

## DR CHRISTIAN URBAN

Practical Experiences of Programming (5CCS2PEP 2025/6 SEM1 000001) (5CCS2PEP-2025/6-SEM1-000001)

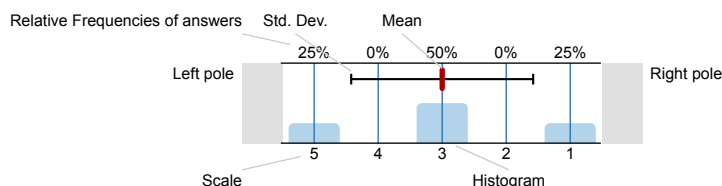
No. of responses = 86



## Survey Results

## Legend

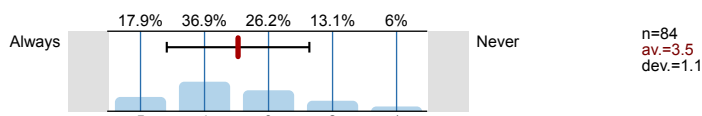
Question text



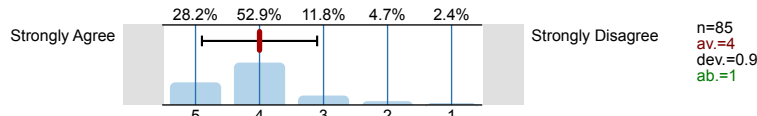
n=No. of responses  
av.=Mean  
dev.=Std. Dev.  
ab.=Abstention

## 1. Practical Experiences of Programming (CORE) - Practical Experiences of Programming (5CCS2PEP 2025/6 SEM1 000001)

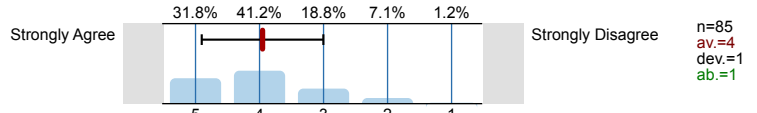
1.1) How often did you attend lectures, seminars, or tutorials for this module?



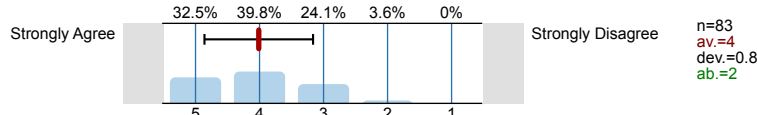
1.2) The module was well taught.



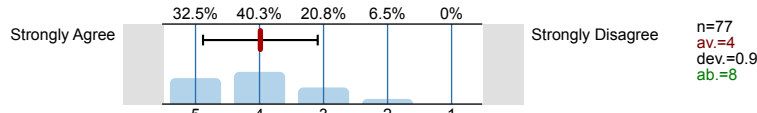
1.3) The content on this module was explained in a clear and accessible way.



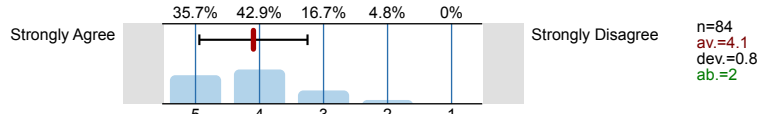
1.4) I felt supported in my learning.



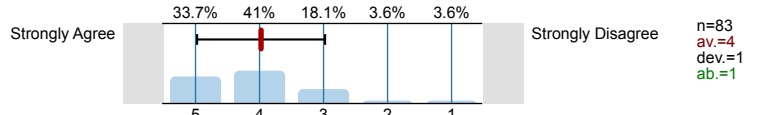
1.5) The feedback I received for improving my work was helpful.



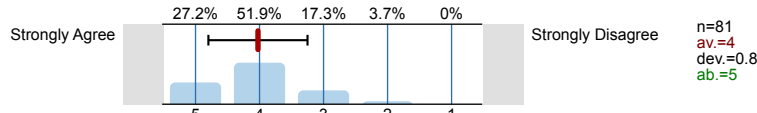
1.6) The subject-specific resources (e.g., readings, equipment, facilities, software) were easy to access when I needed them.



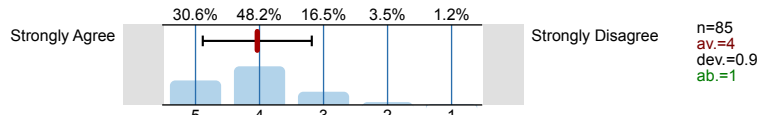
1.7) The module was well organised.



