School of Computer Science & Engineering

COMP9242 Advanced Operating Systems (AOS)

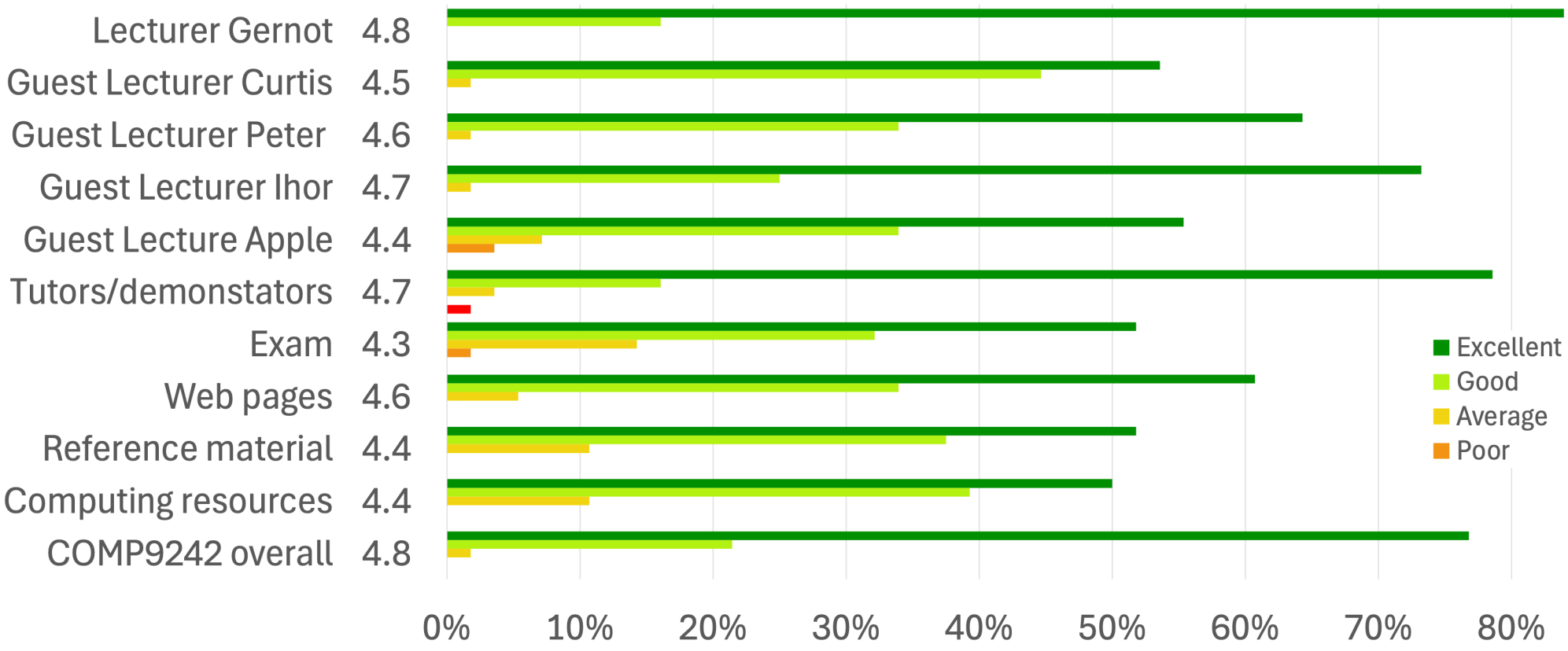
2025 T3

AOS Course Survey Result

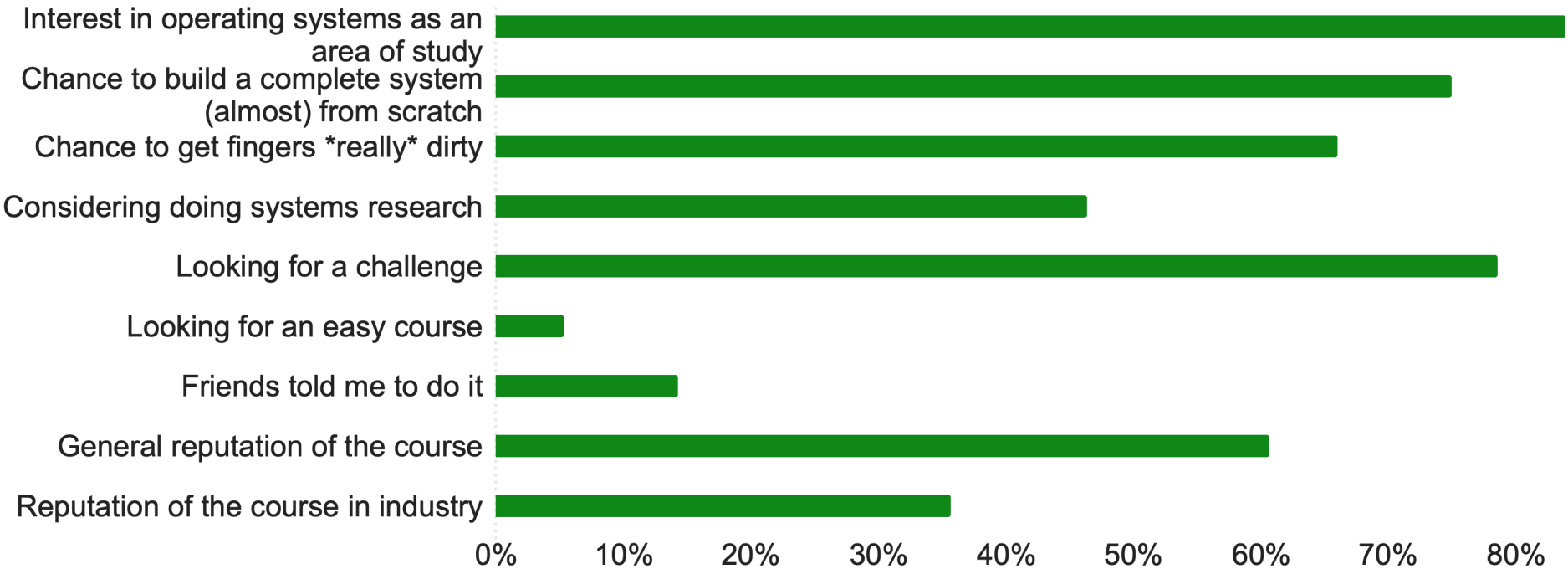
@GernotHeiser

Copyright Notice

Q1: Quick evaluation



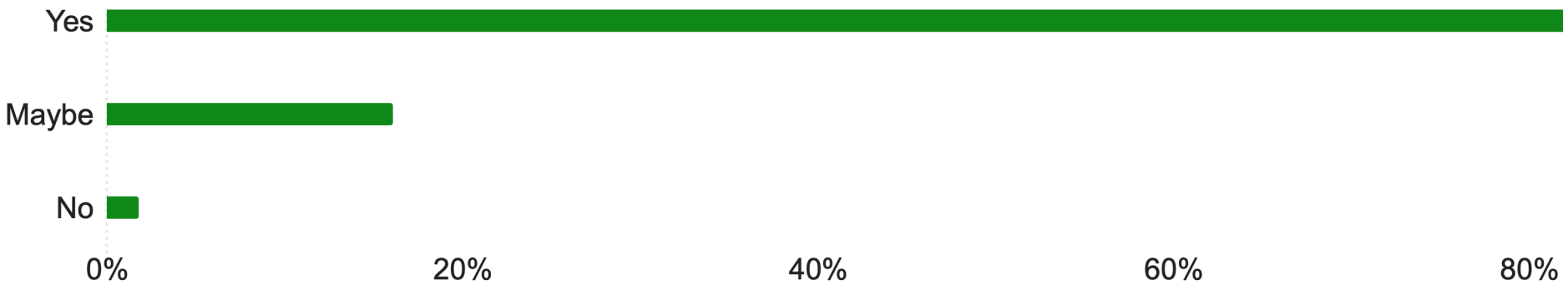
Q2: Your main reasons for taking AOS?



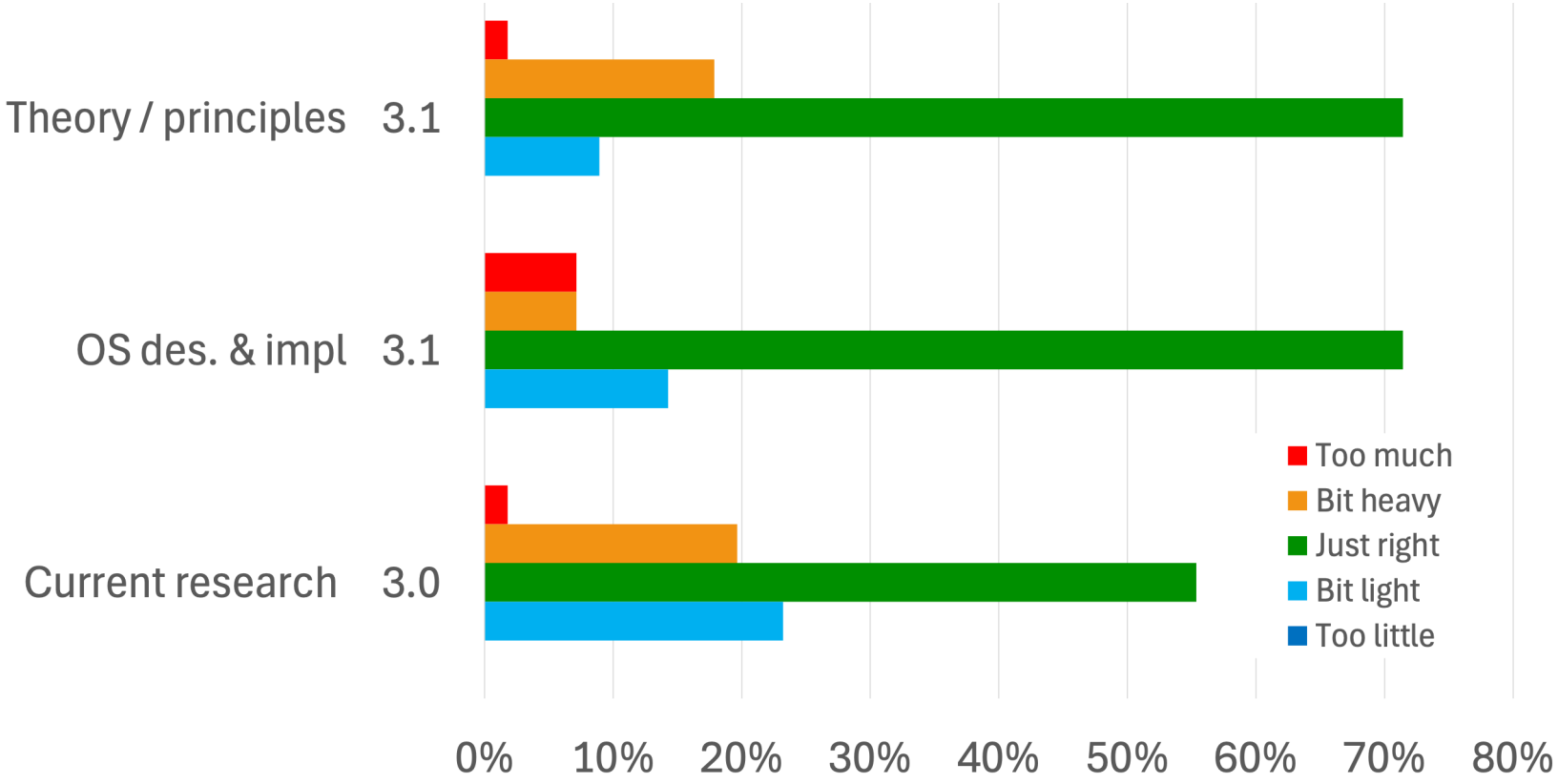
Q3: Your main reasons for taking AOS?

