School of Computer Science & Engineering

COMP9242 Advanced Operating Systems (AOS)

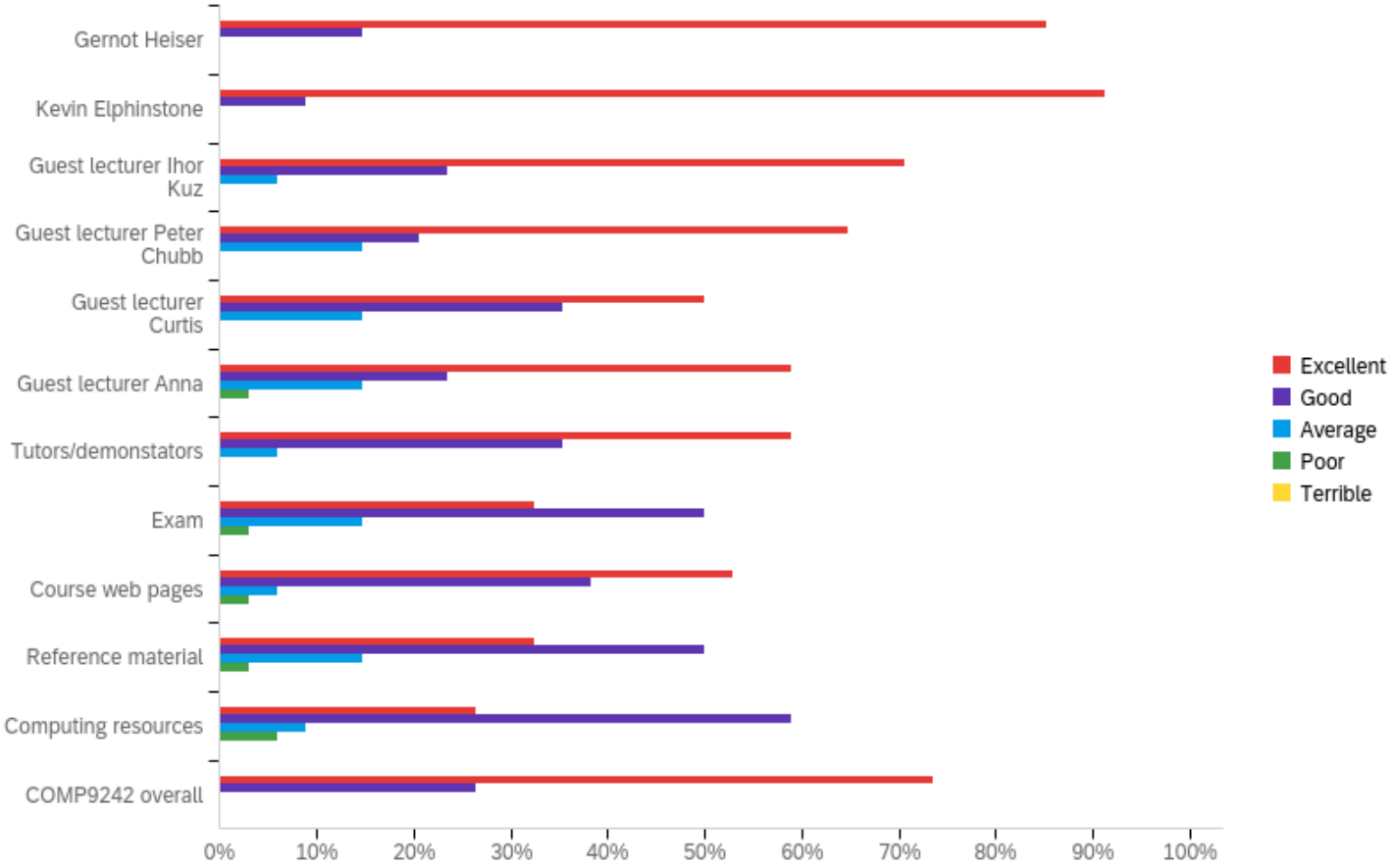
2023 T3

AOS Course Survey Result

@GernotHeiser

Copyright Notice

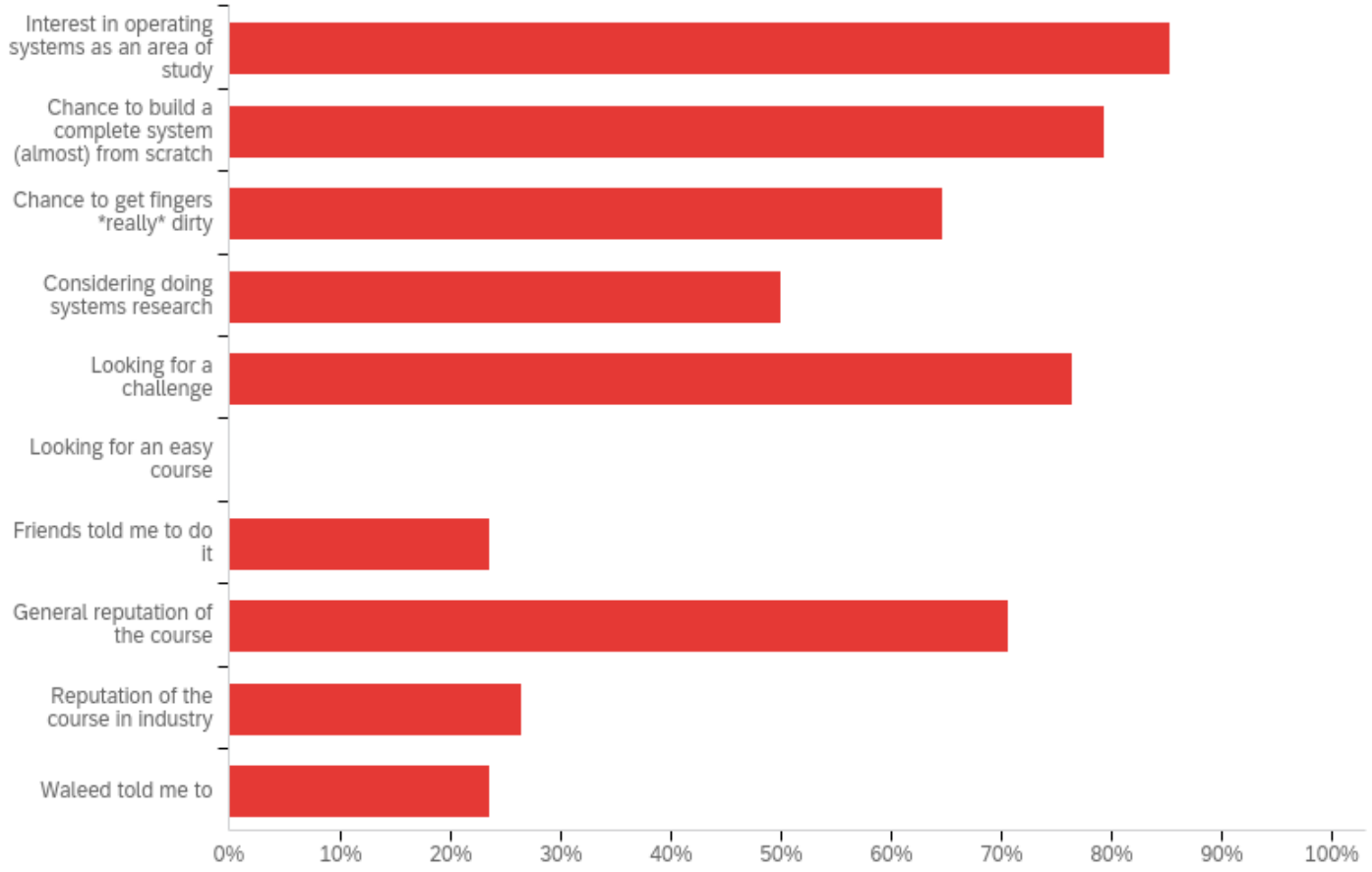
Q1: Quick evaluation



Q1: Quick evaluation

Question	Mean	Excellent		Good		Average		Poor		Terrible		Total
Gernot Heiser	1.15	85%	29	15%	5	0%	0	0%	0	0%	0	34
Kevin Elphinstone	1.09	91%	31	9%	3	0%	0	0%	0	0%	0	34
Ihor Kuz	1.35	71%	24	24%	8	6%	2	0%	0	0%	0	34
Peter Chubb	1.50	65%	22	21%	7	15%	5	0%	0	0%	0	34
Curtis	1.65	50%	17	35%	12	15%	5	0%	0	0%	0	34
Anna	1.62	59%	20	24%	8	15%	5	3%	1	0%	0	34
Tutors/demonstators	1.47	59%	20	35%	12	6%	2	0%	0	0%	0	34
Exam	1.88	32%	11	50%	17	15%	5	3%	1	0%	0	34
Course web pages	1.59	53%	18	38%	13	6%	2	3%	1	0%	0	34
Reference material	1.88	32%	11	50%	17	15%	5	3%	1	0%	0	34
Computing resources	1.94	26%	9	59%	20	9%	3	6%	2	0%	0	34
COMP9242 overall	1.26	74%	25	26%	9	0%	0	0%	0	0%	0	34

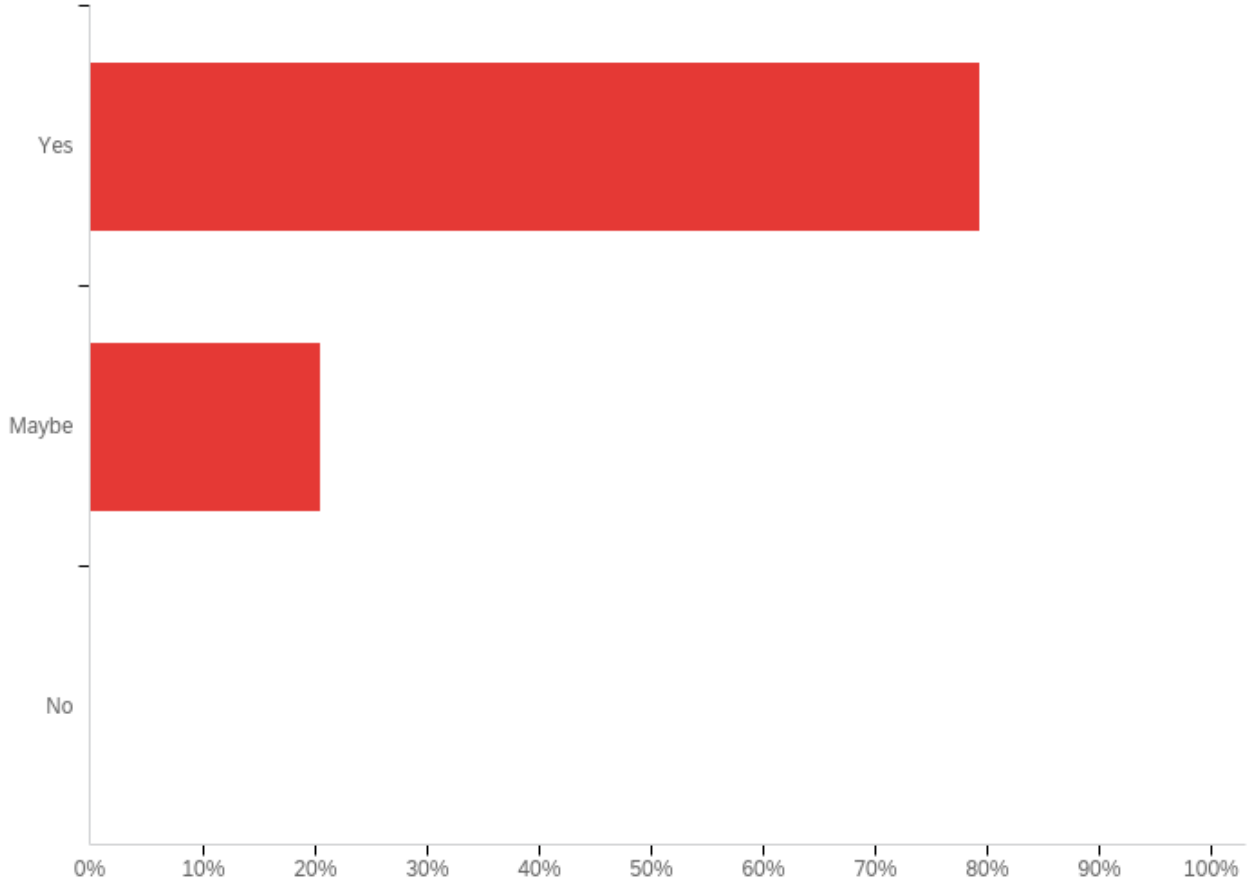
Q2: Your main reasons for taking AOS?