1.8) I felt included and encouraged to participate in this module.



1.9) Overall, I am satisfied with this module.



# Profile

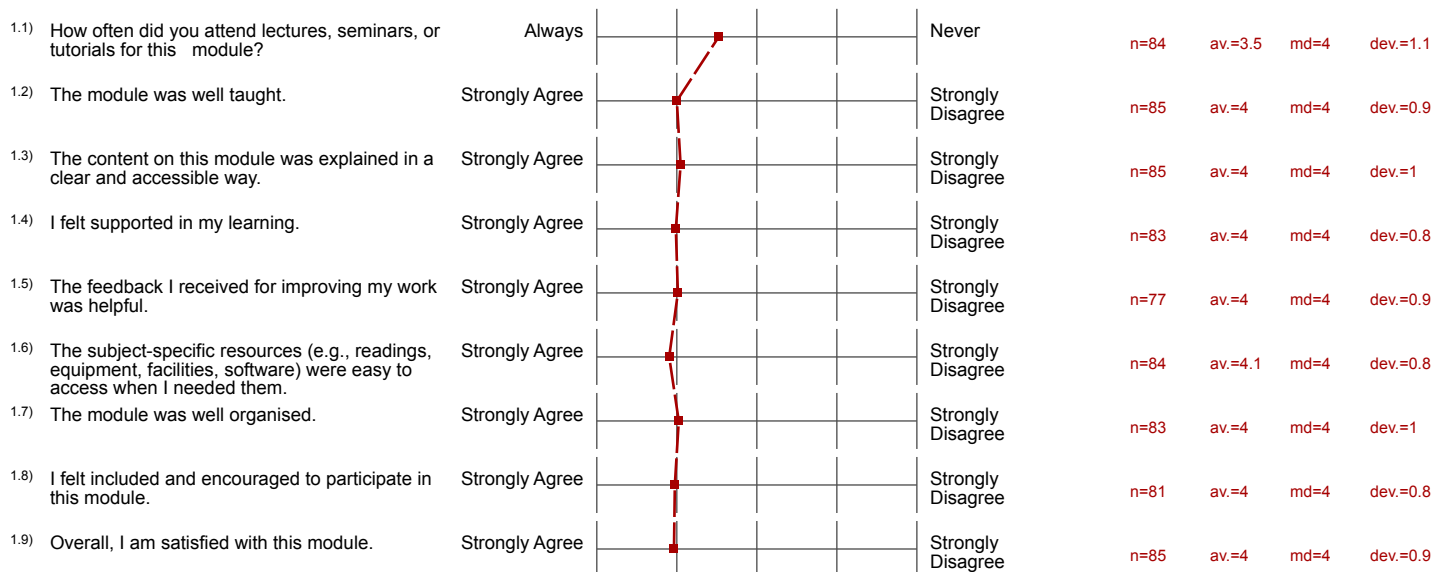
Subunit: Informatics

Responsible for modules: DR CHRISTIAN URBAN

Name of the course: Practical Experiences of Programming (5CCS2PEP 2025/6 SEM1 000001)  
(Name of the survey)

Values used in the profile line: Mean

## 1. Practical Experiences of Programming (CORE) - Practical Experiences of Programming (5CCS2PEP 2025/6 SEM1 000001)



# Comments Report

## 1. Practical Experiences of Programming (CORE) - Practical Experiences of Programming (5CCS2PEP 2025/6 SEM1 000001)

1.10) What have you enjoyed about the module?