Other factors not mentioned above
Reputation was the hardest course. Wanted to prove I was a part of the best.
I just want to understand thing which is unknown to myself
I get to write Zig
On top of the reasons above, a non-trivial part of the reason for me was that it was an opportunity to study under one of the core creators of seL4, which I hold in high regard.
shirt
I got a HD in regular OS and had nothing else to do, so it seemed the logical next step.

Q4: Would you recommend this course?



Q5: Content Balance



Q6: The best things about AOS? (1/7)

What were the best things about this course?
Teaching and the project
Gernot’s lectures.
Being able to implement a system from scratchish.
I really enjoyed the technical challenge.
The satisfaction from getting a working system from a minimal start.
The project. The capstone of what UNSW CSE has to offer.
Being able to take a project from start to finish was a really cool experience
Milestones were fun
The face-to-face discussions with tutors about our designs, and Gernot's lectures. I particularly liked that Gernot gave opinions on certain historical OS-related designs people have come up with.
How much we got to get our hands dirty. Literally all design decisions.
The course project and the flexibity it offers for us to build an OS
I have learned and understood a lot of OS concepts by implementing them by myself.

Q6: The best things about AOS? (2/7)

What were the best things about this course?
The fact that it is very hands on
We got to make our own design decisions, and later on be able to realize how it affects our system overall. For instance, a bad design decision can result in a lot of refactor and hidden bugs. In other COMP courses, the assignments are either not large enough for you to see how your design decisions affect your progress, or not open-ended enough to explore different approaches. The lecture content was very interesting and practical as well, even though they don't directly apply to the project. I'm very impressed with the guest lecturer Ihor. He taught the content very well and easy to follow. The tutors are very helpful and they are very patient to answer our questions or discuss our design decisions. They all did the course before so they can give really good advice on our approach.
You learn a lot
Code ownership and liberty
The content covered in the lectures, OS theory and learning about the current research in the systems field.
Implementing OS

Q6: The best things about AOS? (3/7)

What were the best things about this course?
Other than the topics covered, which are major areas of interest for me personally, I really like how the course really treats computing as, in a way, a "real" science, with intense rigour, which unfortunately not many other courses do.
The course allowed us a lot of freedom in our operating system design implementation, as well as teaching a lot of really interesting theory we would have not otherwise been able to learn.
Loved assignment, lectures were all really interesting
The project was challenging and rewarding.
Was a really good introduction into systems design and OS in the real world.
Loads of practical experience, loved the design components, felt like an actual project you might do...
Getting to build an operating system from scratch
The Project
Having the freedom to design a system, pretty much from scratch (threads ftw!)
I really enjoyed how practical this course was whilst also giving us a chance to learn deeper into the theory of operating systems.

Q6: The best things about AOS? (4/7)

What were the best things about this course?
I liked when the course looked at historical, current and future designs holistically so we could understand the benefits and flaws of the design. I also liked the weekly milestones as they kept you up to date and made sure you didn't leave things behind too late.
Gernot
- the fact that we were able to take full creative liberty over the design - the extended parts were still relevant and important components for an OS and thus gave you great takeaways in terms of understanding (mmap, shared pages, etc) - the community felt super vibrant and unified (talking to other students and tutors) - touched on present issues and research topics and wasn't limited to just seL4 - genuine sense of accomplishment after implementing each major OS component - how generous the late penalties are. The fact that submitting on monday only took 0.5 marks off but gave you the whole weekend was very forgiving given that you would also put yourself behind with the next milestone - that you could choose which event model you had to use.
You get a chance to explore a range of more low-level areas of computer science.
Building a complex system from scratch and I really most of the theory!

Q6: The best things about AOS? (5/7)

What were the best things about this course?
project
The project, especially with how much free rein we have with design. Also very cool lectures and lecturers
I was genuinely challenged doing the project, but the best thing is that after the course, my problem solving skill was so much improved, my critical skills also got improved, and I am now so much more curious about how things work
The level of effort by the teaching staff is very high.
interesting
The freedom to design and implement based on your own ideas
consult sessions
it was very clear about milestone expectations and show stoppers
The project was a really good challenge that pushed me to learn both technical knowledge and develop mentally.

Q6: The best things about AOS? (6/7)

What were the best things about this course?
The project was a highlight of the course, with the exam a close second. Very different from usual CS courses and very challenging but fun.
Freedom for design/implementation, challenging assignment.
Exam was much more interesting compared to other courses. Tutors were all very knowledgeable and helpful.
The wealth of detailed knowledge and the opportunity to practice systems development and get proper feedback.
Almost making an entire OS from scratch. New OS architecture design in comparison to COMP3231/3891.
The lectures were great.
The implementation component of the operating system where we build our own (not-very-good) OS is very enjoyable
Working on the project.
lectures and exam

Q6: The best things about AOS? (7/7)

What were the best things about this course?
The project. I like how the course is taught me on how to build an OS on top of a kernel. Tutors are cool and very helpful too.
lots of smart/good people to talk to about OS
Super great lecturer and lab demos. A good challenge.
Project

Q7: The worst things about AOS? (1/7)

What were the worst things about this course?
bit too much focused on microkernels
hard
There's quite a bit of time pressure, but it's generally warranted. Sometimes the project with its deadlines distracts from lecture content (despite being highly engaging). Partner allocation for groups is highly variable and not necessarily fair
Tutors being unprofessional in consultations. I'd prefer not to be poked at, insulted, and made uncomfortable while getting my code marked.
There is a bit of a gap between the lectures after week 3, and the project. Alot of the stuff we learnt in later weeks we didn't get to implement
Debugging :(
I didn't enjoy the exam at all; it seemed strange writing a paper critique *for the first time* when most of the course work was coding. I wish I was more prepared; and that the course had more resources to account for this.
Tough time frame to fit in. Missed 1 week representing UNSW for basketball and was playing catch up the entire term.

Q7: The worst things about AOS? (2/7)

What were the worst things about this course?
Can’t say there was anything very bad with the course. Workload is definitely harder but also that is an expectation coming in to the course. I think maybe having a bit more on academic paper analysis to help better prepare for the exam would benefit the course.
Doing the final exam
Getting show stopped, although it allowed me to improve my design I suppose
It was easy to go astray in the project, doing implementation quite inefficiently or even incorrectly and it was hard for the tutors to pick this up, they can't go through our entire codebase. This cost me a lot of time to fix and with some more direction I would still have learnt the same amount with a lot less pain.
project
Working with my partner
Debugging and high workload
Beside the first two lectures, other lectures seem irrelevant to the project
Sometimes the spec was a tiny bit too vague. I personally had to ask tutors about certain expectation details that weren't really given in the spec

Q7: The worst things about AOS? (3/7)

What were the worst things about this course?
My partner was incredibly difficult to work with (my fault, should've vetted better beforehand)
Hardware issues, seperation of assignment content and lecture content
The open nature of the course meant that I spent more time than was sustainable at the start which came back to bite me later in the term.
Odroid reliability :(
Terrible tooling--remote debugging is a nightmare
seL4 was compressed into first 2 lectures which was hard to keep up, but looking back contents were enough to finish the project. m0 could be separate from the project and be about seL4
Some of the lectures felt a little disjoint from what we were actually doing, but I suppose its not a big issue since we were learning interesting things
Mainly workload, though I don't really blame it on the course, but rather the university's decision on trimesters. Asides from that, there really isn't much recourse for if your partner drops. I think it's an excellent idea to design the course such that students who will otherwise fail would know to drop within the first few weeks, but for those who stay, I don't think there's enough support on either finding a new partner or otherwise adjusting the workload

Q7: The worst things about AOS? (4/7)

What were the worst things about this course?
The lectures were somewhat orthogonal to the project
exam i didnt like since it was pretty different to the rest of the course, would've appreciated if there was some other stuff similar throughout the term
The should be a warning on the handbook that we must find a partner before enroll in this course
Intermittent unavailability of the ODROID cluster
Mostly HW/network issues, like the odroids constantly being down (which was a result of not having enough odroids for each student/pair I have been led to understand)
It's too hard. It was not always clear what the showstoppers are
I found that implementing paging for event based approaches required a inelegant solution which made me sad.
Lab demoing required a lot of planning and waiting to get marked. A couple of times, my partner and I would wait in a demo session for 2 hours just to not even get seen. This got better towards the end of the course as less and less people demoed as early as we did, but it was quite stressful early on.
our team lacked some seL4 knowledge in the beginning.

Q7: The worst things about AOS? (5/7)

What were the worst things about this course?
The workload volume was a lot
Would've appreciated a bit more guidance on how to begin/where to look for information in milestone specs, the practical project and theoretical lectures didn't feel too linked together which I didn't mind, but maybe some hints on implementation in the relevant slides would be neat
The apple guest lec was average
nothing
Issues regarding the odroid machines were painful at times, and a lack of forum answers for seemingly reasonable questions
Debugging said OS issues, due to having broken GDB a few weeks in, and sometimes there was lecture content such as the caching content which went a bit too fast and attempting to catch up, the lecture slides don't really have much explanation which i think would be useful for revision rather than just diagrams
Unclear criteria on certain milestones and showstopping criteria. To specifically point out something, m6 should've mentioned that the SOS has no concept of a parent/child process, and that any process can wait on any other process. I only found this out through the discord server when tutors were answering student's questions