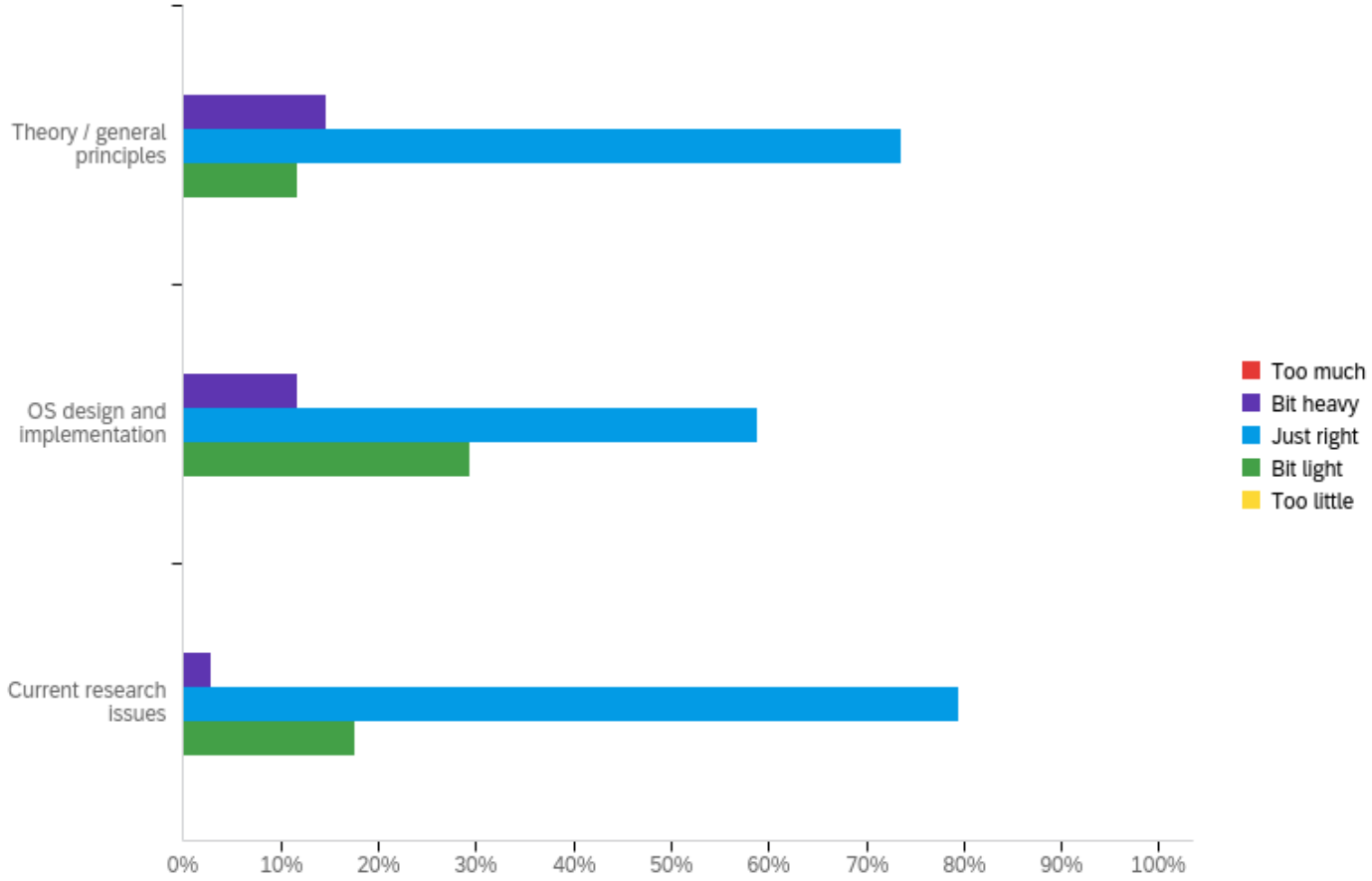
Q2: Your main reasons for taking AOS?

Other factors not mentioned above
BenB told me to. Felt left out in OS tutor cohort. Wanted a T-shirt. Gernot thought I'd already taken it.
Recommended by Lucy

Q6: Would you recommend this course?



Q7: Content Balance



Q8: The best things about AOS? (1/4)

What were the best things about this course?
- One of the few university projects where design decisions truly matter. - Interesting lecture topics. - Generous late penalties.
Nearly everything!
Really relevant and interesting lectures Significant project of large scope, felt like an excellent capstone for the courses I'd taken (OS+networks+rust+compilers), far better than 3900. Great freedom to explore different designs for OS and OS internals.
Very interesting project.
Getting hands on experience with building everything, thinking about design tradeoffs along the way and understanding interactions between different components
Very supportive and friendly course staff/tutors. I wasn't expecting this, considering the reputation this course has for being difficult, but it was a nice surprise. Course content feels incredibly interesting and contemporary.
A very practical and hands on course.
Lecture content was interesting and fast-paced. Project was challenging and rewarding.

Q8: The best things about AOS? (2/4)

What were the best things about this course?
Lots of freedom in how to do the project
The t-shirt
A great hard-course for people who want a challenge, but is still very achievable and a well scoped project. Great support from the tutors during the lab sessions.
Engaging content and general challenge of the project
Significant amount of coding = you learn a lot. The lecturers clearly are knowledgeable and care a lot, and there is enough support by the tutors. It was painful but also fun when the system actually works - it probably seems like a trivial amount of work for someone who doesn't know better but they don't know :)
Sense of achievement. (one of) the only COMP courses worth showing up to lectures for. Ample revision material for exam, which is needed as reviewing scientific writing will be new to the vast majority of students. It is eye opening how dishonest some of the papers were. Certainly something other disciplines does better. Anyone surviving this course will almost certainly be fine in industry.
The quality of lecturers was really high across the board. I especially enjoyed Ihor’s multi-core lectures. Gernot shouting drinks after some lecture was really cool too :)
Term-long project where actually had design choices you could make.

Q8: The best things about AOS? (3/4)

What were the best things about this course?
The lectures were a good mix of interesting and useful. The project was also a great experience in designing a large software project and making impactful choices. A sense of communal suffering was also very comforting throughout the term :)
Chance to implement a working OS as part of a university course, also was fun to make important design decisions. VERY lenient late penalty was very nice.
The freedom to make design decisions, make mistakes and debug; with the support of knowledgeable staff. Most of the lecture content was also engaging and provided a good look into topics I hadn't known much about prior to the course (i.e caches, multicore).
The lectures covered a really wide range of incredibly interesting topics
The challenging project and the lectures introducing different research topics
Really enjoyed the challenge and content covered. The collaborative nature with your partner but also the wider cohort was really fun and helped me learn a lot.
I love the low level programming. I feel the project balances getting good experience designing and developing an OS, without having to worry too much about the low level details (which seL4 abstracts)
It helped me become a more well-rounded developer with deeper reasoning behind my design choices.
Tutors. Great sense of community in the cohort

Q8: The best things about AOS? (4/4)

What were the best things about this course?
The fact that the course uses a world-renowned microkernel as the basis of the Simple Operating System, which means that our knowledge may actually have relevance in the industry, especially if and when seL4 gets adopted by many major corporations.
Having to seriously consider the tradeoffs in designing and implementing OS components in project concerning complex performant vs simple and extensible designs. Learning about the fundamental principles of microkernels and gaining understanding by working with a microkernel in the project.
project, lectures (really liked synchronization, multicore lectures and ones on the design of microkernels)
the project in general (though don't take my feedback into account too much, since I didn't engage with the theory as much as I should have - I could have got a lot more out of the experience by doing so)
The project itself was very satisfying
Chance to get hands real dirty with real OSdev on real hardware
free tshirt
Yes
Get really hands on with os stuff Building such a large system from scratch

Q9: The worst things about AOS? (1/5)

What were the worst things about this course?
- Some questions on the forum went unanswered. - Odroid reliability - Documentation for some components of seL4 was incomplete. - Lab support was lacking since marking was prioritised.
The project milestones greatly varied in difficulty, whilst all individually engaging and interesting, different milestones required wildly different time investments to complete
At times unclear documentation on what is required for each milestone and what constitutes a showstopper. Unreliable infrastructure requiring nursing to work effectively (and if it broke, became a Pomodoro timer to make you take a break).
Lectures trying to cover too much stuff in too little time.
Getting stuck on extremely obscure bugs that result in behaviours difficult to pinpoint, eg non determinism :(
Minor technical issues: 1. Audio quality for recorded lectures was pretty inconsistent. 2. Odroid/netcon technical issues. The exam papers felt a bit more unrelated to the course content than in past exams.
The number of milestones are too heavy for the trimester system. Special consideration requests took very long, or had no reply.

Q9: The worst things about AOS? (2/5)

What were the worst things about this course?
Odroid netcon and serial issues. The bonus task spec was a little vague.
Getting started is hard. Resources on using sel4 for beginners I found to be lacking, and introduction to seL4 lectures were aimed above my skill level coming into the course.
Writing lots of C (I should've used C++)
The lectures (and exam to an extent) required a lot of prior knowledge to understand that wasn't covered in OS or any other prerequisite - particularly about hardware. While many students in the course will have learned this content via osmosis from simply being interested in this stuff, I'm not really a nerd about computers :) so felt very behind in lectures.
Some docs could be clearer
I don't really have any negative comments regarding how the course is run. The only reason I would recommend NOT taking this course is if you aren't willing to sacrifice other commitments during this time, because you certainly will need to, and this is not really a fault of the course. We fell behind because of external factors and just couldn't ever recover from the late penalties, which seems to be a common occurrence amongst complaints of this course, but I can definitely understand why the teaching staff enforce the late penalties in the first place. If you take this course though, you cannot have anything else go on in your life, which is sometimes not always realistic :)
odroids did not always play nicely.

