# 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?

- Avoid if possible.  Dull.
- Do all the tutorials and start all the assignments early, they take a lot of time and coordination with another member of your course so be organised!
- Don't spend too much time on courseworks, marks are likely not to correspond to the amount of time spent.
- Forget lectures, learn content from text / slides
- If this is an outside then drop it for computer systems, or do a 20 credit course instead. If this is compulsory, well have you considered changing degrees to make it optional. You aren't learning computer science but all the administration and management about software engineering. All this is useless. Google or the big companies won't be impressed with this knowledge. They have their own ways of managing software development. They are looking for actual smart people not these kind of people that are able to a whole load of lies in a report and claim themselves to be software engineers. This course shouldn't exist.
- If you are deciding between CS and SE, bear in mind that SE is probably easier content but the fact that it is boring makes it harder.
- If you're expecting a practical, hands-on course, you will be disappointed.
- Make sure you are comfortable with Java programming.
- Make sure you know how to program in Java. Be expected to spend a lot of time on the coursework in comparison to the workload of a 20 credit course.
- Outside students: take the course, it's mostly just memorisation.
Internal students: take this course if you just want to do fact-recall, don't take DMMR. 2C-CS is good for you if you like coding.
Internal students who have to take this course: good luck
All students taking this course: don't try to write good code, and don't try to be smart. this course dumbs you down and will force you to deal with and write the worst code you will ever deal with / write in your life.
- Probably don't tbh
- Read the textbooks, and try to get as much knowledge about the subject regardless of the lectures.
- This course teaches essential working and coding practices and methods if one wishes to pursue a career in informatics.
- Too many work for 10 mark course
- If you have the choice between this and computer systems, definitely take computer systems, if you don't have the choice them try and enjoy and use it to strengthen your java knowledge for a later year.

## Comments Report

What did you find most valuable about the course?

- Although the way the course is delivered is very dry, i.e. you can read it all on Wikipedia, the things in it are useful for coding interviews.
- Going through a software development process as part of the coursework
- Insight into how software development happens in the industry.
- It provided the tools and knowledge for structuring a proper design plan for larger scale programming projects, which was extremely helpful as I am building a game on Java and the tools help me make a better design, and realise my flaws in previous code.
- It was interesting.
- Learning about different ways of working for Software Engineering, and completing the Coursework in a team.
- Learning how to create UML diagrams.
- Lecture slides
- Paul made his lectures fun and interesting. I feel that the content of this course could have been really dry if it was not for Paul's enthusiasm for the course.
- Really interesting learning about different software development techniques. Lecturer was well organised
- The coursework and gaining experience of a group project.
- The final assignment definitely strengthen my Java skills
- The first assignment was a valuable learning experience because I've never programmed a larger system before.
- The first half of the course, covering requirements gathering, UML and various tool/techniques (design patterns, version control, etc...) was really interesting.
- The labs
- The lectures are almost as effective as sleeping pills, great if you need rest.
- The tutorials were very useful and well organised.
- Tutorials and the coursework done in pairs.
- Tutorials, by far
- new approaches of programming, engineering structure of software
- slides