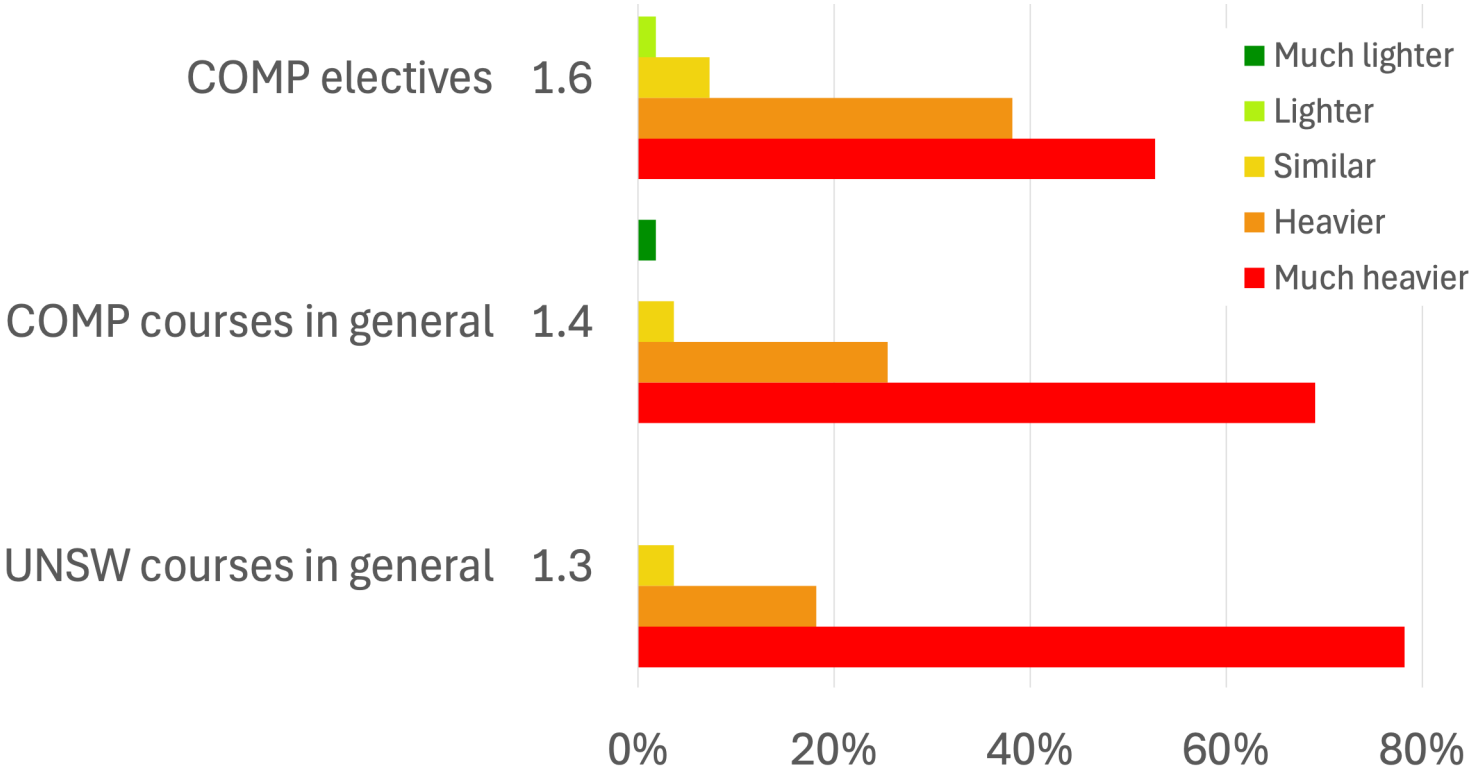
Q7: The worst things about AOS? (6/7)

What were the worst things about this course?
Relevance of lectures to our project workload (or project workload to lectures)
Much of the final grade (30%) relying on W10 autotests and documentation. Felt like we had very little time to put together a good report without submitting late. In future terms it would be good to make the project milestone due at least on the monday of w11 instead of friday of w10
Sometimes the precise wording of specific milestones was a bit vague
No multicore? Not much on the bootloader side, so it didn't entirely feel like an OS from scratch (excluding seL4 itself for obvious reasons). A little harder to connect theory and work done but maybe there's no getting around that
The exam tested skills (academic writing and critique) that weren't enforced throughout the term strongly enough
More demo times would've been good
The content from the lectures was a bit too heavy, required a lot of time outside of class to do more research to comprehend it fully - this is totally fair, but with the workload of the project, which already occupied all of my time, I think it's a bit unreasonable to do that
Compatibility of build system with Windows and mac were not as clean

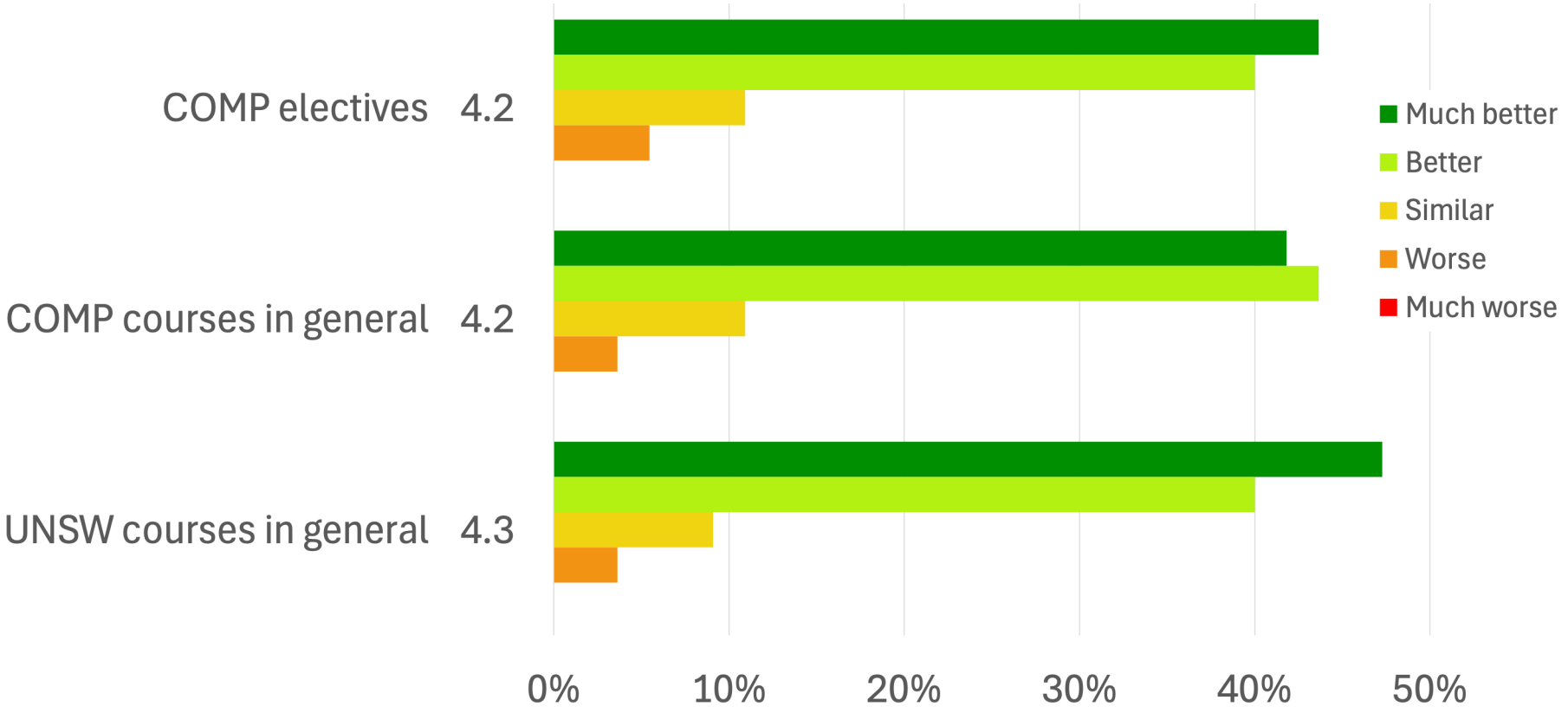
Q7: The worst things about AOS? (7/7)

What were the worst things about this course?
You learn a lot
Not enough content on the seL4 kernel itself
The forum was a bit dead at some time, even though there were quite a few questions hanging around. I don't really like the fact that we had a private chat channel and students tend to use it as their go-to places for questions. The tutors were in the channel as well, which I think it encourages students to use that channel more than the forum. And suddenly there are people (who are not in the private channel) waiting on the forum for not so frequent responses from the tutors / course admins
- i suppose with complete free will it was hard to know if I was on the right track whenever implementing a feature (which i suppose could be a feature of its own)
Maybe the first few milestones being due in consecutive weeks. But this is maybe a skill issue on my part, I was doing this course with ext. algos and ethics

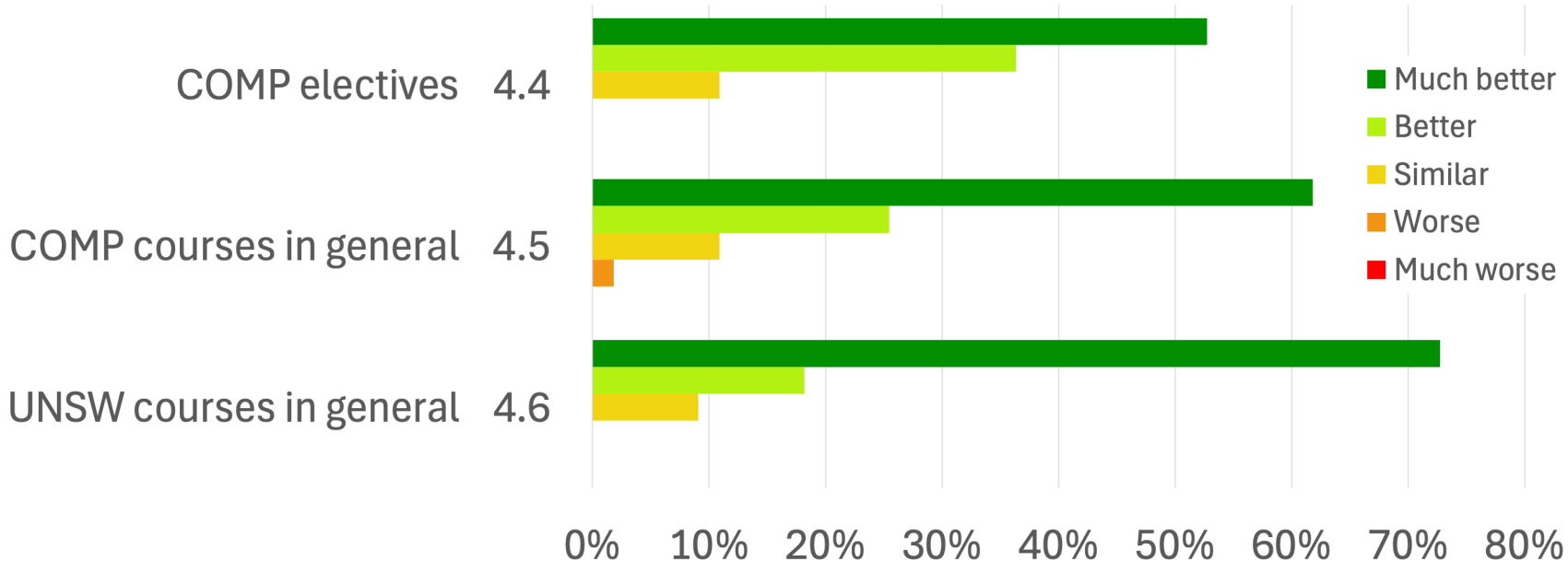
Q8: How does the workload compare?