- Being able to learn new languages have been insightful, and I've come to really love both C++ and Scala
  - Both lecturers seem very easy to approach. They also continually answer the questions given to them in the forum, which is much appreciated since it takes a lot of effort to do so. I also found the lecture videos and explanations given by Dr. Andrew Coles beneficial.
  - Coursework, its level of difficulty, and its relevance to the lecture material -> very satisfied
  - everything
  - Firstly, the relevance of the languages. C++ particularly is one that employers want! I think one of the teams I am doing a placement year with next year uses Scala as well so even better. KCL please keep doing this - don't stop here and please look into teaching us more industry-relevant technology past coding languages. Eg containerisation with Docker, CI/CD.
- In addition I thought the teachers are good at their job. I struggled sometimes with understanding concepts because of the busy time period coupled with just 5 weeks of learning for each language - but the lecturers did a lot to explain things in different ways. Particularly Senir.
- helpful videos, engaging coursework but c++ was difficult
  - I enjoyed the lectures as the lecturer was very approachable and answered all questions very well!
  - I enjoyed the tasks given during the scheduled lab sessions, and the freedom of being able to approach the coursework in any way and at any time.
  - I enjoyed the way the lectures were presented.
  - I enjoyed writing the coursework for cpp and scala. Very well taught, very good TAs very informative and responsive in the forum. No complaints
  - I enjoy the assessment process where we are required to create a working program without being assessed on things such as code quality.
  - I liked learning C++.
  - I liked the depth in which we covered the content .
  - I liked working with C++, the assignments felt like good problem solving exercises.
  - i loved the videos and corresponding demos, this helped me not only to understand the concepts but also see it in action. explanations were really good.
  - I really enjoyed the structure and exercises of the courseworks. I found functional programming quite fascinating.
  - I really liked the live coding aspect of the lectures and felt like it allowed me to really cement my knowledge. Due to it being so hands on with an open floor to questions, I felt very positive that I was going to gain understanding in whatever topic we were covering regardless of whether or not I was able to fully grasp it from the videos
  - It's really a great module and Senir Dinar is really good at teaching it.
  - It never felt like I had to learn a new C++ concept and I felt that I was taught everything I needed to know for the basics of C++ and how it worked, including reasons for why things are as they are. While I struggled with the computational logic in the coursework I enjoyed working on it and never felt lost like I didn't know what I needed to do, I just needed to figure out how I needed to implement what I knew.
  - learning new language especially scala
  - Learning new languages
  - Learning new programming languages felt helpful and a good self-improving journey in terms of a software engineering related path.
  - The coding examples for c++.
  - The coursework
  - The coursework 1 was quite challenging but rewarding to complete. It was very interesting to learn about C++ and Scala. The lecturers and some TAs were fast to respond to emails also.
  - The lecturers both really are very clear in their explanations, I really liked the coding demonstrations, they really helped consolidating the theory in the videos

- the P v NP problem and the idea of NP hard like its just cool
- The scala part of the module was perfect. Especially going to LGTs with Christian and listening to the interesting discussions were very enjoyable.
- The videos explained concepts quite well, if sometimes verbose. Senir was friendly and made the module interesting. Was not able to attend scala lectures so no comments on that.
- This module felt complex however had fun in learning the content and learning C++ and scala where scala taught me a completely different way of programming. There were also a lot of examples given in the content.
- This the most module I have nothing to say bad about. Well Done.
- This was an extremely challenging module that did not hold back in its first coursework. However, it also gave me good skills in programming in C++ and how to approach problem solving in both imperative and functional programming.

1.11) If you could recommend any improvements, what would they be?

- 5 weeks is a tad too short to learn a language like C++ or Scala! Right now I feel like I know enough to pass the leetcode-style questions, but I am not nearly ready to build a larger scale project at all in either language, nor do I really know what that looks like. For Java (while we had 4x the amount of learning time) I know what it looks like. Also for scala I don't know if it is because it was a very busy time or the language is less familiar to us, but I had a hard time understanding things. Mr. Urban is very knowledgable I can tell but I think he really hammers in why Scala is great compared to every other language - I would rather just be told the details of Scala and make it up in my own mind if it's great. It also would distract me less from learning the content. That being said Scala does seem quite cool!
- already good
- Choice of content progression and the expectation of knowledge was confusing alongside the requirements for the coursework. It was never made clear whether what we were learning was useful for the coursework, and how it would be able to be integrated. Ultimately, a lot of knowledge wasn't needed
- Dont teach to fast and try and cram everything into one lecture. At the same time, dont teach too slow to the point where nothing gets done and it doesnt feel productive
- honestly nothing, but i do think maybe quizzes that can help for people who arent as confident with c++ for their own practice kind of like ppa in first year. it helped with a new language to test and see what i understood and what areas i needed to work on.
- I'd suggest spending more time covering C++ than Scala because C++ still prevails in the industry today and there are many really important C++ concepts to be covered.
- I'm not sure how this could be solved but since the coursework deadline for the C++ coursework was after the time Dr. Urban took over the LGTs, it seemed not many people might come to or listen properly during his LGTs due to their focus on the C++ coursework as opposed to the Scala one since the deadline would be approaching, which takes away a valuable opportunity to
  - 1) deepen our understanding of Scala for the second coursework
  - 2) ask questions focused on Scala
 since most people are focused on the closest deadline first and it's difficult to be focused on two things at once. Perhaps adjusting the deadlines might help?
- I believe there should be more materials for beginners in C++ to practice and prepare themselves for the coursework with. I had expected the first lab session to be this, but left discouraged and thought myself in no place to tackle the coursework.  
  
