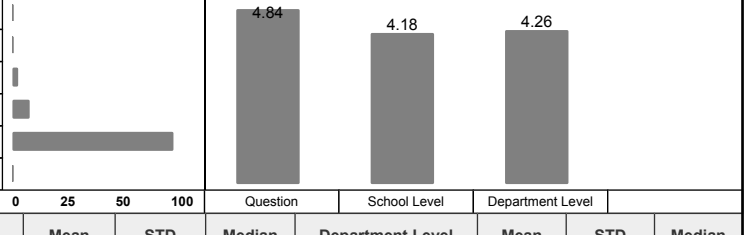
Response Option	Weight	Frequency	Percent
Poor	(1)	0	0.00%
Weak	(2)	0	0.00%
Satisfactory	(3)	1	3.13%
Good	(4)	11	34.38%
Excellent	(5)	19	59.38%
N/A	(0)	1	3.13%



Response Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
32/33 (96.97%)	4.58	0.56	5.00	10702	4.13	1.03	4.00	1617	3.96	1.14	4.00

3 - The intellectual challenge of this course is:

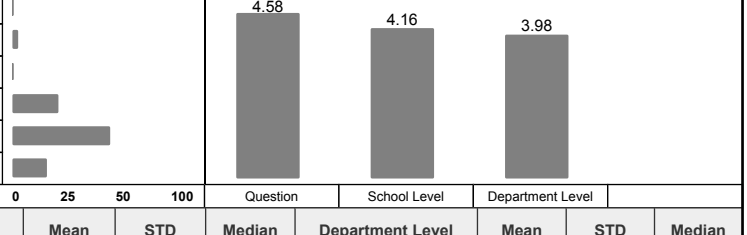
Response Option	Weight	Frequency	Percent
Poor	(1)	0	0.00%
Weak	(2)	0	0.00%
Satisfactory	(3)	1	3.13%
Good	(4)	3	9.38%
Excellent	(5)	28	87.50%
N/A	(0)	0	0.00%



Response Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
32/33 (96.97%)	4.84	0.45	5.00	10032	4.18	0.90	4.00	1614	4.26	0.92	5.00

4 - The teaching assistant for this course is:

Response Option	Weight	Frequency	Percent
Poor	(1)	0	0.00%
Weak	(2)	1	3.13%
Satisfactory	(3)	0	0.00%
Good	(4)	8	25.00%
Excellent	(5)	17	53.13%
N/A	(0)	6	18.75%



Response Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
32/33 (96.97%)	4.58	0.70	5.00	10025	4.16	1.01	4.00	1609	3.98	1.17	4.00

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2018.Fall

Course: EN.601.618.01.FA18: Operating Systems
Instructor: Peng Huang *
Response Rate: 33/33 (100.00 %)

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

Response Rate	20/33 (60.61%)

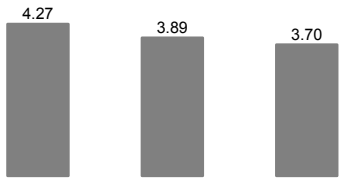
- Will Pryor
- Will
- Zach Silver, Chang Lou
- Chang Lou
- Chang Lou
- Chang
- Chang, Will
- Chang Lou
- Chang, Will, and Zach
- Zach
- Will Pryor
- Chang Lou
- Will Pryor
- Chang
- Chang Lou, Will Pryor, Zach Silver
- Chang Lou and Will Pryor
- Will Pryor
- Chang Lou, Will Pryor, Zach Silver
- Chang
- Chang Lou

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

Response Option	Weight	Frequency	Percent	Percent Responses
Disagree strongly	(1)	1	3.23%	
Disagree somewhat	(2)	0	0.00%	
Neither agree nor disagree	(3)	2	6.45%	
Agree somewhat	(4)	14	45.16%	
Agree strongly	(5)	13	41.94%	
N/A	(0)	1	3.23%	