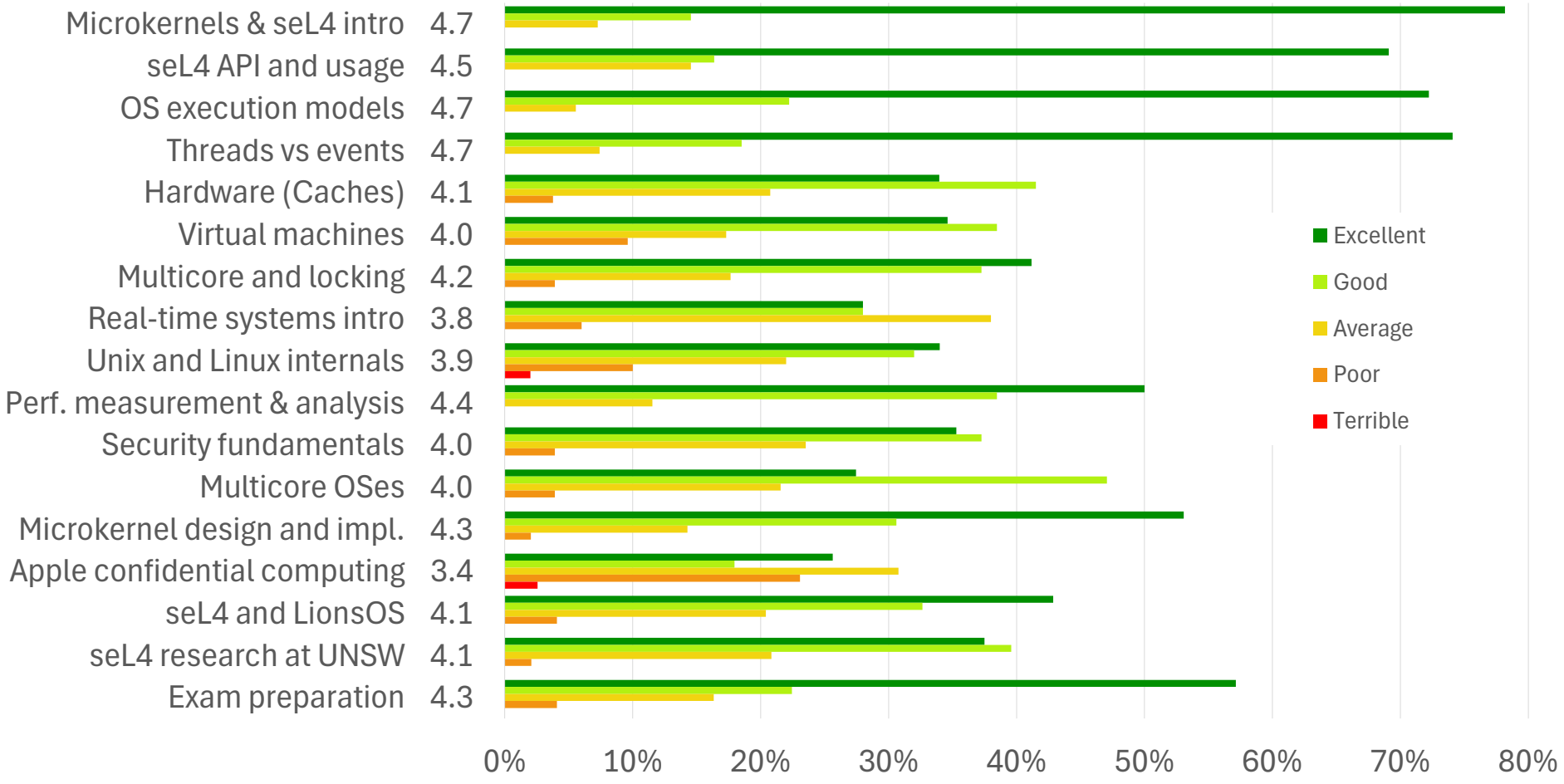
Q9: How does your experience compare?



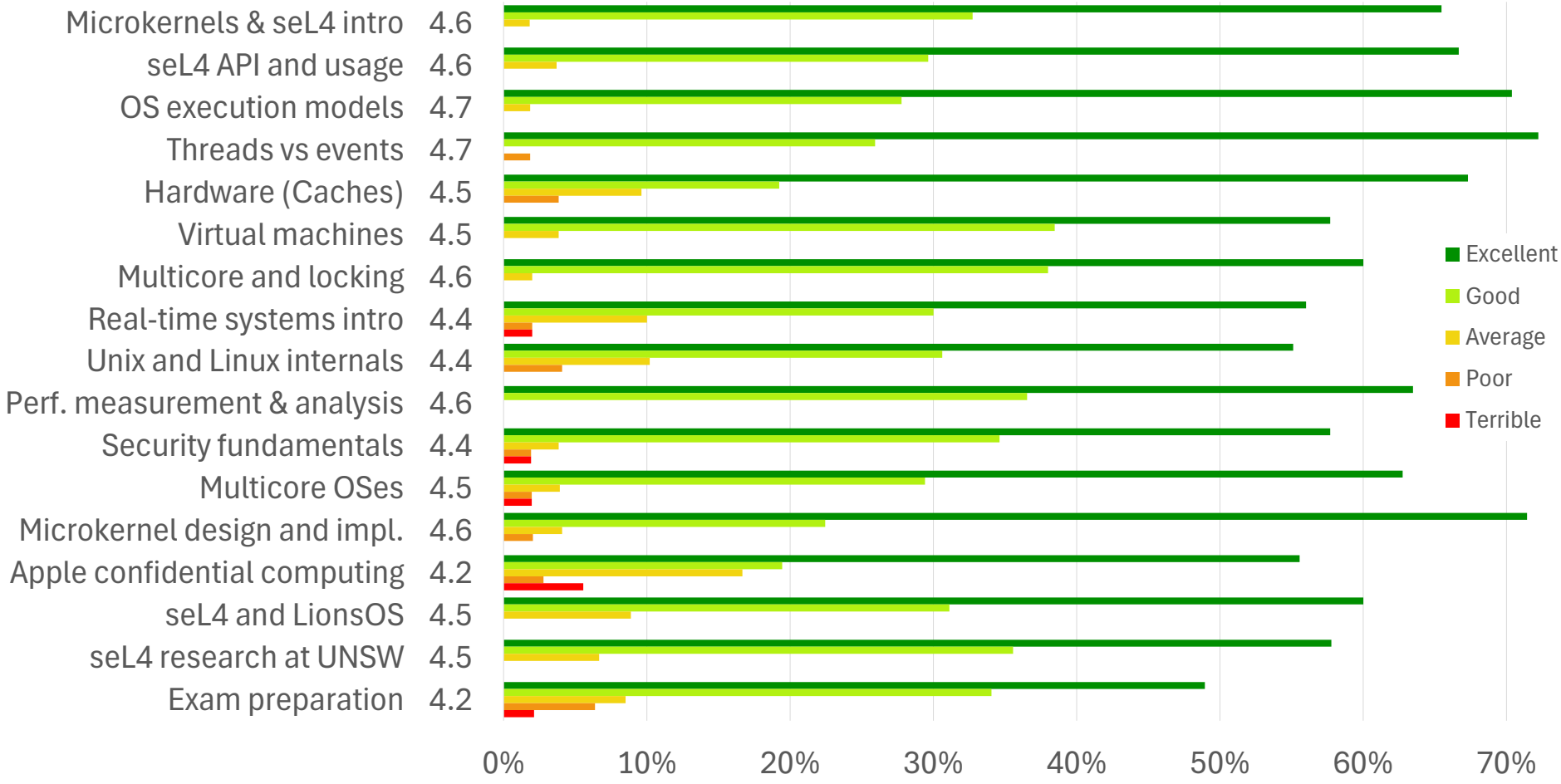
Q10: How does quality/value compare?



Q11: Topics relevance/appropriateness



Q12: Quality of the lectures



Q13: Most useful material (1/6)

Which material do you think will be most useful to you in the future?
performance measurement and analysis, multicore and locking
design
Unix/Linux internals, multicore, performance measurement and analysis(!), security, threads vs events / execution models.
Performance measurement and analysis, followed by security, unix/linux and microkernel design
Everything
not sure
Understanding how computers work: caching, multicore, RTS etc
High level design that I picked up while attempting the major project
Lectures
The project itself. Picking our execution model
Performance measurement and analysis
Personally, benchmarking was actually generally pretty useful for me in other areas or study. Security very nice. I found multicore and caches very interesting

Q13: Most useful material (2/6)

Which material do you think will be most useful to you in the future?
Multicore and locking, reasoning about concurrency is useful
the tshirt
Caching, real time systems, performance analysis
Performance measurement
Performance analysis
All topics about system design as this skill can be used in a wide range of fields
Apple confidential computing
Performance measurement and analysis, virtual machines
I think all content will be useful
Caches and benchmarking
The performance analysis stuff--I never thought about code execution with that level of depth
Microkernel design and performance measurement analysis
All the operating system tradeoffs that are uncovered during the project

Q13: Most useful material (3/6)

Which material do you think will be most useful to you in the future?
A lot, but the general OS theory will always be useful e.g. multicore/locking/VMs/caches
I think it would be performance analysis and measurement, as well as security fundamentals, as it's applicable to virtually anything one would do in computing. Though I'd add that this doesn't necessarily mean that those were my *favourite*; that would be either hardware (caches), virtual machines, or multicore and locking
unix and linux
Threads vs Events
Hardware (caches), virtual machines, multicore and locking, real-time systems, Unix and Linux internals, performance measurement and analysis, security fundamentals, multicore OSes
Kind of depends what I end up doing, but I would say a tie between the cache coherency and multicore stuff, VMs and security/capabilities lectures. All of these were topics I didn't know a lot about, but AOS taught me a lot and made me more interested in them (also if you want accurate lecture feedback, might be good to release a feedback survey each week on the lectures as I can hardly remember the little details now - probably will have less engagement, but at least the answers will be fresh in people's minds

Q13: Most useful material (4/6)

Which material do you think will be most useful to you in the future?
The less seL4 specific content
I think the part about performance measurement and analysis will be very useful
Thread- vs event-based systems
locking, threads vs events
caches +drivers, performance measurement and analysis
Threads and events execution models since it's useful to have knowledge of in many fields. For me specifically I'm into memory stuff so caches and multicore locking will probably be useful in whatever I choose to pursue in the future
Microkernel design, multicore design, and OS implementation experience
hardware (caches), seL4 and LionOS, Multi core and locking
Execution models, hardware, virtual machines, multicore, performance, security, and linux internals
Performance measurement and analysis since it covers many benchmarking crimes which will be useful for anything regarding benchmarking in the future
threads vs events

Q13: Most useful material (5/6)

Which material do you think will be most useful to you in the future?
Research into new fields and Unix/Linux internals
Not entirely sure. Maybe multicore lectures if I had to pick
Caches, Locking, Virtualisation
Caching, Memory ordering & locking, and benchmarking
performance / benchmarking
security fundamentals
OS execution models, threads vs events, virtual machines, multicore OS
Performance measurement and analysis, caches, virtual machines
Security fundamentals
Most of them
Probably Performance measurement and analysis. The lecture was very interactive (a lot of thought provoking questions) so the content really sticks in my head. A lot of real world examples have been brought up as well. The content is not only applicable for OS reasearch and development but also for any fields that care about performance

Q13: Most useful material (6/6)

Which material do you think will be most useful to you in the future?
All pretty useful
Knowledge about microkernel development and design. As microkernel research furthers and the field expands going into the future, these fundamentals will provide good footing for staying up to date and making judgements on emerging ideas
Most of the lectures except for seL4 API and usage and seL4 research at UNSW