Q9: The worst things about AOS? (3/5)

What were the worst things about this course?
Taking this course is so ruinous to the rest of one's life I'm not sure I could recommend this to myself if I were to go back in time. Students can expect to take at most one other relatively easy course. We do not expect high marks to be easily earnt, but I suspect the average mark handed out is significantly lower than the average WAM of students who finish the course. Plus, the academic hit incurred in other concurrent courses makes the course hard on WAM. The exam hurdle is unconscionable - no one passing the arduous project should receive a UF. But I grant that it is not a hard exam just to pass, and anyone who has done a reasonable amount of revision should not have an issue. Odroid TTTTTTTT what is that
Some lab demonstrators were fantastic but there were a few who didn't seem interested in helping much. One specific example is when a lab demo questioned why we'd refactored the starter code a certain way for one of the milestones because they hadn't done it that way when they took the course. They didn't end up helping us at all beyond suggesting that we look at the manual. Responsiveness on the forum is also quite poor. The odroids were generally unreliable. Seeing T T T T when loading the boot file at 2am when you want to test an idea you've suddenly had is truly cruel.
Lack of clarity in various design requirements as listed on the project website. Attempting to clarify w/ tutors (often did not know) or forum (no response) also made it difficult. Odroid serial/netcon very flakey, made it very difficult to work on tasks as sometimes the best solution was to wait 24h.
The workload across 10 weeks is absurd

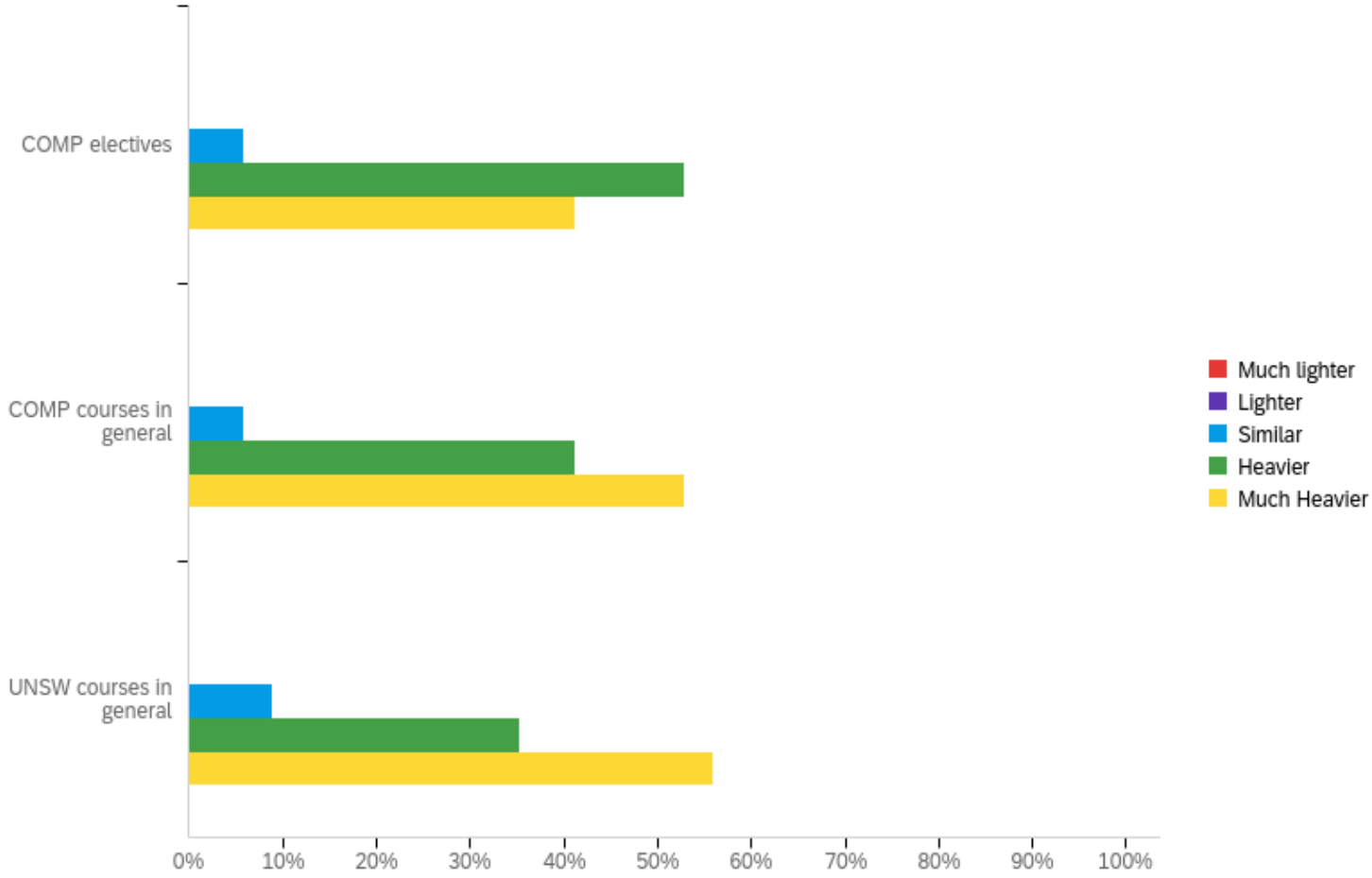
Q9: The worst things about AOS? (4/5)

What were the worst things about this course?
Despite expecting a high workload, it still was very much a shock to the system. Some of this was self-inflicted, taking on more of the project work and other work (other courses + tutoring) than I should have, but it was still rough. I think this comes down mostly to personal preference, I just don't have the brain for 24/7 grinds. Can't even really blame trimesters for this one, the timeline looked just as tough with semesters but longer!
The disconnect between lectures and the project/exam. Also a broader discussion on OS implementations and how they approach problems similar to the ones we face in the project milestones would have been a nice addition, as I found the Linux internals lecture just brushed on this.
T T T T T T T
The time pressure is real and quite challenging. Getting questions answered on the forums was not very timely and given how many people are in the course even going into the lab times wouldn't guarantee you'd get some face to be able to ask questions.
Connection to the odroids was a bit dodgy. Many times the network console stopped displaying output. It was hard to know when it was an infrastructure issue or an issue with our code.
Stressful at times due to the challenges and workload. Sometimes I would have to focus purely on getting the job done than finding the best possible solution.
The unnecessary pain and suffering caused by the lack of debugging features.

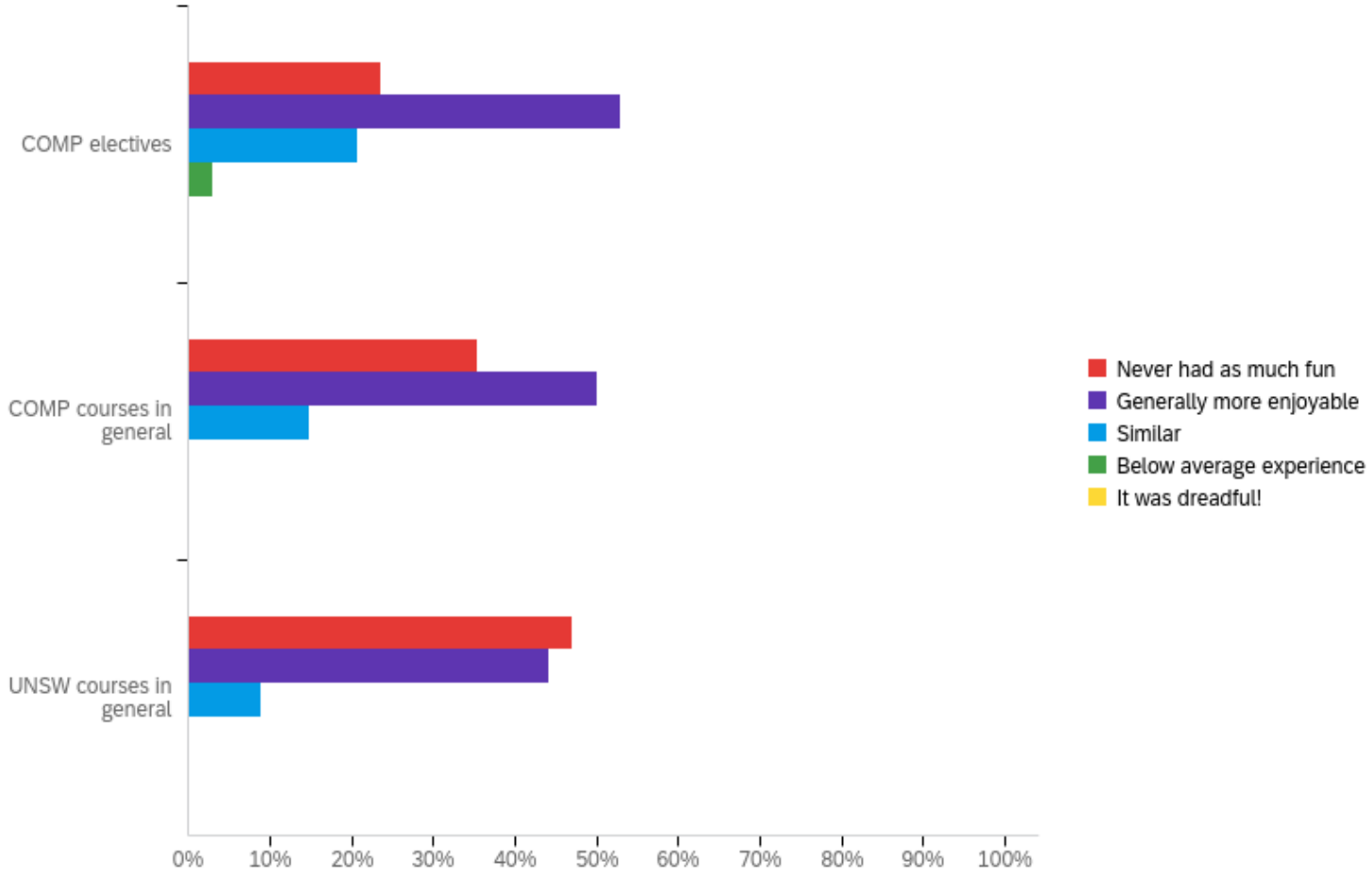
Q9: The worst things about AOS? (5/5)

What were the worst things about this course?
The strict submission deadlines (mostly earlier in term) with an already heavy workload.
Lab time was short, difficult to get help at times because too many groups needed marking.
seems like most of students are coming with a partner from os course, so more centralized system to find a partner for those who don't have one would be beneficial
odroids dying; though notably this was a minor inconvenience, usually back online within a few mins
I would like to spend more time in some of the parts of the os we didn't do, but trimesters make it hard :(
Sometimes hands can get too dirty and drop shits on ground. E.g. printing a stacktrace crashing the OS, mysterious VM fault in mutex library, etc :))
24 hour exam is super draining
Yes
Such little debugging tools Sudden spikes in difficulty (virtual memory milestone)