0 25 50 100

Means



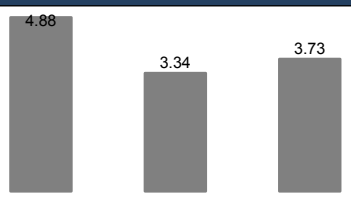
Response Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
31/33 (93.94%)	4.27	0.87	4.00	9971	3.89	1.07	4.00	1605	3.70	1.19	4.00

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

Response Option	Weight	Frequency	Percent	Percent Responses
Much lighter	(1)	0	0.00%	
Somewhat lighter	(2)	0	0.00%	
Typical	(3)	0	0.00%	
Somewhat heavier	(4)	4	12.50%	
Much heavier	(5)	28	87.50%	
N/A	(0)	0	0.00%	

0 25 50 100

Means



Response Rate	Mean	STD	Median	School Level	Mean	STD	Median	Department Level	Mean	STD	Median
32/33 (96.97%)	4.88	0.34	5.00	9984	3.34	1.03	3.00	1614	3.73	0.96	4.00

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2018.Fall

Course: EN.601.618.01.FA18: Operating Systems

Instructor: Peng Huang *

Response Rate: 33/33 (100.00 %)

8 - What are the best aspects of this course?	
Response Rate	20/33 (60.61%)

• It is an extremely challenging course which takes a lot of effort and makes you work hard to earn any sort of achievement. You learn throughout the course and the professor as well as the TAs were just awesome throughout.

• I really enjoyed the material for this course, it was very interesting and I had no previous experience in operating systems.

• Material is very relevant to current jobs in the tech industry. Instructor is extremely easy to follow, and breaks down difficult topics so that they are understandable.

• The material you learn in the course (locking, caches, etc) is useful beyond the scope of just operating systems, and the Pintos Labs are incredibly good assignments. This course has some of the best assignments in the entire department.

• You get to learn about something you've probably always been curious about as a CS major.

• Great deep dive into operating systems. Get to deal with important implementation details. Rewarding in terms of the understanding you get of how OSes work.

• Really interesting course material. Great course assistant and professor help on projects.

• You will learn a bunch

• Learning challenging and interesting item

• Great course staff. A lot is covered and learned. Very intense but rewarding.

• The professor has a passion to help you understand the material and is aware of the level of difficulty of the course. Review sessions and office hours were very helpful.

• PintOS! All of the hands-on experience working with system primitives to build some powerful features by use of indirection and other fundamental system design principles. I feel more prepared to tackle more advanced low-level systems problems. I loved it!

• The best aspect of the course was the helpful and engaging professor.

• It is mostly project based, which is fun and forces you to learn actually how to build an OS.

• You learn so much about a very important topic and it will benefit you in the future.

• I found this course to be intellectually stimulating and rewarding.

• This course throws a lot at you. It's the first time I was exposed to a giant code base that I needed to make changes to, the first time I needed to use synchronization, and the first time I was using most operating systems concepts. The projects are a huge challenge, but if you can finish them then you've really accomplished something.

• Very useful and practical knowledge. Very detailed and useful course material. Very intelligent professor. Very helpful TA and CAs.

• Learning how to build an OS through a hands-on experience. Learning how the OS works.

• The Pintos Lab assignments are very challenging. They give you a strong practical knowledge of how OS works, and at the same time you learn a lot about system design.

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2018.Fall

Course: EN.601.618.01.FA18: Operating Systems

Instructor: Peng Huang *

Response Rate: 33/33 (100.00 %)

9 - What are the worst aspects of this course?

Response Rate	21/33 (63.64%)

• Extremely challenging, takes some all nighters to get through. Will make you sometimes feel why did you take it.

• The coding projects to very large time commitment to get them done. Many times they were very open ended and the instructions either weren't completely clear or we had to work in a different order than recommended to actually successfully complete it.

• The heavy coursework. Assignments are the most time consuming I've ever had, though the work put in ends up helping you understand the OS much better.

• The Pintos Labs are a semester long group project where each lab builds upon prior ones. Therefore, a bad start means much time will be spent debugging prior labs, and issues in prior labs will double/triple/quadruple count against you if they are not adequately addressed.