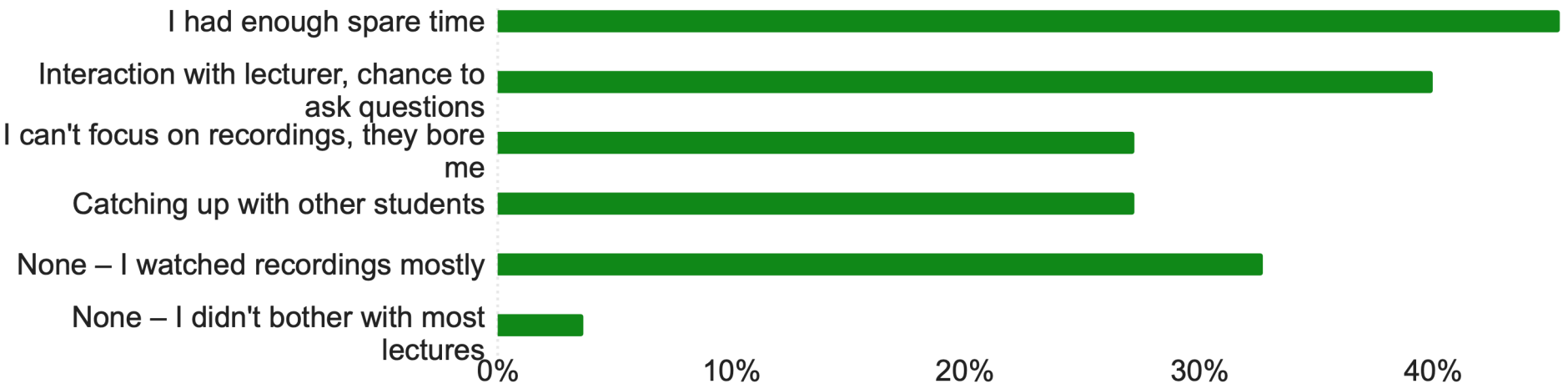
Q14: Missing material

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.
Exam preparation was covered but maybe a workshop or some more advice on tackling heavily technical academic papers with an OS/computing focus
Maybe more about allocators, allocation strategies, allocation with microkernel principles, dynamic allocation and programming for embedded / memory constrained environments. (I'm happy with the content in the course, this is the first thing that came to mind when I read the question and I know that it may be a bit less connected to OS than the existing topics, its just food for thought)
I would have liked to see a brief overview of the use of formal methods in the development of seL4, though I don't think there are any topics that were covered that I would exchange for that either. Perhaps if the university decides to change back to semesters, that would be great.
Asynchronous kernel APIs like epoll, io_uring, kqeueu etc
More detail on the implementation of sel4 vs other micro kernels
Alternative microkernels and their implementation specifics and differences to seL4
Bootloader, Unikernels (one I am aware of is Unikraft)
Not too much comes to mind, maybe some more on performance optimization / performance engineering? (as they were in some of the extended parts)

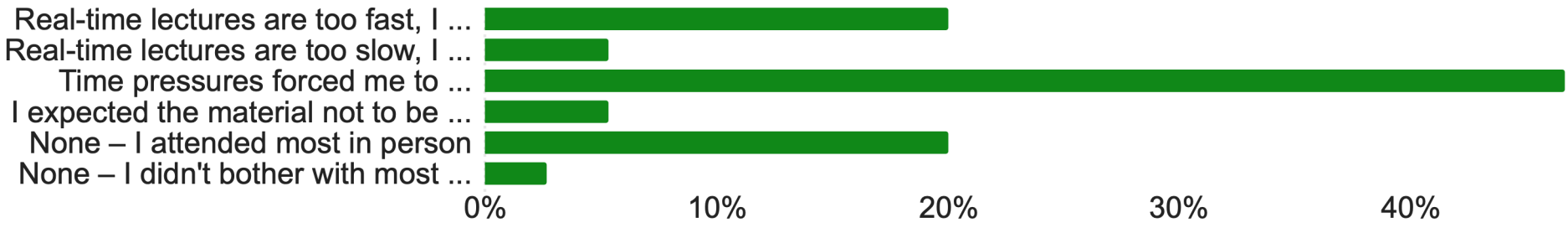
Q15: Material to scale back/exclude

Which of the current material would you like to see scaled back or excluded?
ts shill
I did not see the Apple lecture, but considering it wasn't (allowed to be?) recorded, I would offer that lecture slot up first on principle. Ignoring that, I personally think that the multicore lecture could be a bit shorter. I don't want to talk too much about the lectures I skipped because that would be unfair of me, but the fact that I skipped them at all indicates to me that they might not be as relevant (in regards to exam cramming at least).
research stuff like papers
Maybe hardware caches, not because the content wasn't interesting, but it didn't seem too relevant to the project. Obviously, this only holds if the main goal of the lectures is to be useful for the project, which I don't think is necessarily true
Caching
Linux internals

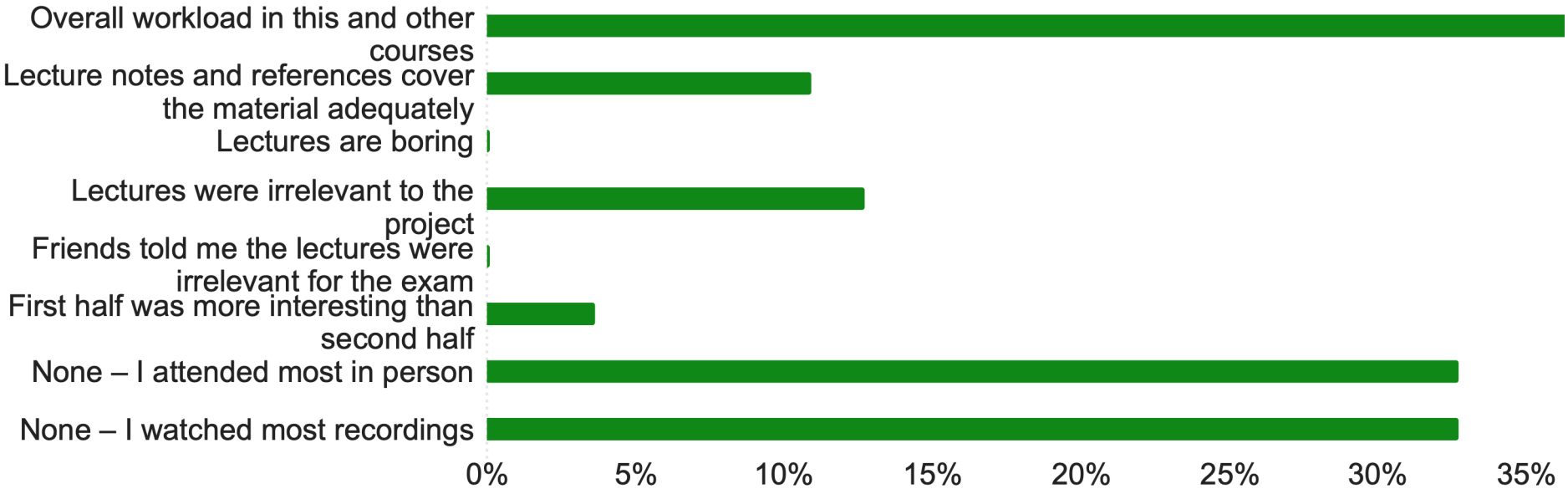
Q16: Why attend lectures in person?



Q17: Factors for watching recordings?



Q18: Reasons not to watching lectures



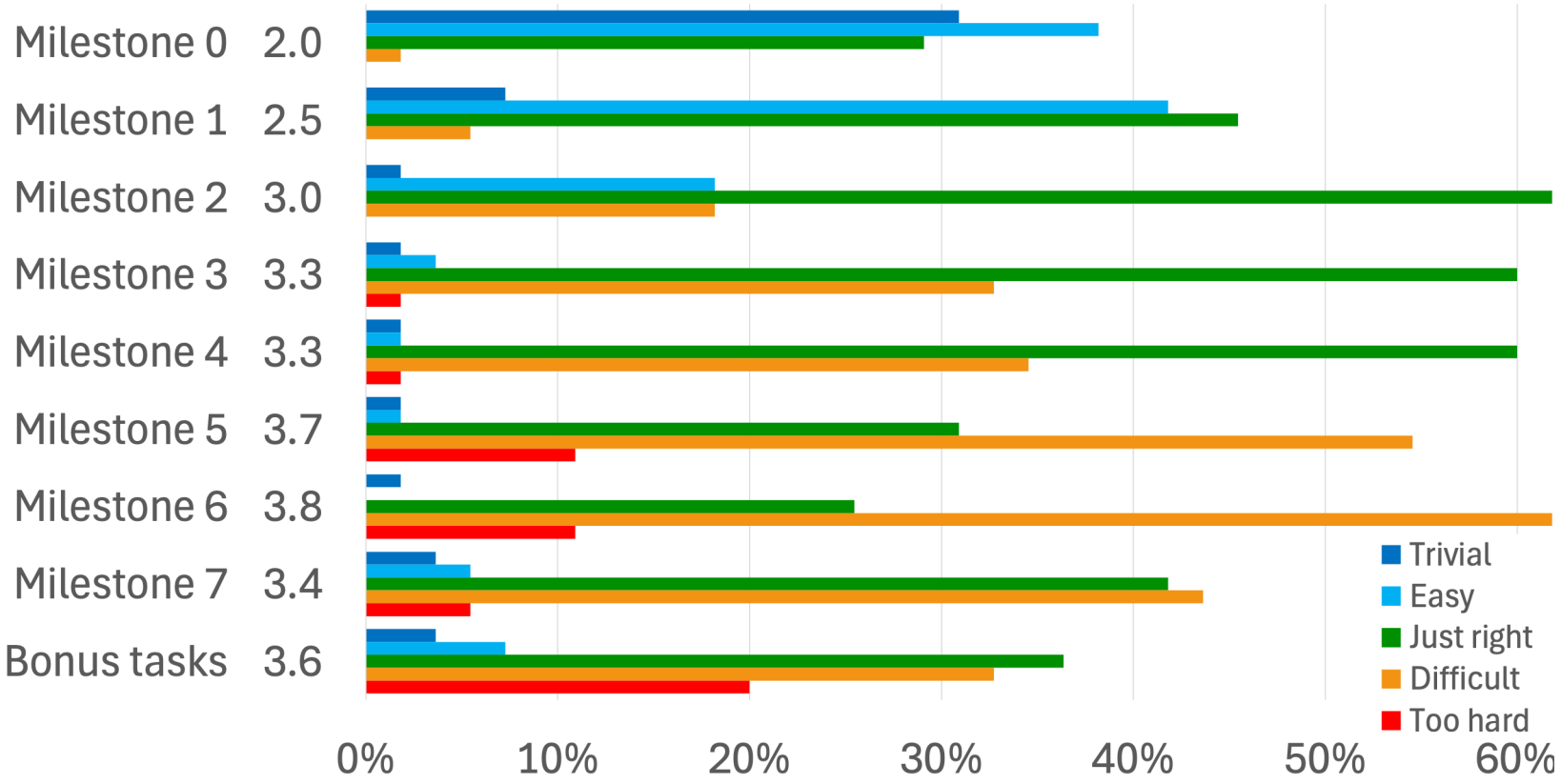
Q19: Other comments on lectures? (1/2)