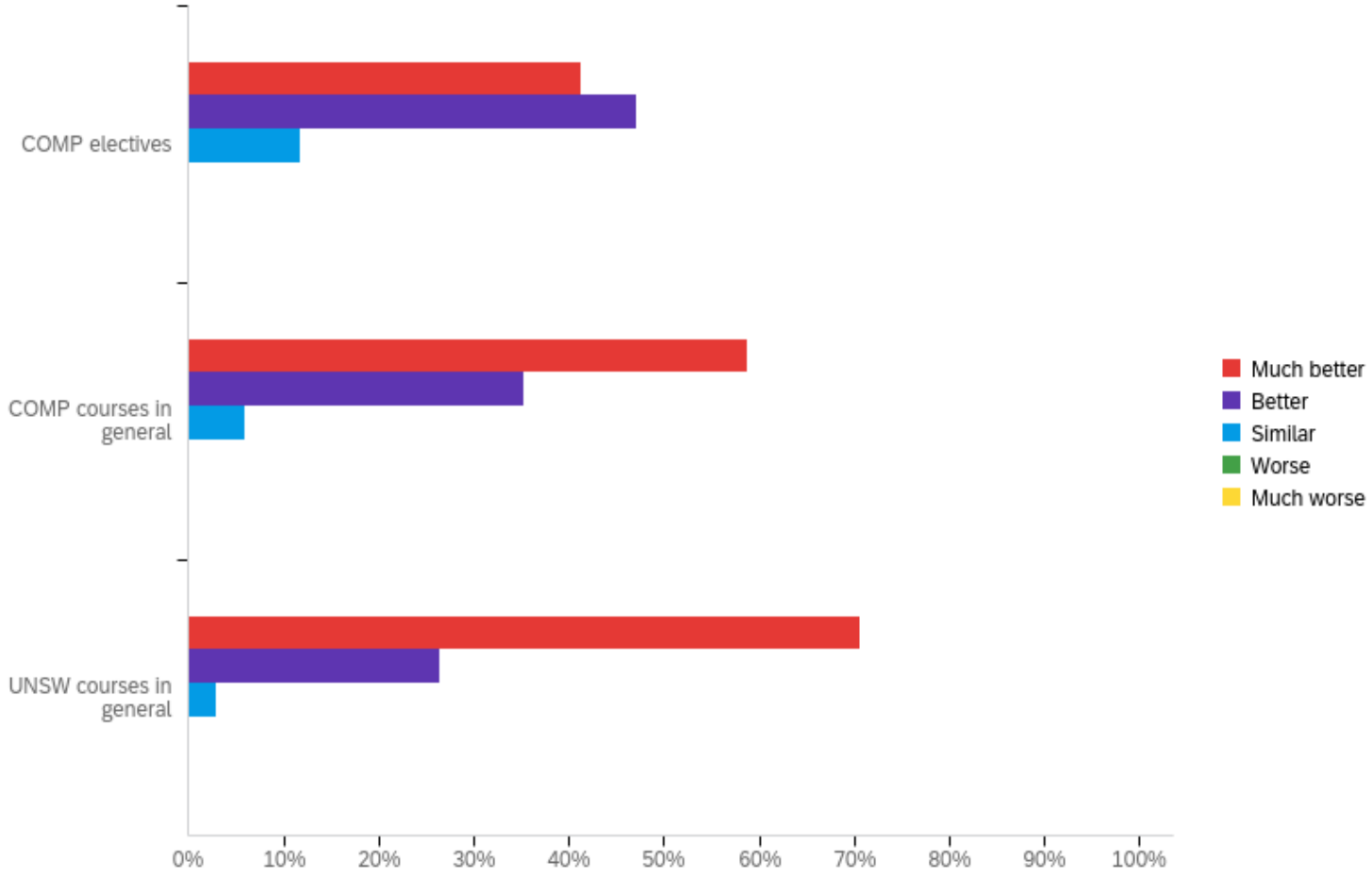
Q10: How does the workload compare?



Q11: How your experience compare?



Q12: How does quality/value compare?



Q13: Required background (1/3)

What background knowledge do you think you were missing that would have helped you in this course? Is a distinction grade in COMP3231/9201 a suitable preparation? Is it too harsh?
Prereq is largely reasonable. Maturity also definitely helps. If you want to use more papers about rust perhaps add COMP6991 as a prereq lol
OS was helpful, particularly the assignment on VM meant that this VM milestone could have been done in a week.
See previous comment - lots of assumed knowledge about hardware and prevalent systems/architectures that is not covered in COMP3231 (e.g. I didn't know what Intel was coming into the course).
I think less time between my completion of the prerequisite and taking the course would have helped, to stop losing as much familiarity
I had a HD in normal OS, and still found the content in AOS pretty out of reach. I don't think the grades in COMP3231 will be that representative of performance in AOS. Things happen to students, and marks don't tell a full story. But the reasoning makes sense, so I wouldn't say it's too harsh or anything. It felt like literally everyone else had prior background knowledge coming to AOS that wasn't in OS, such as demand paging, for which the course website provides almost no extra information. Sometimes, it wasn't quite clear to me how other people *were* getting this extra information nor how long it took for them to acquire the information. I'm also not sure how you could improve this aspect, or whether other students feel the same ;) Some lecture content overlaps between the two, so that isn't the issue, the coding aspect felt the most challenging to me (huge jump between OS and AOS e.g. Milestone 3 and 5). It felt like other students also had past friends who did the course who gave helpful tips, but not sure.

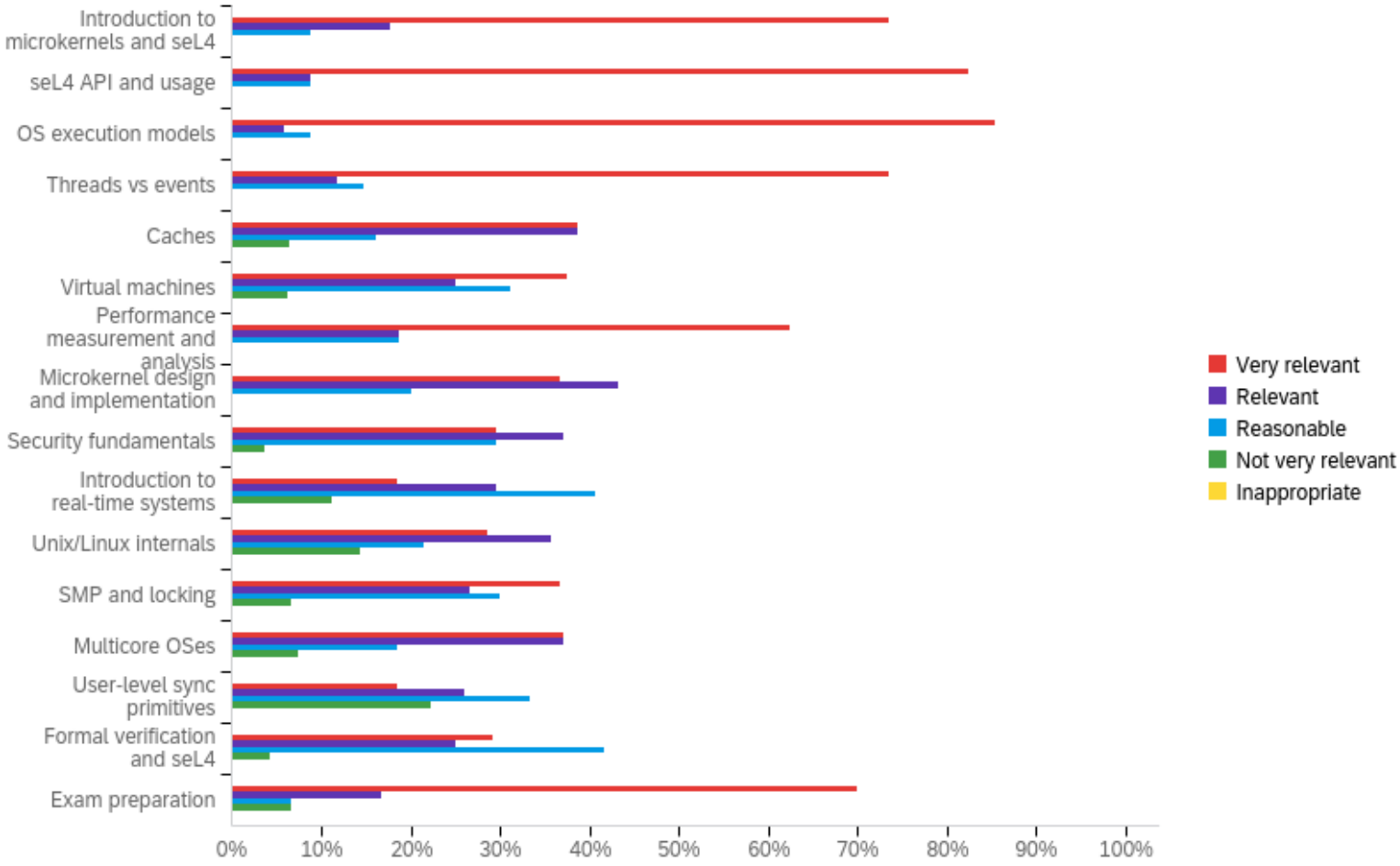
Q13: Required background (2/3)

What background knowledge do you think you were missing that would have helped you in this course? Is a distinction grade in COMP3231/9201 a suitable preparation? Is it too harsh?
This is reasonable preparation in my opinion
DN is probably not enough, but the prerequisite should not be raised for keen students/those who have improved. But displaying a similar performance to a DN grade in 3231 will probably not end well here.
I think COMP3331 (networks) would've been good to have done before this course just so I'd have been familiar with concepts like latency and throughput. A distinction in 3231/3891 is definitely not harsh. I received an HD and struggled to come to grips with seL4 in the first half of the term.
I think I was as prepared as I could have been for this course, but you don't know what you don't know :) The difficulty (for me at least) was time-based not knowledge-based. As heavy as the course is in OS-content, the bigger barrier to entry in my opinion is coding maturity. This course and OS are very different beasts, so it's hard to say if a certain grade in OS will correspond with good performance in AOS. I think the course's reputation and first 4 weeks are enough of a filter as is. The distinction requirement is unlikely to trip up anyone who could make it past those already. Overall, not too harsh, but also not super indicative that someone can handle AOS.
Unfortunately not sure.
I strongly suspect that aptitude of COMP students shouldn't be limited to OS/EOS, but rather the grade average overall needs to be a distinction, and OS/EOS is a prerequisite.