While I understand the thought process of introducing new concepts in labs, too many of them can be overwhelming, especially for beginners. Smaller exercises that introduce concepts that weren't taught in the videos (like structs) would help in this regard.  
  
Furthermore, in the case of the first lab session, an "intermediate" level where the static variables and method signatures being already provided, as well as comments on what each method should do would have been useful. The idea being to remove the difficulty in coming up with ideas for solutions oneself and focusing it on writing code. As it is, beginners would likely only write a shuffle algorithm in the easy level of the lab, which isn't much practice at all.
- I found the feedback from Github very useful, however setting up the repo was a bit of a long process and slightly difficult to set up, though I agree a lot of support was provided on this.
- I think Scala should be taught first than C++. In addition, the general content and coursework of the C++ side of the module should be revamped from scratch, because the current content, the coursework, and the lecturer has not been helpful in my learning.
- I think that we could have covered more content.
- I would prefer if we were guided better towards how to approach the problems we were given. I felt the guidance given actually worked against me and I found the problems much easier once I decided to do things a different way instead.
- I would recommend breaking up the lectures into parts, instead of continuously building on code because if someone falls behind in understanding at one point they're unable to follow on for the rest of the lecture. I would also recommend using Menti so students can interact and see how much they know via quizzes.
- labs are difficult to engage with, the jump in knowledge required to complete them is high compared to what's been taught in the materials
- less hard maybe

- Longer time period to do coursework
  - Make c++ coursework due earlier so we dont have so many courseworks at the same time at the end of term.
  - more content related to the actual coursework
  - More formal introduction to the theory of functional programming.
  - More weekly quizzes like in week 5 for cpp where you actually write code.
  - Most of my grievances with this module will be with C++. I think this module should not be as dense or fast paced as it is. It felt like drinking water from a fire hose at times and almost felt like competitive programming / leetcode knowledge was a must. With regards to some of the problems proposed in the C++ coursework, I am not sure exactly what the industry relevance was besides being hard leetcode interview problems.
- I think the lecturers / materials could also attempt to bridge the gap between coding in C++ and approaching how to solve the given problems though. The coursework and lessons felt worlds apart in that sense.
- Providing more programming exercises for each week's content to challenge what was learnt during that week worth of content, and perhaps incorporating small group tutorials where effective solutions to those exercises are explained and walked through alongside a T.A.
  - Some TAs were not as helpful giving tips for the coursework. The C++ videos were a bit hard to follow as there was a lot of content really fast.
  - Sometimes the videos are too long. Senir sometimes gets lost in his train of thought, and it is a bit confusing. Was not able to attend scala lectures so no comments on that.
  - Sometimes the videos felt a bit long for C++
  - The C++ coursework felt a lot more challenging than I expected which I understand as it examined half the course. However as someone who didn't code much before university I watched the lecture recordings and videos hoping to understand how to tackle the coursework. Due to the heavy emphasis on computational logic however I felt that this instead disadvantaged me as it took up a lot of time and effort that wasn't needed and would have been better spent working on the coursework. Everything I needed to code the coursework was covered but it didn't feel like the coursework tested C++ at all rather I was tested on how well I understood the assignment and the logic behind the implementation. I never felt the need to use many concepts like C++ pointers, unique pointers, inheritance, maps and other such features. Lacking the ability to import classes meant I didn't have the option to do so oftentimes either. The labs did not give me a headstart on the assignment and didn't line up correctly either hence I disagreed with the organisation of the module.
  - The lectures are largely a waste of time. A lot of time is spent covering things that don't actually get applied, even in the module itself. It feels like that time could be far better spent covering some content that would be useful for the coursework, rather than just sidetracking with no organisation and bringing up random concepts that aren't mentioned anywhere else
  - The mic was a good idea but a little tricky when people who sit quite far apart wanted to ask questions. Maybe having multiple people on mic duty would help.
  - The Scala lecture is just a repetition of the videos, so I would prefer if we could go over new examples
  - The videos are very long and there are so many of them. It avoids going over the important base content that we may have already covered in a module like PPA last year, which makes sense, but it goes over very niche and circumstantial techniques that will already become forgotten. The best approach for the coursework was to learn online through the internet as you went along, so there was no point in me watching the content and I barely interacted with it at all.
  - The videos for C++ are really hard to sit through, I get lost in it
  - To spend more time on the coding elements of a language themselves - to increase students' depth of understanding through application.