Any other comments on the lectures, especially suggestions for improving them?
More after lecture drinks with Gernot ;)
Please have air conditioning for every lecture
Just was to busy fumbling through the project to spend any time at lectures. Best way to learn is just to get hands deep
The lectures were good, especially in person. The recording quality tended to be quite poor, with audio or visual issues early on in the term. As the project took up more and more of my time I had to watch lecture recordings rather than attend in person more to finish milestones on time
Major improvement: no unrecorded lectures (I'm aware that Gernot wasn't aware that the Apple guest lecture couldn't be recorded until the day of, I just want to hightlight the issue). Personally, my schedule is inconsistent due to part time work and varying workloads, so I really appreciate having the lecture recordings. For the lecture content themselves, (my memory is a bit fuzzy on the content at the time of writing, I could be way off), they felt very theory heavy. I would like to see more historic examples / implementation details. I feel like the seL4 api lectures could also have more actual code examples in particular. The pacing of lectures could be more consistent too, in general I feel like lectures started off too slow and ended too fast or vice versa

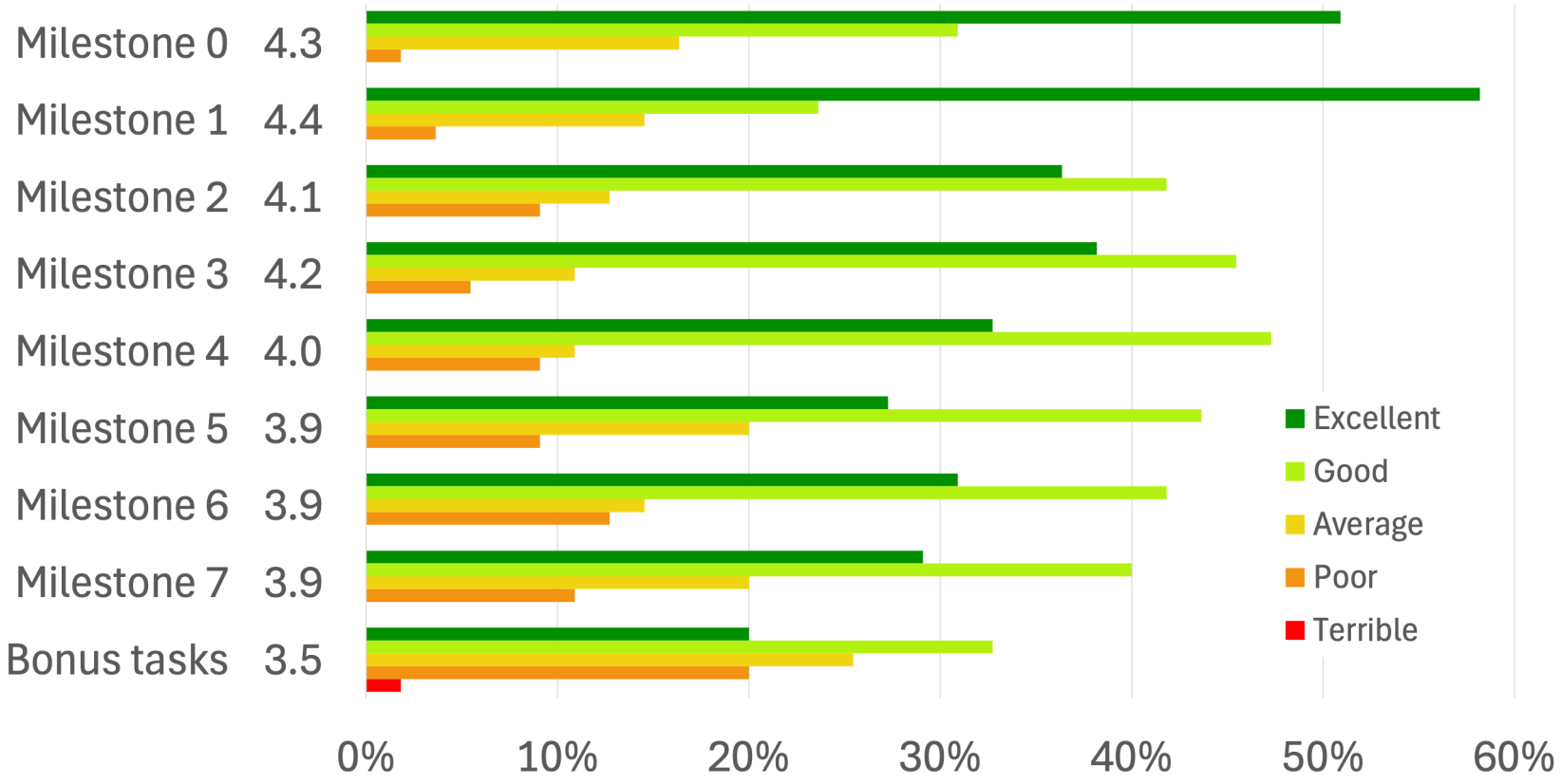
Q19: Other comments on lectures? (2/2)

Any other comments on the lectures, especially suggestions for improving them?
i would've went to the lectures if I wasn't working full time
None of these options were really representative to me - I came to the lectures I could in person because I think it gives a better learning experience and wanted to get the most I could from this course. However, timetable clashes meant I couldn't attend them all (specifically monday's) - not really something you guys can fix though ;(
I really liked the lectures I went too, I did not go to them in the second half of the term due to being too busy. In retrospect, this may have been a false economy
Some in the first half ended early, if I remember correctly. That time could have been used for more QnA, maybe
Sometimes i got stuck in a time crunch trying to submit milestones in time, which made watching the lectures slightly lower priority. Esp lectures such as benchmarking, multicore and rtos didnt feel relevant immediately, so this caused a backlog of lectures to watch later on
Making it a requirement to read the papers and restructuring the lecture to be discussion driven around those, using the lecture slides more as a guide for the emerging concept taught by the papers. This would get students to engage more even with theoretical content

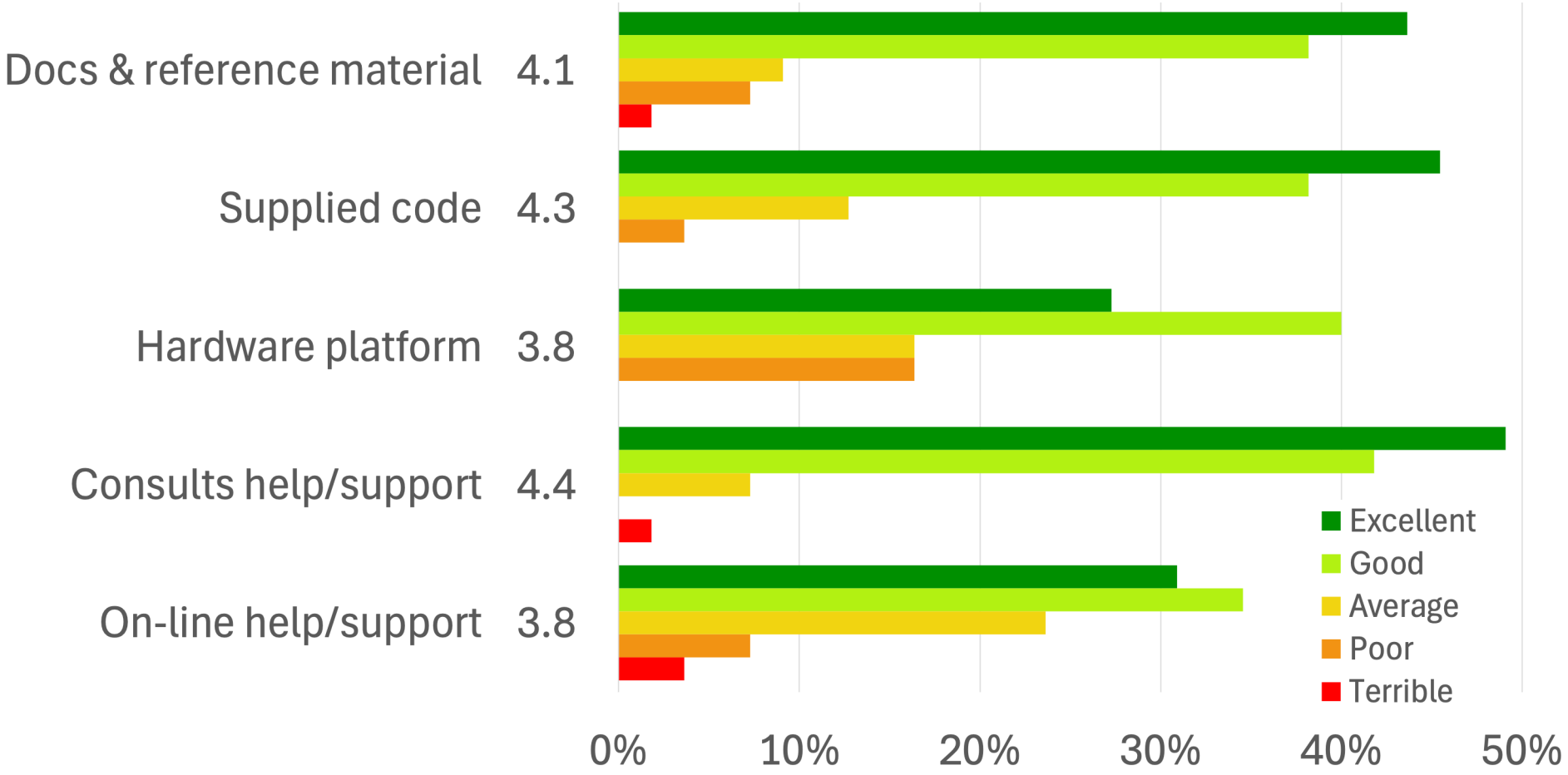
Q20: Difficulty of various project parts?



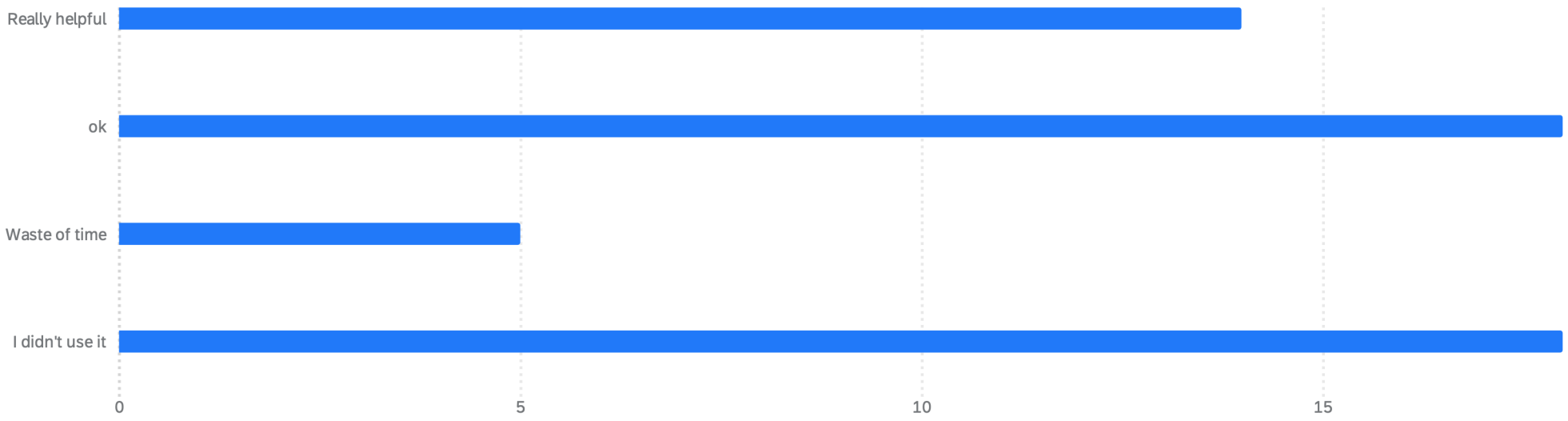
Q21: How well was the project specified?