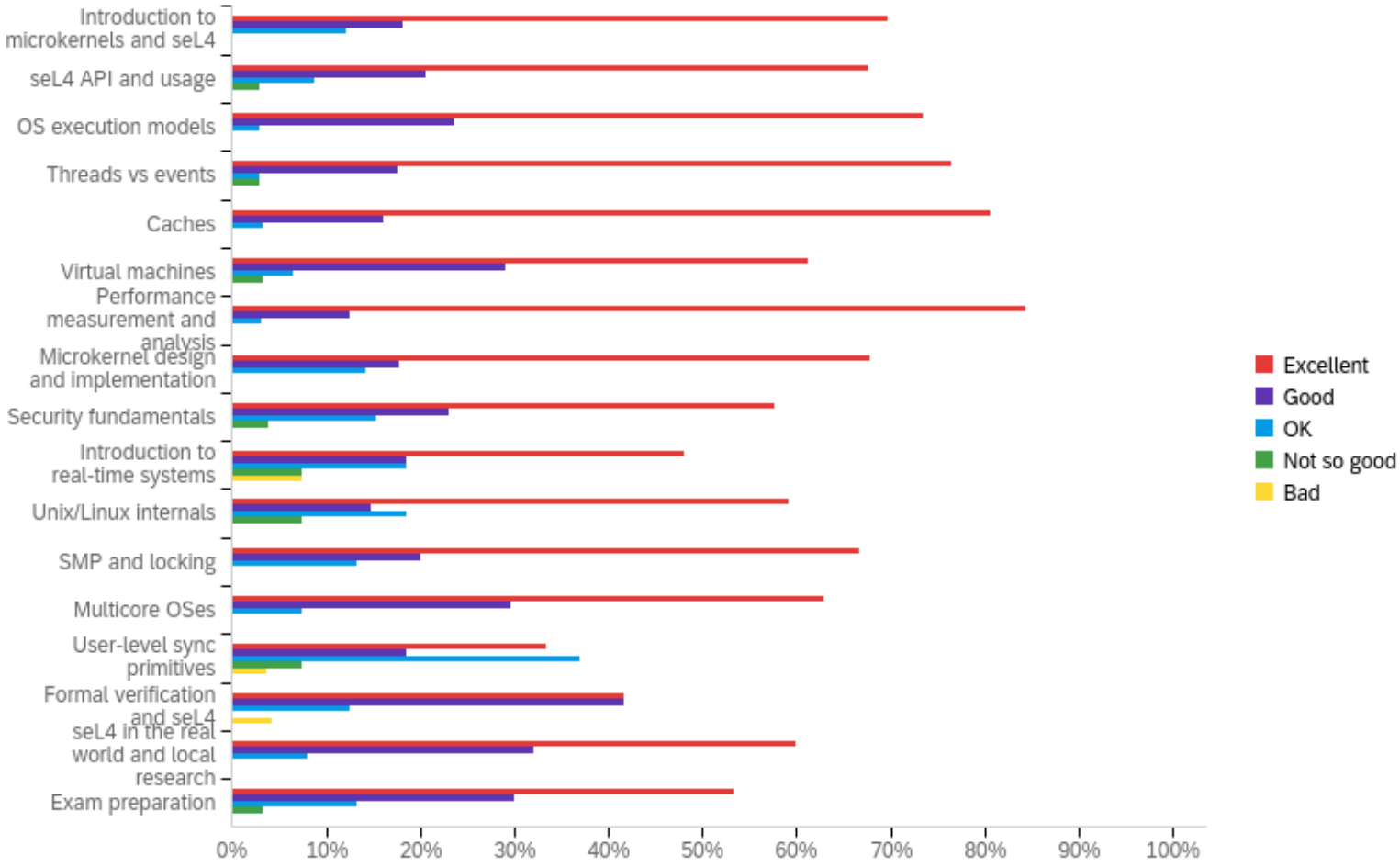
Q13: Required background (3/3)

What background knowledge do you think you were missing that would have helped you in this course? Is a distinction grade in COMP3231/9201 a suitable preparation? Is it too harsh?
Doing well in COMP3231/9201 I would say is a bare minimum to completing this course. It's obviously hard to determine how competent someone is off of their grades alone but beyond just having good programming and theory knowledge this course required a lot of work to just understand how things work in a short space of time.
I needed to be more proficient with programming in C. I am a Masters of IT postgraduate student and the main exposure we have with C is from Data Structures and Algorithms course, which doesn't quite have us ready to have the practical skills ready for this course. I think the distinction grade is about right. I wouldn't recommend this course for those without a distinction grade and above because it reflects that students should be ready to make a significant commitment to the workload.
I got a HD in COMP3891, and I still wasn't prepared for the onslaught of this course. But I guess getting my hands dirty in the code really helped to reduce my suffering.
A DN in OS/EOS is not harsh at all, in my opinion, it is quite a lax barrier of entry into this course. A very high level of maturity will be needed to do remotely well in this course. I think more general knowledge in how compiler setups and industrial strength OSes work before going into AOS would help.
Yes

Q14: Topics relevance/appropriateness



Q15: Quality of the lectures



Q16: Most useful material (1/3)

Which material do you think will be most useful to you in the future?
- Performance Measurement and Analysis - Hardware considerations – Caches and Devices - Virtual Machines
Performance analysis
Multicore, Linux internals, RT OSes, security, benchmarking (crimes)
Personally, I think the OS execution model section really interests me and give me a lot of ideas.
Caches, VM, benchmarking, UNIX/Linux internals
Performance measurement and analysis.
Threads vs event
Caches, virtual machines, real time.
Hard to know for sure, but a general knowledge of a wide array of areas will hopefully be generically useful
Not sure
Formal verification, security - those aspects relevant to other areas of software engineering
Research and real-time, hopefully
I enjoy the abstract concepts and performance benchmarking the most.

Q16: Most useful material (2/3)

Which material do you think will be most useful to you in the future?
Probably not much but that is irrelevant, good to know this stuff
Multicore, security and performance
unsure what will be my future
Hard to say given I don't know exactly what my future holds yet :) If going into research a lot of the academic skills and technical learning skills will be super helpful. A lot of the design stuff will stick with me, the need to consider all the implications of a design and practice employing this are a major takeaway for me.
Caches, locking, user-level sync primitives. The lecture on priority inheritance and inversion was really interesting and practical, so was Kevin's lecture on spinlock algorithms and how they interact with hardware and caches.
Caching, SMP and multicore systems.
Unix/Linux internals, user level sync primitives
Performance measurement and analysis
Security was a surprisingly dense and useful topic. It helped explain a lot of the reasoning behind the way certain aspects of the system were designed (it also provides background to the seL4 capability system which seems unintuitive until you under security models).
Execution models, caches, virtual machines, performance analysis, multicore OSes

Q16: Most useful material (3/3)

Which material do you think will be most useful to you in the future?
General knowledge of microkernels. This is valuable because even if we work with monoliths in future such as Linux, we have a greater appreciation of the trade-offs we face and the best design choices to make for the given use scenarios. I really enjoyed learning about multicore topics. I feel like this leaves me closer to the forefront of current software/hardware research and ready to progress into the future. It's also just really fun to learn about.
Most likely the security fundamentals lecture because I'm going to be working in cybersecurity next year.
The material on SMP and multicore (I still believe in multi-thread execution models over event based, with the heavy adoption of many core CPUs building scalable systems will require good design of locking primitives). The microkernel material should also serve to be very useful.
Execution models
OS execution models (and events, threads), hardware , VMs, SMP and locking, and benchmarking
everything i watched was useful, though I really didn't engage with the theory as much as I should have
The coding generally
Most tbh :))
more lecture content at the beginning of the course explaining relevant sel4 features for the project
Perfoamcne analysis, locking stuff,

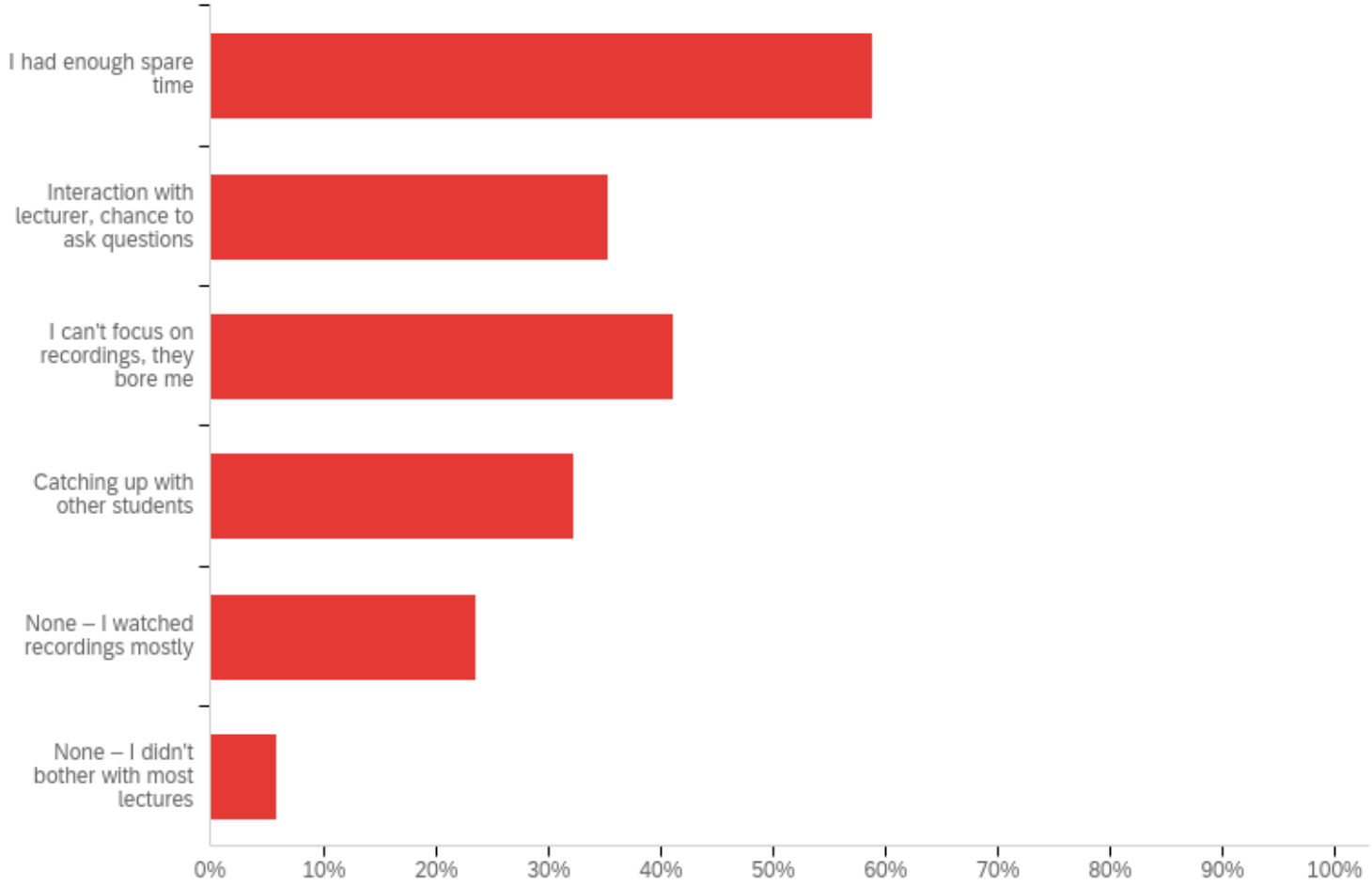
Q17: Missing material

Which material, not presently in the course, would you have liked to be covered? When commenting here, please also comment below, as we cannot add stuff without removing others.
Kernel bypass networking? Bring distributed systems back🥹🙏
More on OS implementations, including both micro and monolithic kernels.
Learning more about how to write device drivers as this seems to be a very practical skill to learn.
Maybe more on how Windows internels is similar to and differs to Linux internals.
Device driver design and implementation. A discussion of some windows internals or design approaches taken in comparison to Linux (in the Unix/Linux internals).
seL4 implementation
some topics on distributed systems and consistency and more networking would be nice, user level sync primitives seemed a bit repetitive and light on information

