Informatics Student Course Feedback 2017/18

http://www.inf.ed.ac.uk/teaching/surveys/2017-18

This report contains feedback from students about a course taught in the School of Informatics during the 2017/18 academic year, in response to the following questions:

- What would you say to students interested in taking this course?
- What did you find most valuable about the course?
- What improvements, if any, would you make to the course?
- Please add any other comments you have about workshops and tutors

Each course organiser receives this report as well as statistics on multiple-choice responses. All these reports, together with student feedback about individual members of teaching staff, are collected and sent to the Director of Learning and Teaching.

Please note that these are personal responses from individual students: some courses only have a few responses and a small sample can be unrepresentative.

Stereotyping and bias, especially unconscious bias, is a serious concern in any survey gathering personal responses. All students received the rubric below before completing the surveys, and you can read a brief introduction to issues of unconscious bias on the university web pages at http://edin.ac/2iypZBv

Rubric given to all students taking the end-of-course feedback survey

We value your opinions on the courses you take here at the University, as they allow us to shape future delivery and development. We welcome constructive comments about your courses, whether positive or negative, and ask you to give details about any issues in order to help the course organiser to understand and address them.

We encourage you to be aware of the potential for bias in the completion of these questionnaires, so we have developed resources which may be helpful to you:

- Equality, Diversity and Unconscious Bias (http://edin.ac/2iypZBv)

You also have a responsibility to provide feedback in a manner which does not breach the University's Dignity and Respect Policy:

- University of Edinburgh Dignity and Respect Policy (http://edin.ac/1Cq0VZY)

The results of the questionnaires will never be analysed in a way that seeks to identify individual students from their responses. However, should you wish to remain anonymous, please do not identify yourself in your answers to the survey questionnaire implicitly or explicitly.

Comments Report

What advice would you give to a student taking this course in future?

- Although you may know java, go to the tutorials!! The exercises made you understand the concepts rather than just programming
- Do all the labs
- Do as many of the exercises as soon as possible, do some past papers on DICE so that you know how it works
- Do the exercises and go to tutorials
- Don't dodge the lectures even though you feel confident doing the tutorials yourself - there's extra content in the lectures beyond syntax and you'll get a better idea of what might come up in the exam.
- Don't fall behind with the lab exercises; programming can only really be learned by doing it. Also, try out the programming club.
- Don't let exercises pile up. Study systematically and ask questions.
- Don't rely on this course to learn programming. You will need to learn a lot more to be considered even an average programmer.
- Don’t fall behind on the labs
- Explore the advanced section of the lab exercises.
- Get into groups with other students and seek out advice from your peers. There's definitely going to be at least a few people in the class who have done Java before and can help you out, don't be afraid to approach them in labs or ask on Piazza.
- Get to grips with data structures before arriving in Edinburgh
- Go to tutorials, and try to keep up with the lab exercises - practice is essential.
- Have fun with the labs and by extension the language. If you have an idea for something small you could do with the language, do that thing - something you feel invested in completing will do more for your general work ethic and confidence in a language than an exercise. You are not in this course to learn which answers to give, but how to get them.
- If you already know OOP, just do all the labs at the beginning of the semester and do exam practice for the rest of the semester. Don't bother with lectures or tutorials.
- If you are new to objects and classes, do read around the subject. It might be difficult to understand from lectures alone.
- Learn Java in the summer before you come to uni because Calculus will demand all of your time.
- Make sure you keep up to date with the lab exercises
- No advice, the course is easy -> you don't have to go to the lectures, if you want look at the tutorial exercises, but it's probably not necessary for the exam questions, so just take care that you now the organisational stuff such as passing the basic tests, auto marking, including Junit test etc.
- Practice makes perfect
- Practice more.
- Sure.
- This is the basic things about java.
- do all the lab exercises
- if you’ve never done object oriented programming before familiarise yourself with the basic concepts and maybe take an online java course

Comments Report

What did you find most valuable about the course?

- All lab exercises were posted at the start of the year so I could do them at my own (faster) pace and therefore have more time for other more time consuming courses at the end of the year
- All.
- Classes and objects, the way to code real life objects was interesting
- Every lecture is efficient and fast-paced.
- Examples, pieces of real code.
- For someone who hasn't used Java extensively the course was a good opportunity to familiarize with the language.
- Hmm... not sure anything stands out, but I guess.. the lectures? They were pretty consistent, solid looks at the language that helped introduce each topic.
- I hated it. Seriously. It was too fast and only few lectures can't make us understand anything.
As a beginner, it was not at all good. The lecturer is good, but not understanding of us beginners. It was way too fast for me.
- I thought that the lab exercises were quite well designed and I thoroughly enjoyed doing them. They were certainly the best features of the course.
Lectures were also really good. In comparison to other programming lecture, Volker was very good at keeping the lectures interesting and well taught. Even though it was his first year teaching the course, I think he was quite good at it honestly.
- Labs
- Lectern recoding is very helpful.
- Mock exam was a wake-up call.
- Object-oriented programming is an essential skill which this course helps you develop.
- Refresher on OOP.
- The labs and tutorial exercises
- The lecturer was very good and gave us plenty of opportunities to join the programming club and encouraged to do things outside of the curriculum
- The online learning material, and the mock exam.
- The programming skill
- The tutorials - going through 'does it compile and if so what does it print' was an especially helpful exercise in understanding the concepts in action.
- The tutorials for early weeks are well explained.
- The way the lecturer compressed such huge topics into 9 lectures and did it so well.
- Tutorial exercises were very well explained and easy to follow
- lab exercises
- Lab exercises and tutorials. Getting people started on OOP

Comments Report

What improvements, if any, would you make to the course?