## Comments Report

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

- Clearer learning outcomes!
- Content is very boring, the lectures are dry and difficult to follow. Very hard to determine what is course content and extra items.
- Feedback is not very helpful. Also, the actual tasks for the courseworks are so fuzzy all the time, I'd happily work for weeks on a coursework and struggle doing something nice if I knew what I had to do. Asking 1000 questions on piazza and going to labs is just confusing and frustrating.
- I believe the course was really dry, and not too engaging. I think the course would be better if it was tied in more closely with programming practicals, as only one part of the assignment is programming (and it's not truly following your own design). This would give you a better idea of what good and bad programming design is like.
- I felt there were some areas in the second half where we dived in to solutions without really addressing the problem. For instance, I felt the software processes and process improvement sections could have done with more of an explanation of what the abstract idea of a "process" was. In retrospect it's pretty obvious but I was a little bewildered for the first few minutes.On a slightly different note, I was much disturbed by your relentless promotion of spaces over the vastly superior tabs. Tabs are clearly the One True Form of indentation.
- I have to admit that I was very disappointed with this course. I was expecting the course to be far more practical and hands-on, especially much more coding, testing. I felt that we spent too much time on the general ideas like requirements analysis and design. I'm not saying that these aren't important but we could have gone over them much quicker and spend more time on the practical aspects. After all, the only real opportunity for us to do something practical was the third assignment.
- I'd explain the topics covered by the course basing on one or a few real systems, analysing how they were built, what challenges and obstacles were on the way and how they were overcame. The lectures were too abstract, making them difficult to follow and not as valuable as they could be. For the version control systems - a set of simple exercises on a zipped repository (like undo the change to file A, merge with branch R, tag release in commit F) could help the people who had no experience working with VCSs before.
- Learn a lot of outdated techniques for a large part of the course. Could be more interactive and actively teach students how to use git and write tests rather than relying on them doing it in their own time. Would be a lot better if there were case studies to look at large scale projects.
- Lectures were literally boring.
- Less memorisation-intensive exam.  Make version control non-optional for coursework and let students better experience git, etc.
- Make the delivery more interesting and stimulating.
- Make the lectures more applicable to real life, more interactive. It is very dry at the moment. Inspiration could be taken from Ian Stark! His course is also quite heavy in theory, but he puts a lot of work into adding real life examples and so it's actually fun.
- More interactive and less dry lectures: the slides have a large amount of content, and can be demonstrated in a much more interactive way during lessons, and some nice photos (and examples!) here and there is all that is required to make the lectures... bearable.  Better coursework code. The coursework is absolutely unreadable and follows terrible practices, forcing those who *can* code to put themselves on the level of someone still in the womb.
- Real-world use of the material of the course is not really clear until we start doing the assignments.
- Suggesting what improvements need to be made to the course would be like suggesting how can you can you improve waterboarding. It just needs to stop all together.
- The course work requirements are very vague. Generally solutions to questions are too. References or more specific requirements would allow the individuals to not stray from goal.
- The coursework is low quality. I was hoping to create a proper piece of software given the course is called 'software engineering' Instead we only had to fill in a template to pass some test cases. Very disappointing.
- The coursework one and two tasks were so heavy on writing. Cut this back and more programming would be great. I really enjoyed CW3.
- The topic as its core either really isn't that interesting or the course just presents it in a very uninteresting way. Learning aims were unclear as most of the practices are things that you would probably be better off learning about once you get into industry anyway, especially given how much development processes differ between companies. Shorter segments on makefiles/javadocs were ok.
- This course should be cancelled or at least it should be required, I would much rather do something interesting and challenging then learn bunch of definitions which most people will never use anyway. I think this course is helpful for people who have already done many projects and want to see how to improve productivity. Yet I think there are more engaging ways to learn this stuff.
- Use less bureaucratic language (in slides and coursework). Spend more time on motivation to learn this course.

## Comments Report

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

- Would improve the lectures at time they don't feel that exciting or engaging? Possibly use a clicker system to engage the class.
- lectures are quite boring, TOO MANY theoretical stuff which we do not apply

## Comments Report

Please add any other comments you have about workshops and tutors

- At least it helped us through the awful coursework so I can't complain too much about the tutorials
- Doing the tutorial work definitely made the content make more sense
- Felt like lab demonstrators were reading from a script a lot of the time.
- I can feel the importance and usefulness of the tutorial, but it was a pain in the ass to attend the tutorials, since very often there were only 1 or 2 other students show up in the tutorial.
- Mai Ahn Nguyen is a very good tutor, very good at explaining the content.
- My tutor Rado Kirilchev was fantastic
- Small tutorials especially useful since answers can be so subjective
- The tutorials were helpful, but some of the question sets were too short - we could finish them in about 30 minutes, and some of them were too long to be fully covered in-depth.
- Tutorial sheets are good quality. No complaints about them.
- Tutorials definitely helped me with the course, my tutor was very good and always came prepared with questions for s
- Tutorials helped and were well led