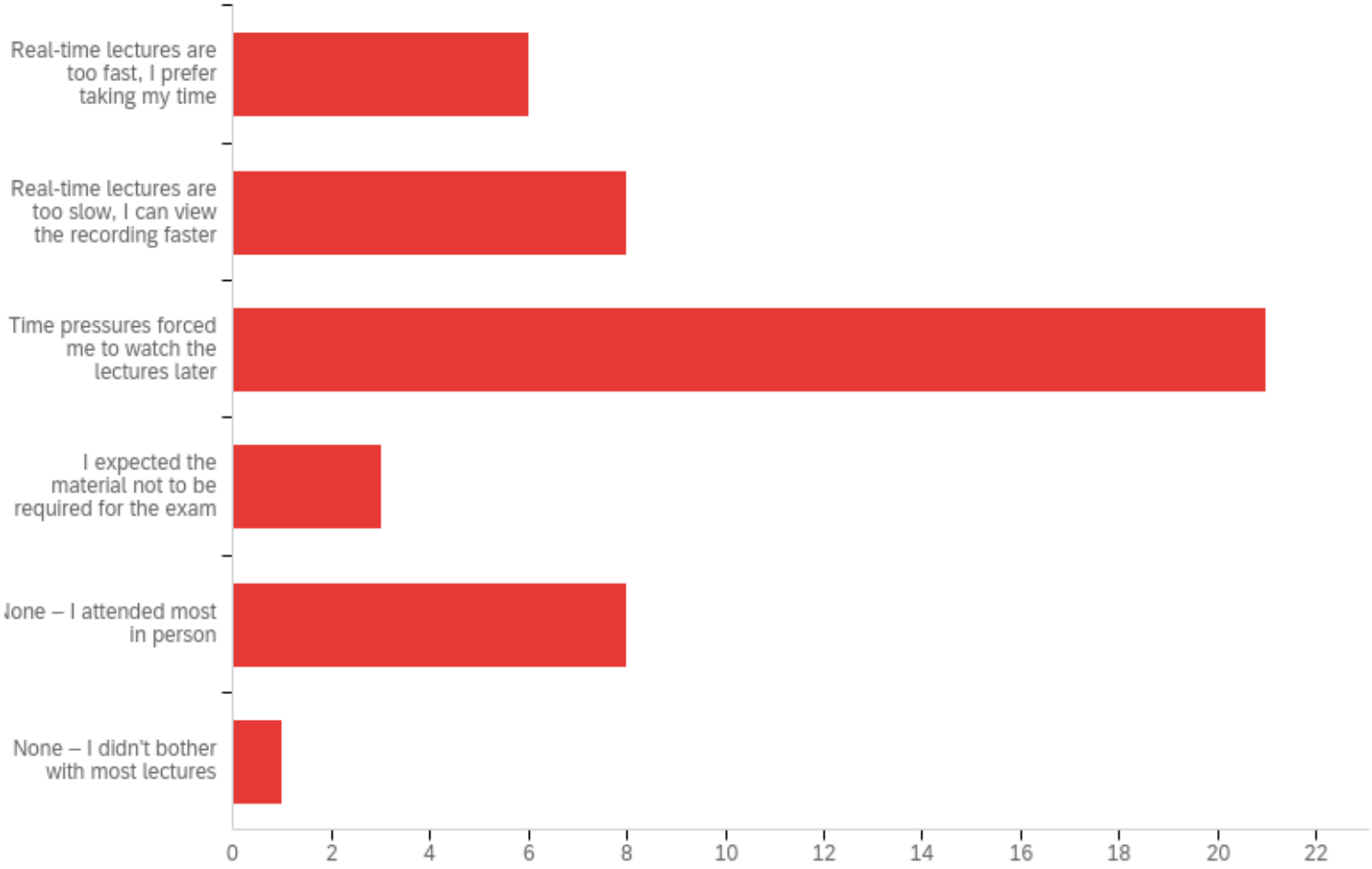
Q18: Material to scale back/exclude

Which of the current material would you like to see scaled back or excluded?
Virtualisation zzz
some sel4 stuff? (sorry😭)
User-level sync primitives
Some of the formal verification, especially the history of OS verification felt less useful to me, especially since verification appears to be covered in other courses, but it certainly made sense to be included in the course.
I am not particularly interested in formal verification, but saying that I can appreciate why it's being included in this course as it's a fundamental part of seL4.
Some of the discussion of Unix/Linux internals since some of the Unix internals is already quite known by most students (especially from the COMP3231 course), also some of the Linux internals in my case were actually self-studied as part of the project when designing subsystems. Some of the early material in the user-level sync primitive lecture is already known from COMP3231 material.
User level sync primitives
SMP

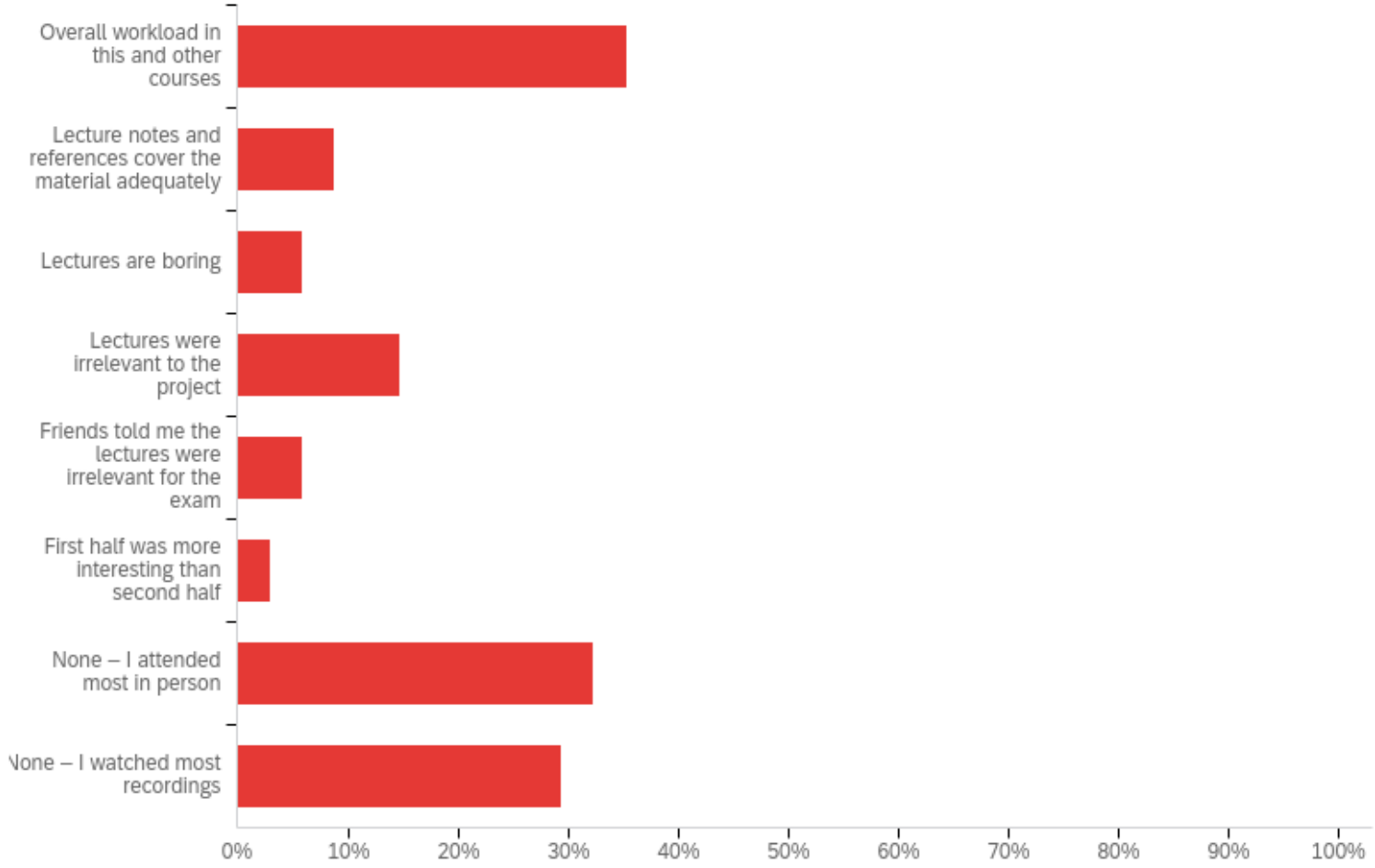
Q19: Why attend lectures in person?



Q20: Factors for watching recordings?



Q21: Reasons not to watching lectures



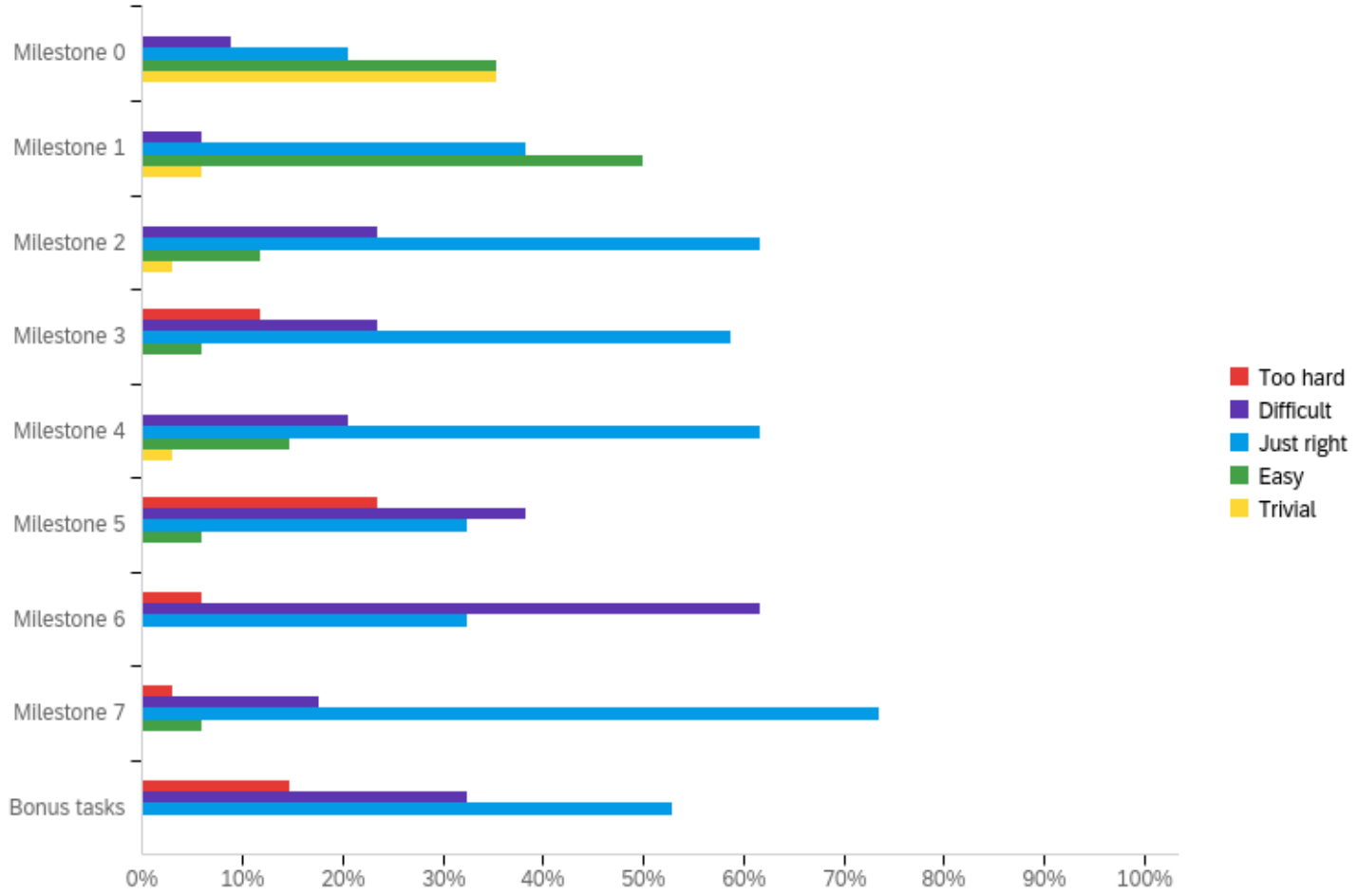
Q24: Other comments on lectures? (1/2)