• It's really challenging.

• Massive time commitment, insufficient time to complete midterm, later projects require a good amount of work done correctly before tests even start to pass.

• This is not a class. It's a lifestyle. if you take 3 credits (just this course) in a semester, you still will not finish all the work in this class. One bug in a project could take up to 5 days to fix. Also the instructions are vague enough that when you are trying to understand them (and need them to make sense) they are basically gibberish and once you have finished the project, then finally, it makes a little sense but its no longer important to figure them out. For each project, the week before it was due you are forced to spend most of the week working on the project and when the project is done you have to catch up on other work but the time it takes to catch up makes the cycle happen again. Also the grading for the project is not great. The course staff took off lots of points for not mentioning very specific things in the design documents without ever specifying that that needs to be in there in the first place. Often the question asked how did the group choose to do something but by putting down what you did, that is not the desired answer and you would loose points. Also all the lectures were taken from other professors which lead to points that often didn't relate to the other material in the course as the slides all came from different places. The course often felt like it was someone else's course that the professors was just showing us and doing no work on his own.

• you will stress a bunch

• Sometime materials are too abstract and difficult / workload...

• Homeworks are too difficult; there’s a back-breaking reliance on an instructional operating system that is honestly the bottleneck of the course. I understand it’s helpful to utilize what’s already be set up with a system like Pintos, but my group members and I felt that it just didn’t work too well for the course.

• Incredibly time consuming. High level of difficulty in understanding the material too. Often found myself lost for a week before understanding what I needed to do for an assignment.

• Not much! This course was mostly great. Unfortunately, there were instances where I was misled in office hours with incorrect information and some of the teaching staff were condescending.

• The worst aspect of the course was the intense work load.

• The assignments are very, very long. Sometimes you could be stuck for a long time with a problem, and then find a hint or answer to the question hidden among either the lengthy project description, the huge Pintos documentation, or the lab overview slides. Sometimes the biggest challenge was locating information about the assignment amongst all the resources, as opposed to solving the parts that were the actual exercise. It did, however, mean that you had to get a great understanding of Pintos just to get started, which was good.

• So much work.

• This course is very time intensive and very stressful. You will spend a lot time debugging your code, only to find that your bug is quite simple and only requires you to add one line, or change one line.

• The virtual memory section spent a lot of time covering things I already learned in Computer Architecture. From what I understand, Computer Systems Fundamentals also covered this, so I think it could have been covered more briefly. Additionally, the projects were gigantic, and often felt insurmountable. I often didn't even know where to start, and spent weeks trying to understand the codebase to realize in the last three days that my implementation was wrong and there was a much simpler way. I think this is due to poor documentation of Pintos.

• Workload is very heavy. But it's OS after all.

• don't like how later projects are dependent on work from previous projects, but probably just a personal preference thing.

• For someone who is taking OS for the first time(read Ugrads), the pace of the lectures can definitely be overwhelming.

• The course is too compressed. I feel it would be better to have a Operating Systems I and II spread over two semesters. The Pintos projects could be readily divided up to reflect the new course structure.

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2018.Fall

Course: EN.601.618.01.FA18: Operating Systems

Instructor: Peng Huang *

Response Rate: 33/33 (100.00 %)

10 - What would most improve this class?	
Response Rate	20/33 (60.61%)

• The tutorial slides for the projects could be made better.

• I enjoyed the material and lectures, but felt that the programming assignments could have been structured better to make them less of a time commitment but still allow for the same amount of learning and experience.

• Calendar guidelines/ recommended breakdowns for the project. Students would be less overwhelmed and assignments would be easier to finish.

• More TAs to handle crushload questions and requests before deadlines

• Lessen the workload by a bit.

• I think for what it is, the course was taught well. No real critique here.