Q22: What was the quality of...



Q23: GDB Experience



Q24: Comments on GDB? (1/4)

Any other comments on GDB use?
it would be good to explain how the custom gdb code works in sos. helpful if you accidentally break it
We had trouble effectively using GDB after implementing M3/M5. Weird issues appeared which pressured us to use printf debugging instead, as there wasn’t enough time to identify any root cause. Consequently, debugging was a(n even more) tedious process that took a *lot* of time
It was very helpful for finding where an error occured but for most serious problems I had to rely on printf debugging to log information about factors gdb is unable to track such as control flow, variables in functions higher in the call stack and data flow
Wasn't sure if GDB had rich support for Zig. Also, I've observed that debugging with GDB was slower than without.
idk like genuienly std::cout faster
Good for stack unwind but then used mainly LOG
Doesn't work with multithreaded model.
I definitely think GDB helps the course when you know how to use it prior. I think for a first timer, using GDB on SOS can be very overwhelming, however, for example if you had use GDB in other less technical courses, you would know how to use it to benefit you and not waste time

Q24: Comments on GDB? (2/4)

Any other comments on GDB use?
Just a printf guy. Everytime I get told printf wont help it still is helpful and allows me to track code line by line.
Found it difficult with multiple threads
For some nasty memory bugs it wasn't that useful, and that's where I really needed it
I have some experience with GDB, but could barely get it to work for complex cases due to really deep stack traces. Also many bugs were either in m5 or m6 and it was just easier to debug without GDB at some points.
Need to improve the workflow of using it.
Using GDB plugins such as pwndbg helped quite a bit by adding UI elements that constantly shows things such as which line in the source code we are at.
We set up a semi-threaded model pretty much from m2, and it seemed like a massive pain to get gbd to work with multiple threads the way we were going. This was obviously an issue with us, but idk, maybe making the gdb stuff a bit more multithread-happy would be nice?
We mostly used it to track down which LoC lead to a seg fault or other error, and used print debugging a lot as well.

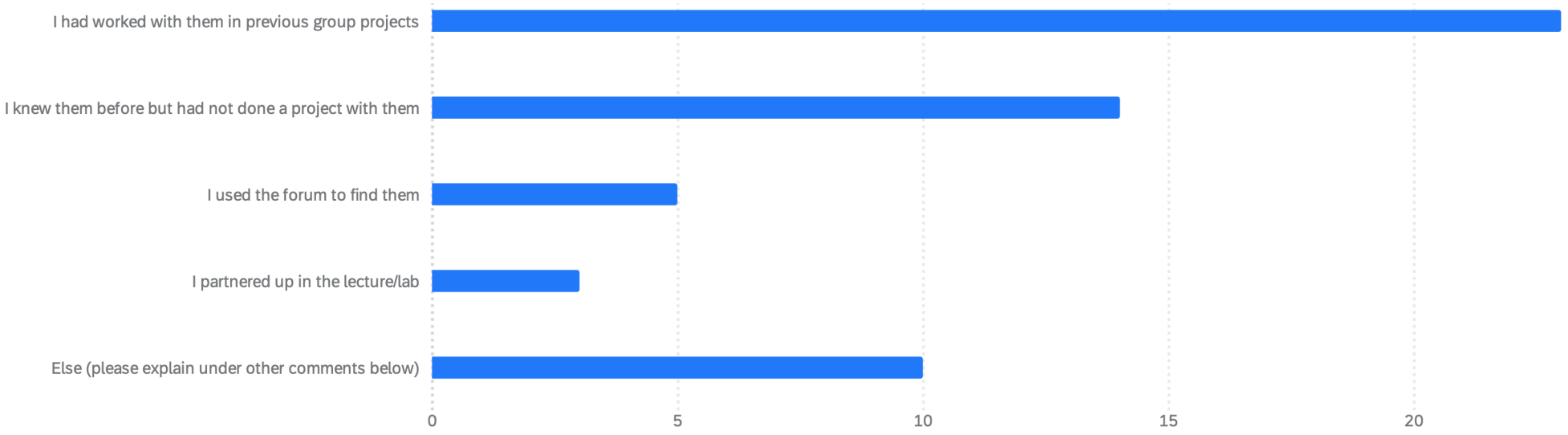
Q24: Comments on GDB? (3/4)

Any other comments on GDB use?
multi core and user side gdb examples could be helpful
Found myself using GDB toward the end of the project to search for bugs because that's when everything gets complex and printf was insufficient. GDB just helped us find where it broke which was useful great and sometimes insight to how it broke. I felt like the m0 demo for GDB was good in a sense that it helped smooth out problems with running it (and adding break points etc.) . However, it was to print out a variable which i didnt really use that much because I just used a printf
It wasn’t helpful for my nfs related bug
IT brOKE :((:(:(:(:(Very specifically, GDB seemed to have stopped the execution of the program in breakpoints, and it's initial connection/stop worked. However something went wrong between it recieving more data or something? It basically didn't update the cli beyond that and since the program's execution stopped, it was basically a waste of time.
GDB broke after a bit and it would've taken too much time to figure out why. Note that we had multiple threads in our kernel. I'm guessing that GDB would be less likely to break on an event based model.
Awesome for bugs that caused program to die, not so good for more abstract bugs

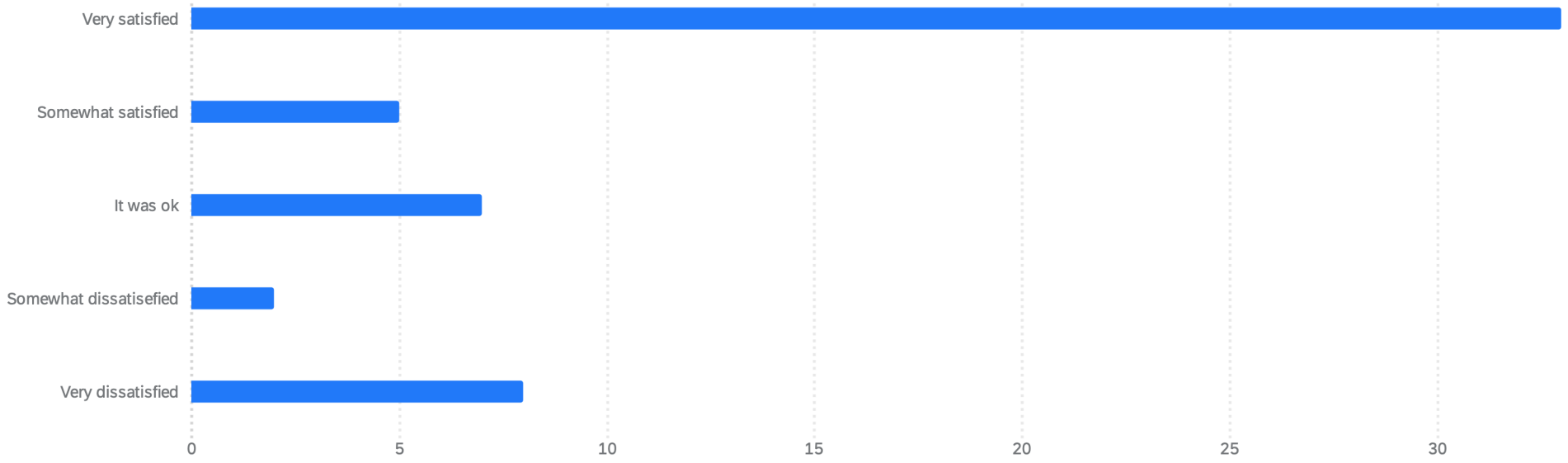
Q24: Comments on GDB? (4/4)

Any other comments on GDB use?
I remember I did try to use it for the later milestones. However by that point the codebase was already so buggy and terrible that it wasn't terribly useful to me and we ended up just rewriting a lot of problematic components from scratch
I think it is super useful. Sometimes I couldn't use GDB to debug on some particular threads because we did not spend the time to get it properly working with GDB, but 95% of the time it is very useful to debug
Our GDB broke halfway through M5 and thus we were not able to utilize it after that... so we probably would have used it more if we could have found the time to fix it
I find it mostly useless.

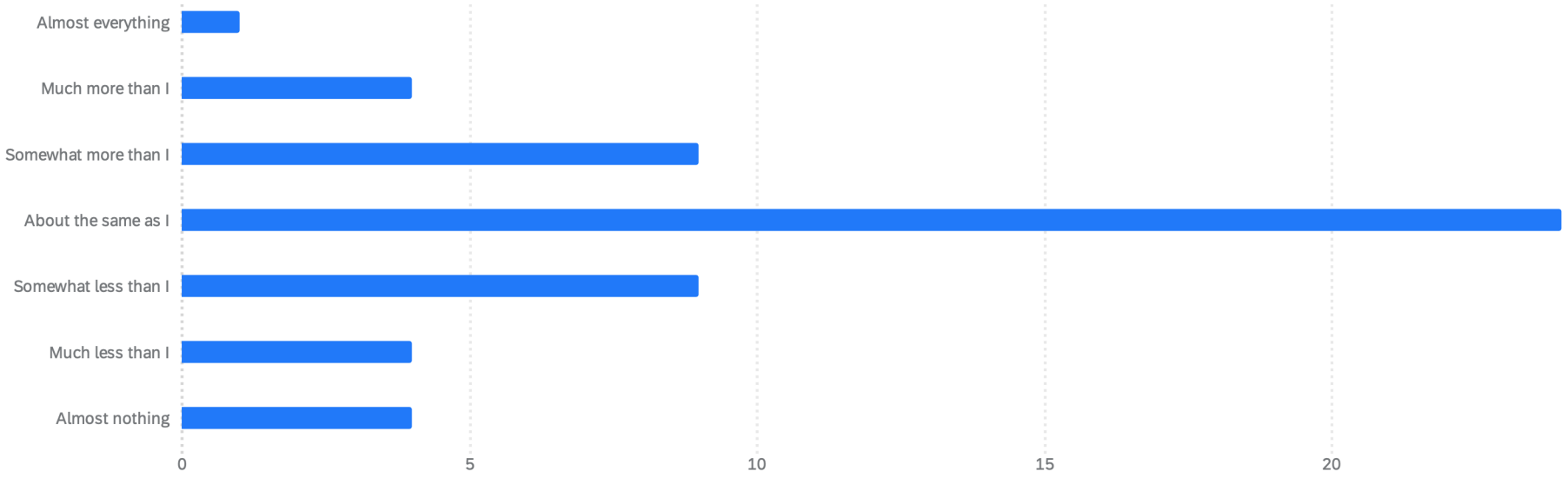
Q25: How did you find your partner?