Any other comments on the lectures, especially suggestions for improving them?
I have some health and personal factors which make can make it difficult to attend and learn from in person lectures. The recordings are invaluable for someone like me to experience the course. The technical issues with some of the lectures (mission lecture slides, occasional poor quality audio) were a little bit of a problem for me, but thankfully I could refer to previous years recordings for particularly problematic sections. Hopefully some of these (very understandable) issues are ironed out for future iterations of the course.
Recording quality wasn't great, some technical issues. I watched youtube instead
At some points they were very technically dense and could do with more "explain this in plain English" moments (illustrations/examples/analogies). I appreciate this takes time away from covering a wide breadth of content.
Gernot's lectures on caching and VMs are the most important imo, but they were quite crammed. Gernot's teaching is great - there is just too much content right now and not enough examples I think. (e.g. stuff like exact path followed by a trap with hypervisors I think a few people were confused by)
I don’t know how viable this would be but a live demo in week 1 during the seL4 API and usage lecture would’ve been great. Something simple like creating an echo server using badged endpoints (or any other simple demo, maybe with collaboration from students) would make the early concepts more digestible than the code snippets that are currently on the slides. I think this would help because seeing how something is done from start to finish always teaches me more than just having the final result explained to me.

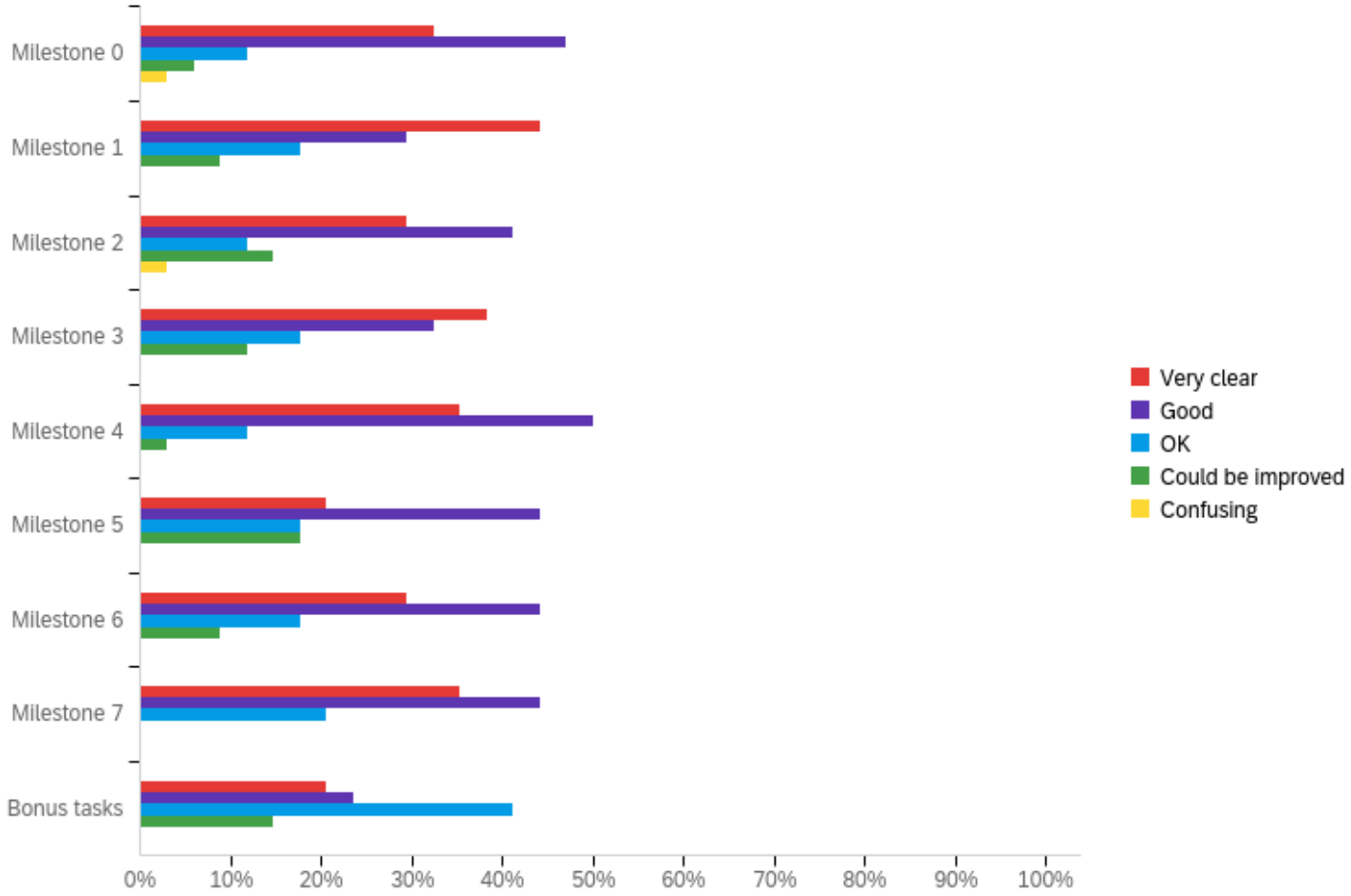
Q24: Other comments on lectures? (2/2)

Any other comments on the lectures, especially suggestions for improving them?
It's hard to make a time to suit everyone but personally a lot of lectures I didn't attend/watch were due to being late. The recordings were nice for when I needed the extra explanation but usually the notes are enough. I prefer reading over listening.
The sequencing could be reorganised so some stuff (like background on security and the way Linux implements things) could be presented earlier.
I preferred to attend most lectures when I could make it. However, I found when watching the recordings that some had poor quality and I had to default to watching older years' lectures for the worst of these. Another consideration would be to also broadcast these live along with having a recording saved for students to re-watch later. There's something to a live recording that at least for me, make me more attentive and focused.
I've become too accustomed to the flexibility of online lectures, as they allow me to speed up familiar content and slow down for complex topics, a practice I had to become used to during covid
Main reason for me not watching lectures is just incredibly poor attention span. Even if the content is interesting I find watching any lecture to be very painful. Generally will watch at 2x-5x speed if at all. So probably not worth taking my views on this into account (:
Lectures are generally paced very good, host and guest lecturers are generally engaging and deliver their contents well. But Anna’s RTOS lecture was kinda sketchy at best
Time crunch really hurts chances to attend lectures, which sucks particularly if you cant make it for a guest lecturer

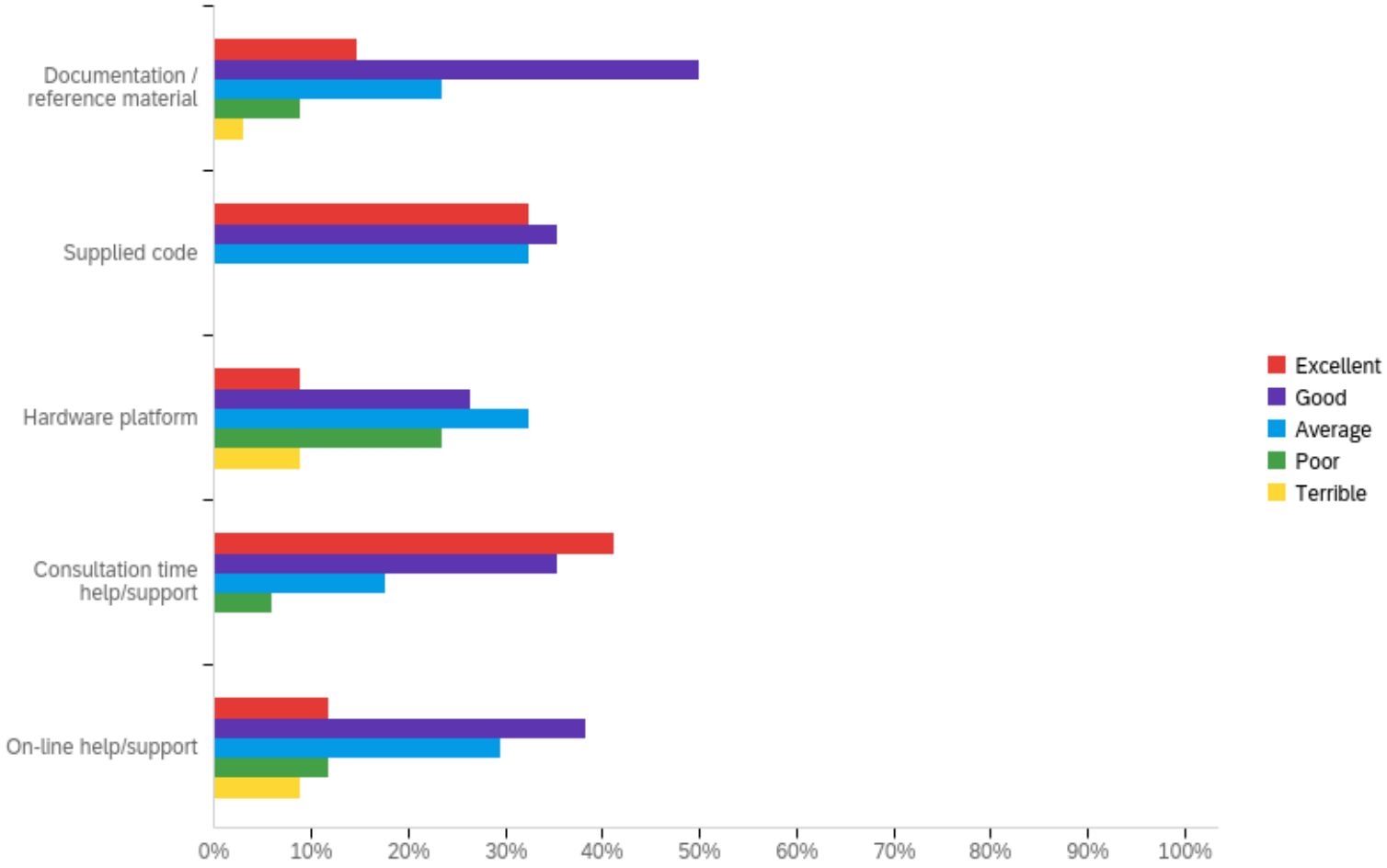
Q25: Difficulty of various project parts?