• The projects need to be trimmed or changed. I would consider taking them from large group projects into smaller individual assignments. Often times there were large portions of the material in the projects that coding was unnecessary as it was already easy to understand. Also often in addition to the projects being really large, it was also hard to keep the group together (~½ of the class dropped the class) making it hard to get through the assignments. I would spend more time on how to use synchronization in practice, in addition to talking about what the synchronization primitives are as it is the most basic part of the class and needed throughout the course material. Also the class needs its own slides and projects not just ones taken from other people and repurposed. It very much felt like this class was just a clone of someone else's class from another university. The professor is great at teaching it. It just didn't feel like it was his and this is a class where it is important for the material to come from the professor as it is extremely hard and requires a lot of work.

• posting lecture notes before class so that it is easier to follow along

• lowering difficulty(?)

• Ryan is a fantastic teacher; he should create his own slides rather than adapt others. Then his abilities could really shine through and the students could learn better. Ideally, Pintos would be replaced by other more meaningful assignments, but I'm not sure how this would be done. But please, switch to something else? I didn't feel it was as effective as it could've been.

• A thorough walk-through of Pintos. Understanding of what the OS currently does / how it works, would have been beneficial to knowing how to build on top of it.

• Covering semaphores and synchronization earlier on before announcing the threads assignment. Formal design reviews, in particular for the virtual memory assignment. It would be productive to review what students plan to implement that goes beyond just a 2-minute conversation because there was a lot of time wasted from false assurances given by the teaching staff about design. Increased attention to code quality, for example evaluating modularity and extensibility in features built for assignments. It felt like my teammates didn't care about refactoring their code largely because it wasn't a part of the grading criteria. It annoyingly left me with technical debt that caused me to waste many hours debugging related issues because of poorly written code. Better test coverage on expected system behavior, which instead was left to implement after reading the documentation template. More weighted grade distribution for assignments and maybe another assignment, given the amount of time put into them.

• N.A.

• It was mostly a great class. The assignments are already pretty well laid out, and contain links to resources. Perhaps something can be done to take that further, and make sure all the resources that are provided are explicitly stated in reference to a lab section. Like "these lecture slides and overview slides pertain to this question, read the documentation for this and this, check out this site for a good explanation, and here is a selected reading that explains how this works." That is to say, having that all in one place for a single section on the lab, because it's hard to scour the thousands of pages of resources if you're still trying to figure out what the problem is. It was great that there were so many fantastic resources provided on the website and documentation linked to in assignments, but I found that many times I would spend hours thinking about how something works just to find that it's explicitly stated in one line somewhere in all these resources.

• Splitting projects into smaller projects due more often.

• I felt that the labs in this class actually made things more confusing for me. Before doing the labs I would review the lectures and read the relevant chapters in the textbook. After doing so I felt like I had a pretty good understanding of the topic at hand. I would then read the lab handout, and at this point I still felt like I had a good understanding of the topics, and usually I felt confident as a I started the assignment. However, as I started to implement things, the topics actually became more blurred and confusing in my head, rather than making the topics more clear like homeworks are supposed to. I would get so bogged down in the details of everything, and would try to debug my code so as to pass the tests I was failing, and in the process everything would get jumbled in my head the topic would become so blurry and confusing that I think it actually made my understanding a little bit worse.

• I think that better guidance on the projects would be helpful. The descriptions often leave a lot of decisions to the students, which is fine, but there should be some chance for feedback on designs before implementation. Often when discussing the design with TAs, they would respond that it's a choice we needed to make, rather than point out possible flaws or advantages. This really wasn't very helpful. I think a design review could make the projects much more successful.

• Some topics may need more detailedly lecturing.

• Better powerpoints for labs.

• 1. The instructions and Questions in exam frequently had not so clear language, and the exact things asked in a Q with big paragraph were not highlighted. 2. The advanced topics should have been probably discussed with more time, and in a bit more detail.

JHU - Krieger School of Arts & Sciences / Whiting School of Engineering
ASEN.2018.Fall

Course: EN.601.618.01.FA18: Operating Systems

Instructor: Peng Huang *

Response Rate: 33/33 (100.00 %)

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.)