- A book that is actually available online (the link doesn’t work). More support for those who want it! Help using own machine!
- Felt it didn’t really cover that much of Java
- Go at a faster pace and cover more material. Just like how I would rename "Functional Programming 1" to "Haskell syntax", I would rename this course to "Java syntax", because that more accurately represents what was taught. I would have actually enjoyed the course if the focus was less about "this is how it's done in Java and that's that" but more about "this is why Java implements OOP concepts as it does", covering OOP design problems involving multiple inheritance, the difference and pros/cons of composition vs inheritance, examine how other languages achieve the same features as Java without being object oriented, looking at common OOP design patterns, etc.
Another thing I would suggest is looking at large scale examples of OOP in action. At the end of the course most students understand the basic principles of OOP really well, but many of them would significantly struggle if asked how to structure a large program such as a game or a desktop application. I think clever incorporation of real world examples into the weekly assignments would be a good approach.
- I would put greater emphasis on explaining what an object is. On a related note, I would also put more time into explaining what really happens when you 'assign' an object to a variable and show it how it's different from primitives.
- I'm not quite sure - this felt a lot more like a 'here's a language, learn it', which I understand is kind of the desired approach. Maybe small, optional, submittable coursework throughout the semester? Kind of in the style of the past paper auto-marker, but on a much smaller and more manageable scale.
- If there were enough people (can be found out by doing a survey before the year starts, like the mathematics diagnostic test), I would run an alternate advanced programming pathway, that would be much more accelerated than INF1-FP and INF1-OP
- It was too boring. In the beginning far too slow, but of course it is not easy to organise a course that is challenging for everyone. A bit more straightforward help for doing some advanced projects would be great. I'd would have loved to do something, but it is not always easy to stay motivated while working just by myself. If there would be more active session for teaming up or/and some kind of presentations for projects people did during the course, this would be very motivating I think.
Maybe something like that could be implemented.
Another option would be to create tutorials of various difficulty. So that someone who has already got Java experience does not have to sit bored in an easy Tutorial that doesn't teach them anything, but can get together with other people and work on more challenging problems and questions.
- Less Java focus, but more programming concepts themselves
- Teaching how to use programming tools
- Make past exam solutions available.
The exam not allowing USBs like in FP is less than ideal.
- Make the course a bit harder, go a bit more in depth
- Maybe have more extra stuff that isn't "Go to Oracle and do X exercises" idk my motivation really dropped when it was completely separated and self-guided. I'm not too sure why though, I would be open to discussion [content deleted]. It would especially be nice to have alternate tutorial sources because while Oracle's stuff
- More attention to tutorial exercises and improvement of tutorials overall. We didn't get any feedback from the tutors, we didn't solve any of the lab exercises in the tutorial and they were not organised.
- More lectures
- More practice questions for beginners.
- More theory would be great.
- No improvements to suggest
- No.
- The exam grading criteria seems quite harsh, and I would make it a bit more lenient.
- The syllabus is very limited for a university course. It shouldn't be tailored to students who have no programming experience as that was the purpose of inf1-fp and every student will need to learn programming on their own in the future anyway.
Concepts like generics and class design should be discussed more extensively. The Java API should have been explored more deeply (serialization, network i/o are things which all students will need to an extent).
Since the school doesn't offer a dedicated web development course this course could be used for that; particularly server-side scripting.
- have more lectures
- more lectures and tutorials
- none this has been my favourite course all year not sure

Comments Report

Please add any other comments you have about workshops and tutors

- Hugh was largely supportive but intolerant to those who fell behind.
- I am disappointed with the tutorial structure because even though the lab exercises were challenging and fruitful, the tutorials didn't play any significant role in learning and understanding the exercises or the concept.
- I get more work done at home rather than attending laboratory/tutorial sessions.
- I know this answer is tinged with the effects of the industrial action, but I don't feel like there were enough of them. Also, we began by working through some problems in a more presentative-cooperative manner, and I think that was a lot more helpful than some of the later work which felt more of the style of everyone working on their own individual problems.
- I mean they were okay but I didn't find code review especially helpful, I personally found the more foundational explanations about why Java makes X decision or like performance type stuff to be more interesting.
- I really liked my tutor, although I have only been able to attend one session in the whole semester. I think tutorials are great for those who don't really know what they are doing.
- It is very useful for individuals.
- Make the tutorials more consistent across the board (i.e. have all tutorials the same as some groups had different things to others)
- More challenging tutorials for people with Java experience would be nice (advanced tutorials that replace the normal tutorials for those students who decide to sign up for them)
- My tutor seemed like he did not want to be there, and in addition to the content of tutorials not being useful (ie. nothing new taught), I stopped attending them.
  It would be more useful if they were FP style, ie. Solutions to that week's exercises were discussed.
- The tutorial time was too short.
- The tutorials have been mind-blowingly boring. To be fair the lectures were probably just as boring, but being in a room with just 5 other people listening to someone slowly explain these OOP concepts everyone is already very familiar with is just painfully mind-numbing.
- The tutorials were the most helpful part of the course
- The tutorials were very helpful, see above comment.
- Tutor was great and stimulated my interest more than the lecture, but attendance was low which meant there was not so much room for discussion
- Tutorials are so good
- Tutorials were quite standard, and somewhat boring in my opinion. I much enjoyed the other aspects of the course, and the labs were wonderful.
- Unlike inf1-fp this course doesn't have a lot of advanced (optional) exercises. The ones which exist should have been part of the core exercises (and they are in terms of difficulty).
- While finding my tutorials beneficial - personally, my tutorials weren't well run because we had a different tutor each week. I think having a consistent tutor would have benefitted my learning.
- none
- [Please note that content has been removed due to identifying factors of student data]