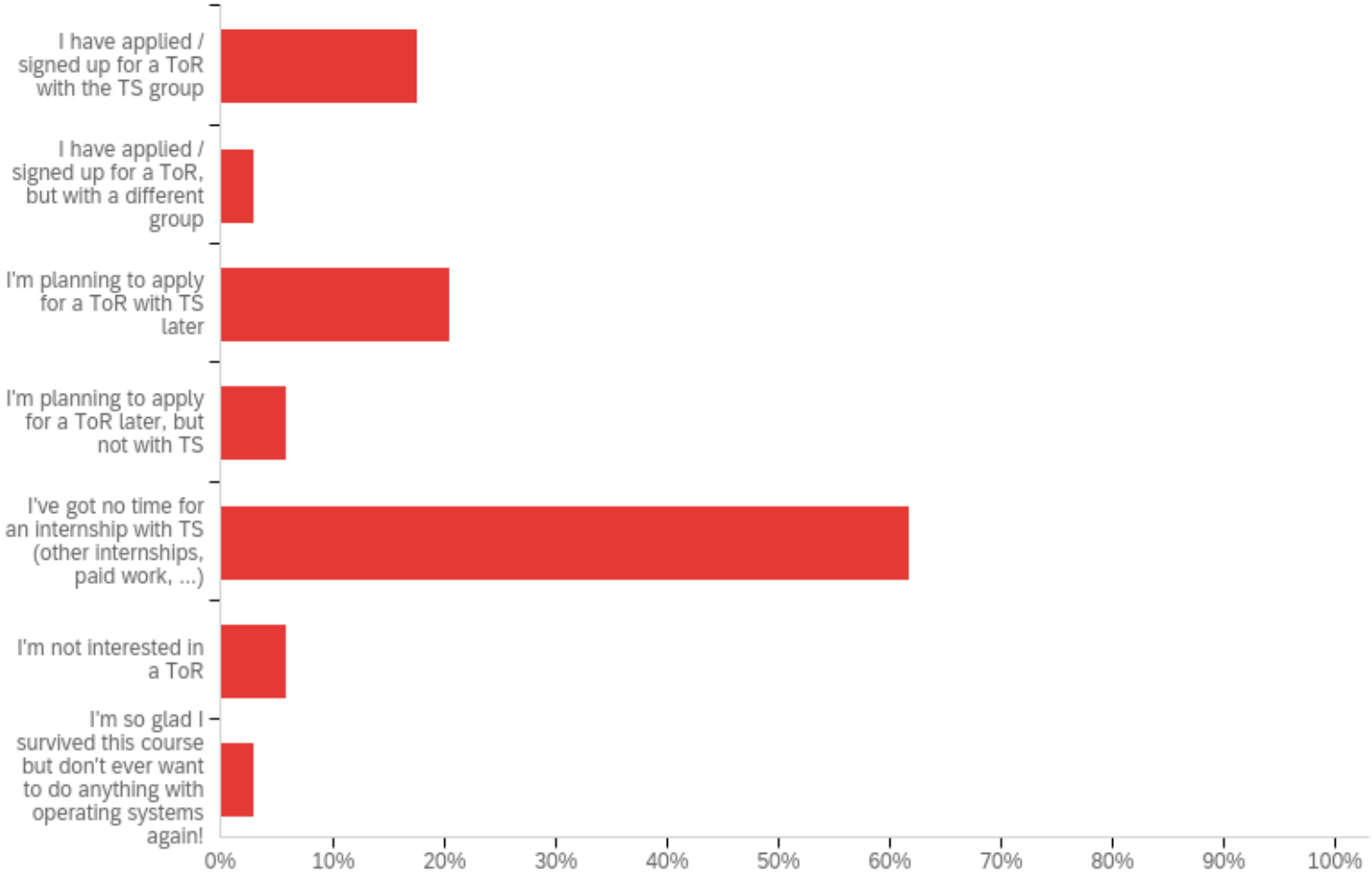
Q26? How well was the project specified?



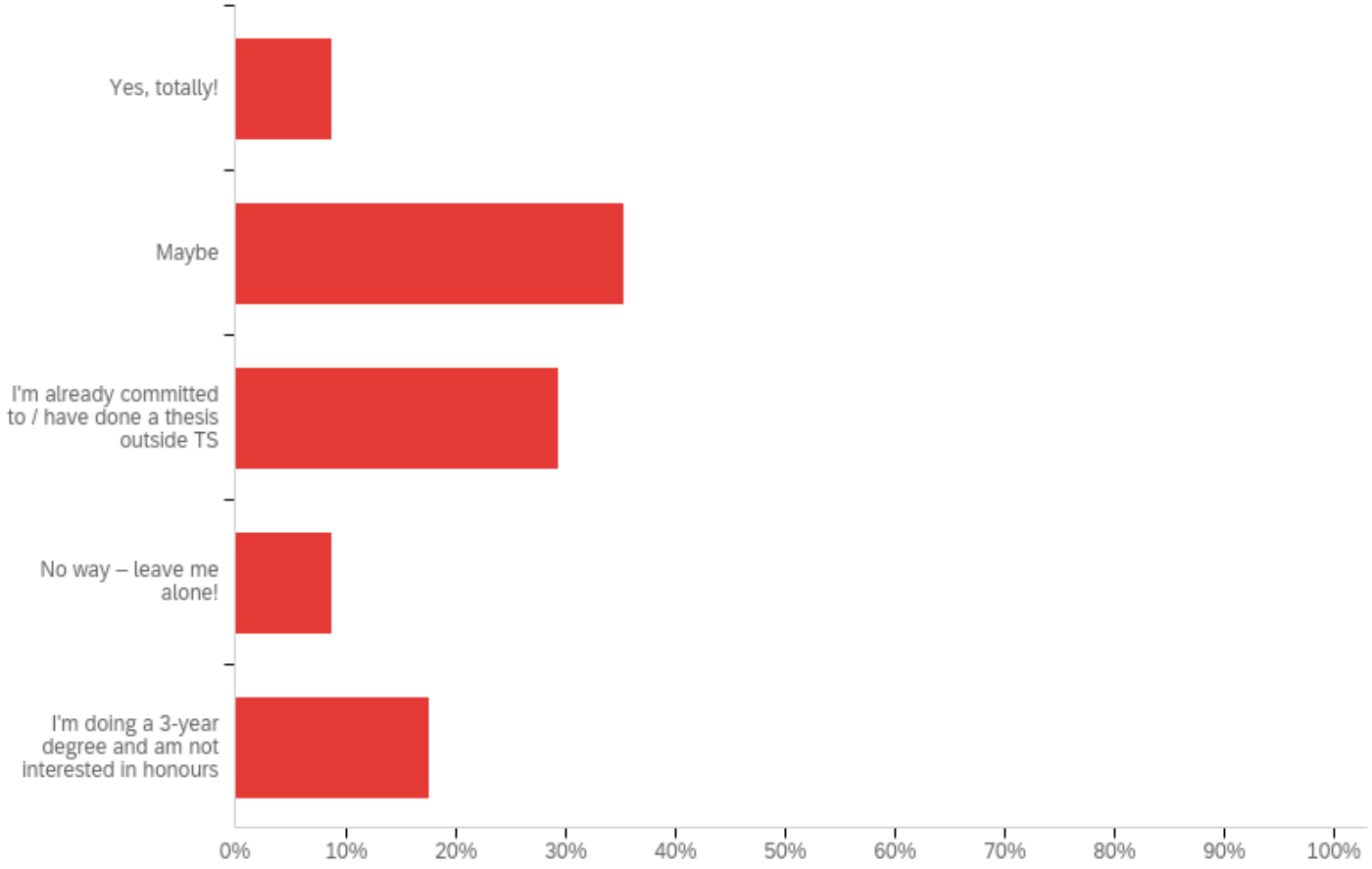
Q27: What was the quality of...



Q28: Interest in Taste of Research with TS



Q29: Interest in Thesis with TS



Q30: Comments on TS ToR or thesis?

Any comments on TS internships or theses?
Currently doing TS ToR
This is my second masters, so not sure ToR is for me. But the course has piqued my interest in systems, microkernels and security.
If there's a possibility of coming back after being in industry a few years I'd be interested
Doing elec eng + CS, so despite really wanting to do a thesis or intern with TS, not sure how that fits in with needing to do an elec thesis + industrial training and (at this point) wanting to pursue a career in elec.
Whilst TS work is interesting to me, I'm not too keen on research.
I'm doign a masters degree, this is the type of course where I wish I had done it earlier, as I think I'd have seriously given consideration to doing research to finish off my degree. This is not the courses fault, the prerequisites for doing OS meant the only way I could have done the above is to stretch out my degree or to have done an undergrad so I could tick off some of those prerequisites.
Would like more information on opportunities for postgrads / graduates in general
ToR pay is absolutely abhorrent, otherwise i wouldve been much more eager

Q31: Any final comments? (1/3)

Any final comments you would like to make?
The online support of the course should probably be improved, the forums were quite inactive whilst the unofficial discord server was frequently used by students and even some tutors and was a great resource. I feel like either the forum needs to be interacted with more or better yet, an official discord server created so all students know about it. As a student who was not aware of the discord server I feel like would have been disadvantaged.
Excellent experience :)
Thanks to Gernot, Kevin and all course staff involved in running AOS!
Thanks for the course and accommodating me given I took it remotely in Melbourne.
Thank you for making the course possible - it's been one of my favourite at UNSW!
Thanks for the course :)
Thank you for putting so much effort into this course. It's been one of the highlights of my time at uni so far.
Big thanks to the team for running a very tough, but very smooth course! Also please make the odroids more consistent if you are going to continue using the remote setup, it got quite stressful and annoying at times. Hope to maybe be around TS stuff more in the future :)
Thanks for running the course :> it was fun

Q31: Any final comments? (2/3)

Any final comments you would like to make?
It was a good course that featured way, way too much content imo. I think a part a/b would be significantly better and allow much more depth into individual components.
Great course to take! Thanks to all the lecturers and tutors! :)
Really enjoyed the course and felt like it was challenging but rewarding. I would recommend making it a bit clearer upfront that part of the challenge is working out things independently and also as a cohort (would suggest making sure students know about the discord as that's a primary place for sharing resources and discussion).
This course has given me a greater depth of understanding of a whole system and how everything works together. I feel that I am a better developer because of it regardless of where I end up professionally. I loved the course - it has been the highlight of my university degree :)
While I did this course, I did not enjoy the pain and suffering. However, now that I'm done, I definitely enjoy the fact that I've learnt so much and survived!
enjoyed the course
Thanks for the wonderful course! Virtual mem and deman paging were huge difficulty spikes The hardware platform was super unreliable

Q31: Any final comments? (3/3)

Any final comments you would like to make?
I think it would be cool if the analysis of papers we did in the exam was integrated throughout the course. Perhaps requiring a summary and critique of a paper on the current lecture topics each week, or something like this. The active critical engagement necessitated by this kind of task is very effective for learning for me personally. Thus it would be cool if for example, instead of a final exam, there were 4-8 paper critiques throughout the term alongside the project. To accomodate the increase in workload, perhaps the last project could extend past wk10 if this is possible (or just make it a higher workload course (:). These are just some rough ideas off the top of my head. But really any way of integrating critical analysis of papers throughout the course would be incredible in my opinion. (of course there is nothing stopping students from doing this independently, though sadly I am too poorly motivated to do this without external pressure):)

The End