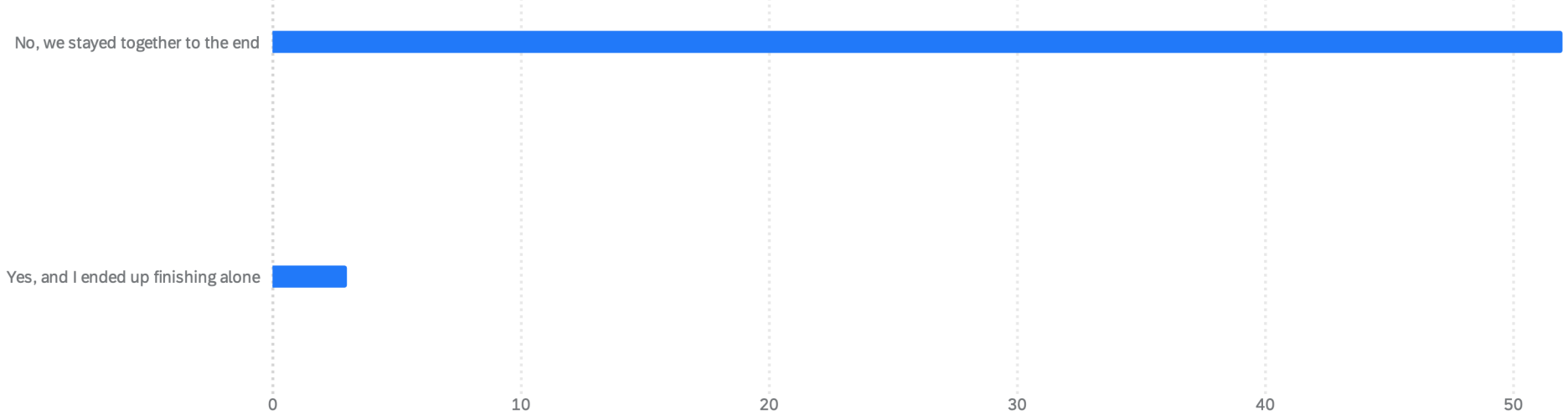
Q26: How satisfied with your partner?



Q27: Fraction of the project the partner did



Q28: Did you split?



Q29: Other comments on partners (1/3)

Communication is key. Despite having a close friend as a partner, we sometimes faced decision paralysis, as well as leaving the other to say something first. Perhaps providing a couple of quick pointers/reminders early in the course could help groups get into the right headspace.
my partner overestimated their capabilities, not really much that could be done about it
Found partner on CSEsoc discord. They were pretty good
Our group had a bad execution model which forced us to rewrite the entire OS before M6. I was happy to rewrite, as it would've been hard to do together. I do want to complain that we weren't given any foresight by the demo/staff, which could've prevented this big time sink, as our group was left in a bad position.
Maybe send out a form a few weeks before term with a poll on expectations for the course and offer to match people with similar expectations?
Met them in the (unofficial) class discord. Maybe next time an official one should be made and students should be encouraged to join
Would be nice to have some sort of board for people whose partners dropped and would rather not solo the course to put themselves on to find new partners more easily.

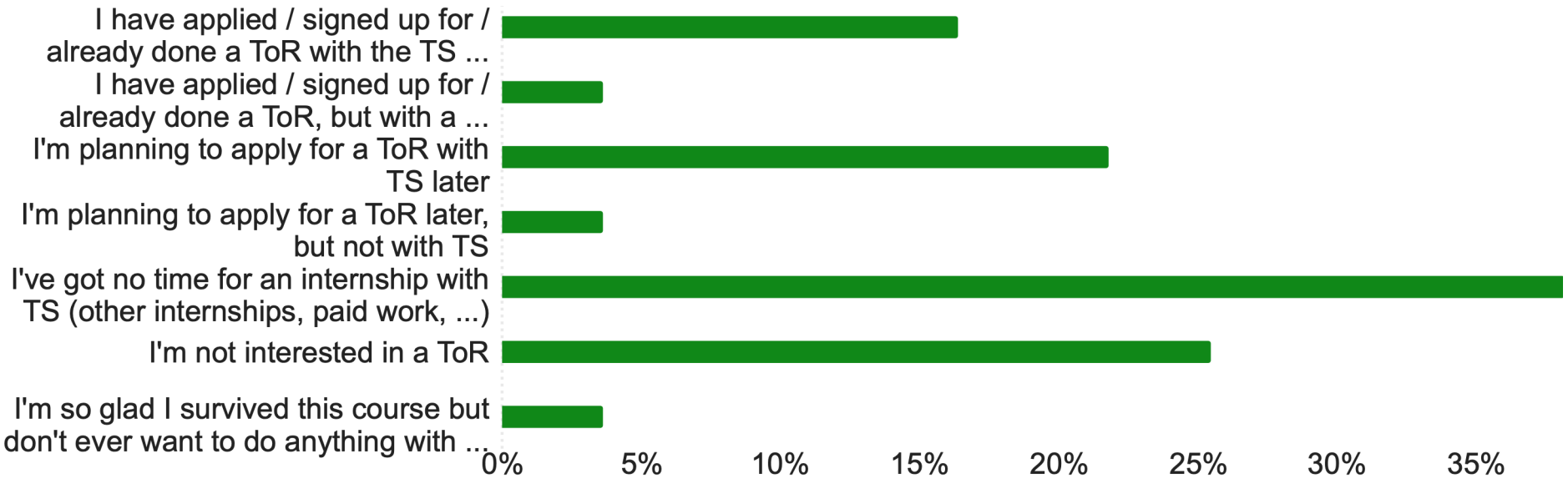
Q29: Other comments on partners (2/3)

I found my partner on the CSE Discord, I'm not sure if that's what you meant by forums but the discourse forums were empty at that time. I don't know how exactly the course can help with my issues, because they only became apparent in m3 onwards. On paper the course prerequisites are strict, but I really don't think my partner was qualified for this course. I feel like m0 could perhaps have a stronger weeding out component? Or maybe not weeding out but more like bonus optional tasks? Basically it would be nice to have a builtin way to find partners of equal skill level, if desired. This is on me for not vetting properly, but I felt like I didn't need to based on the requirements to even take the course and I was wrong. My main issue is that by the time I needed to rely on my partner, its too late to do much about it (I don't expect them to gain competence overnight, and I don't want to risk swapping). Also, a minor thing overall, but it would be amazing if the report could be graded individually (by tagging sections). This was a time management issue on our side, but I had finished my part of the code and my part of the report, and I was waiting for my partner to finish his part of the report. We have different standards of writing and polish, and I feel like I shouldn't be penalised for my partner's lack of polish. I also feel like I shouldn't need to sanitise his work before submitting, which I couldn't anyway because he was running late. This doesn't apply to the code because the milestone marking system is very generous and we have time to polish up afterwards. Another solution would be to recommend doing the course solo more / discourage it less. I feel like it is more doable than people make it out to be, and I feel I would've be more productive solo than with my partner. By the way, the survey textbox for this question is the tiny single line one, for arguably one of the more important questions in my opinion, not sure if this is intentional or an error.

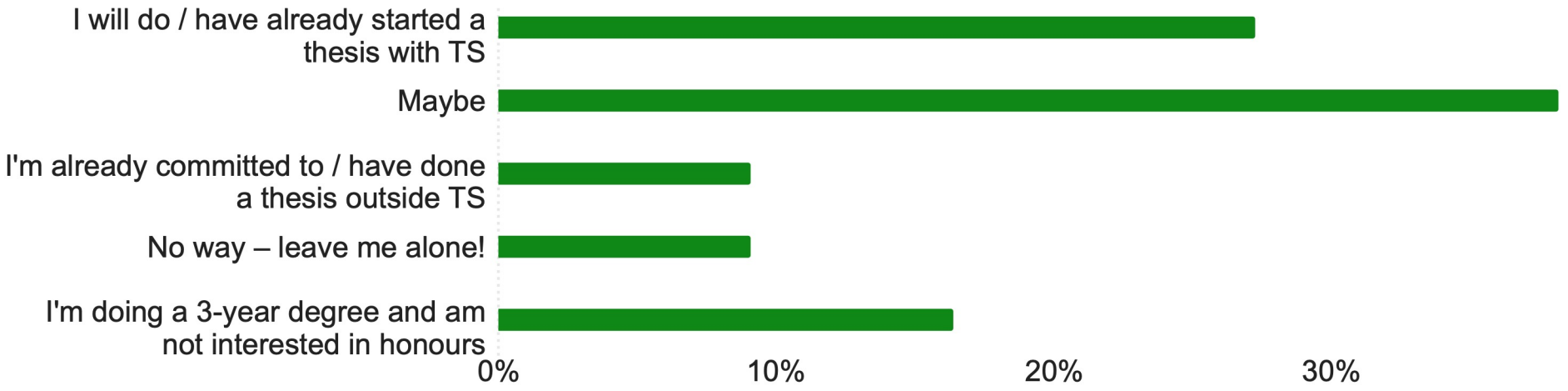
Q29: Other comments on partners (3/3)

Partner was a highschool friend, so no complaints here
1st partner did almost nothing then he quit. Saying anything about my 2nd partner would reveal my identity, but needless to say, I finished the course on my own.
found partner on course discord
Some parts were hard to split up so it resulted in whoever was more available doing the work first, and then it wasn't easy to take over for them halfway or do the leftover tasks
They reached out to me from a discord server
I'm not very vocal about the way group work needs to be done. Allocated tasks were done sort of isolated, which meant that a lot of the thinking that goes behind code is lost, even if you can explain the functionality. As a result this made demos quite stressful as I didn't implement something my partner did, and hence couldn't communicate better understanding when being quizzed. Maybe this is indicative of underdeveloped critical thinking skills on my end. I would like there to be check ins on teamwork during demos not just the code, which will prevent such experiences in the future.
Maybe can have a partner allocation system for students who enrolled solo, random teammates do traumatize people

Q30: Interest in Taste of Research with TS



Q31: Interest in Thesis with TS



Q32: Comments on TS ToR or thesis?

Any comments on TS internships or theses?
Keen to start!
I would've done it if it wasn't for the fact that I'm unfortunately an international student
I am interested, but I have heard that it's a high workload. I'll be in my 5th year in 2026 so I'd probably want to do a thesis instead of another TOR
I'm an elec student who has not yet found an elec supervisor, so potentially a TS thesis might be something I'm interested in
id be interested but im currently already working 😓

Q33: Final comments

Any final comments you would like to make?
Make it 12 units! I had to drop a course to keep up!
I really enjoyed the course!
Just was awesome. Litearlly excruiactingly difficult but so happy to get through.
Thank you Gernot
Nothing to add, just want to say thank you for a good course.

The End