Response Rate	21/33 (63.64%)

• If you are taking the course to just fulfill the systems requirement, don't. There are other courses to do so, this is a much more challenging one. If you really want to learn OS and core software basics, this is the best course one can take in CS at JHU.

• This is a very rigorous course in terms of time commitment and you must start projects very early and invest a considerable amount of time if you want to be able to be successful in this class.

• Need to prepare to spend a lot of time meeting with your partners. We found that paired programming style worked best for us— seems wasteful in short term but saved us a LOT of trouble and miscommunication in long term.

• You must do better in writing documents THAN programming.

• C and CSF

• This is one of the most time consuming courses offered at the Johns Hopkins University. For those who persevere, however, it is also one of the most rewarding. Do keep your schedule free of other difficult/TIME CONSUMING courses though, or you're in for a Bad Time.

• It's really, really difficult. If you're planning on taking this class, please make sure to go to all the lectures.

• Familiarity with C is a must have. Challenging but rewarding. Good professor. Helpful to come in with a group you can work with, since almost all work in the class is group project based.

• This class has the most work of any class you have or will ever take. The projects are in C and need constant and large scale debugging that can take from a minute up to a week. Be prepared for a class where you have to stay on top of the course material and work or you will never be able to catch up. Expected background in C and low-level programming.

• Be prepared to devote ample time

• A TON of work.

• Do your reading, ask questions, don't let yourself get lost. Take it with lighter classes around it because it takes TIME and insane brain power. Also, you work in groups, I highly suggest people you know and you feel confident can contribute to the course projects.

• This course is as hard as it is rewarding. If you put the appropriate time into it, you'll get a lot out of it. Some of the bugs and edge cases can take you days (weeks?) to debug and solving those problems helps you grow as an engineer. Ryan encourages his students to tackle the unyielding complexity of operating systems with a friendly attitude. I distinctly remember a moment when he said we should panic, to use the panic and assertion functions to confirm our assumptions about the system, and not to actually panic, by getting upset. He meant for us to hone our debugging and design skills while at the same time acknowledging for us that it will be an uncomfortable process. This really showed to me how much he cared about the learning process and his students' growth. While most of the course was great, I thought there was some disconnection between the lectures and assignments. In particular, the material on synchronization was covered too late in the course causing unnecessary confusion in the threads assignment. Also, time put into the assignments and doing them well isn't fairly represented in how much their worth for the grade in the course. Good luck!

• Students should prepare for a difficult yet rewarding class. A strong background of C is a must.

• I HIGHLY recommend every student work on a team with at least one other partner for the assignments. I don't think it's possible to do these labs individually, unless you aren't taking any other classes. You must know C, or be able to get back up to speed with it quickly (once you do the first assignment, things start coming back to you). The amount of work required for the assignments in this class is comparable to any of the other upper-tier CS classes, so plan accordingly. The course is not nearly as math-heavy as other top classes, but the programming and debugging workload is heavier than any other class I've taken.

• You better be prepared to spend so much time on this course. No part of this course is unfar, but it's a lot of work.

• This course is very tough and time consuming. However, if systems programming is something you are interested in this course is very cool and stimulating.

• If you don't know how virtual memory, threads, or file systems work, then this course is essential to learning them. It's difficult and time consuming, but there are valuable aspects. I'm not sure how much the projects helped me understand operating systems, but they definitely taught me how to navigate large code bases and use synchronization. Also, the teaching staff is generally helpful, you just need to make sure you can ask direct questions.

• Super heavy workload!

• Don't procrastinate with the projects. Choose good teammates. The tests and in course material are similar to other high-level computer science courses in terms of difficulty, but the projects take up an insane amount of time due to the fact that you're debugging a whole system that three different people code separately.

• 1. In case you don't have interest in systems, don't take it. This course requires a lot of effort and patience when you get stuck (and believe me, you will get stuck). 2. Be prepared that you will need to visit office hours of CAs/TA/Professor. So plan accordingly. 3. Don't underestimate you design of the system, design doc is worth 40% of your lab and gets scrutinized by the CAs and TAs extensively. These guys are very very thorough in your checking.