Object Oriented Programming
Semester 2 2024

Fiona McNeill
f.j.mcneill@ed.ac.uk

THE UNIVERSITY *of* EDINBURGH

Who we are

Fiona McNeill
Course leader
& lecturer

James Cheney
Leading on assessment

Adriana Sejfia
Leading on Assessment

Christopher Dalziel
Course TA

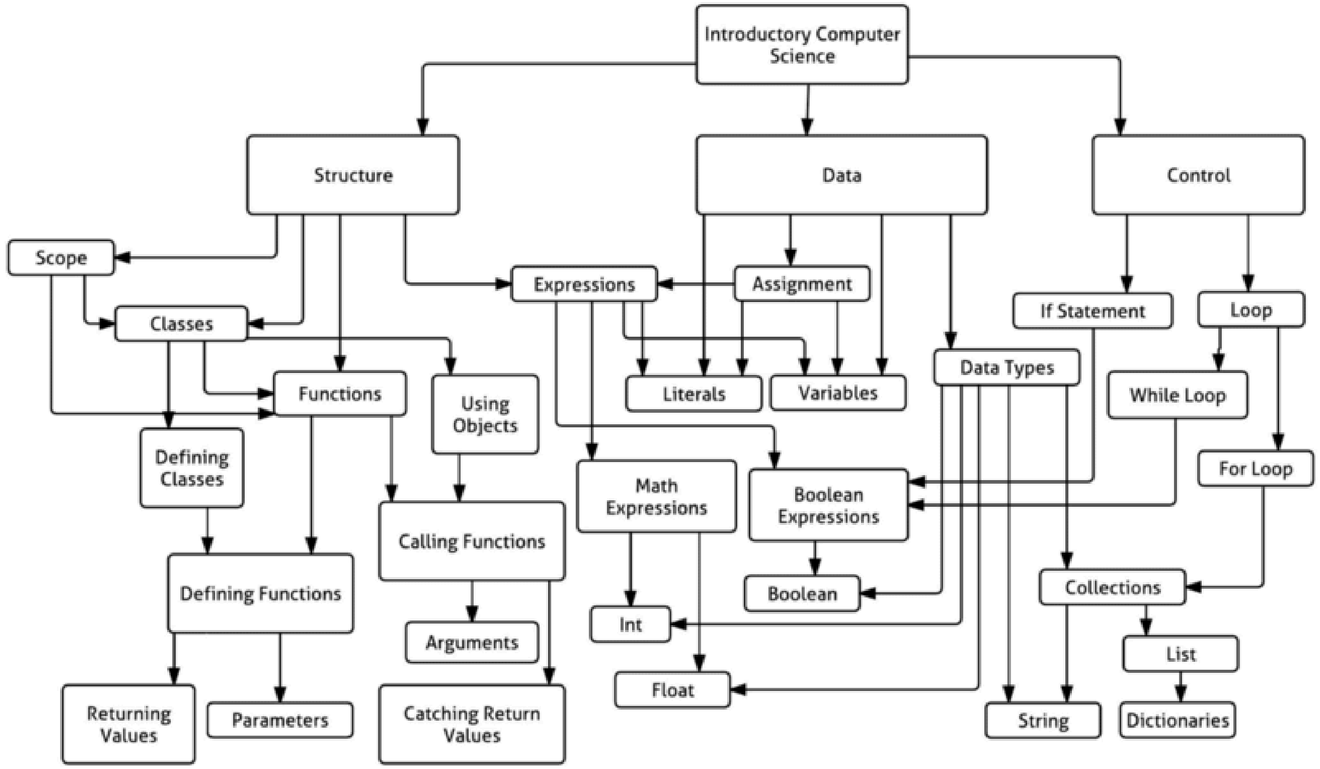
Colton Botta
Course TA

My dog

What is this course for?

- Take a very mixed group of students and build their programming skills
  - Huge range of experience
  - Narrow range of ability (you're all brilliant)
- OOP is a common programming paradigm
  - Many languages: C++, Ruby, Python, Swift, Visual Basic, etc, etc
  - Java is very popular and well supported

Learning Coding



Purpose of the course

- This is a practical course to help you create the basis of your programming practice
- The things you *learn* in this course are only useful to the extent that you are able to put them into *practice*
  - *Knowing things doesn't make you a good programmer; programming makes you a good programmer.*
- The most important thing to do in this course is to practice!

Good code v functioning code

- Writing code that appears to achieve the results you want is not the same as writing *good* code
- Good code must be:
  - Easy to read and understand - for you (down the line) and for others
  - Easy to debug
  - Easy for somebody else to take over, use, add to, etc.
- Code that works but doesn’t do these things isn’t much use
- There’s a lot of bad code out there! We don’t need more bad code in the world.

Good code v functioning code

- We have created this course to teach you how to code well, not just how to get things done with coding.

- The assessment is set up, as much as possible, to evaluate your ability to write good code, to understand what good code looks like, and to analyse the code of others with this in mind.

Code hacking vs Software Engineering

- Writing code that works is an important part of the this course - but not the only part.

- You also need to think about code design, documenting your code and communicating about code to coders and non-coders alike - otherwise code is not very useful.

- You will be assessed on your ability to do this, not just on coding!

Course Structure

- Two lectures per week: Mondays and Thursdays at 1210 in the Larch Theatre at KB. Slides for lectures also available.
- One tutorial. No brief preparation is expected - you will be solving problems during the tutorial.
- Lab exercises - to do on your own and use drop-in sessions.
- Weekly quizzes to earn your weekly badge.

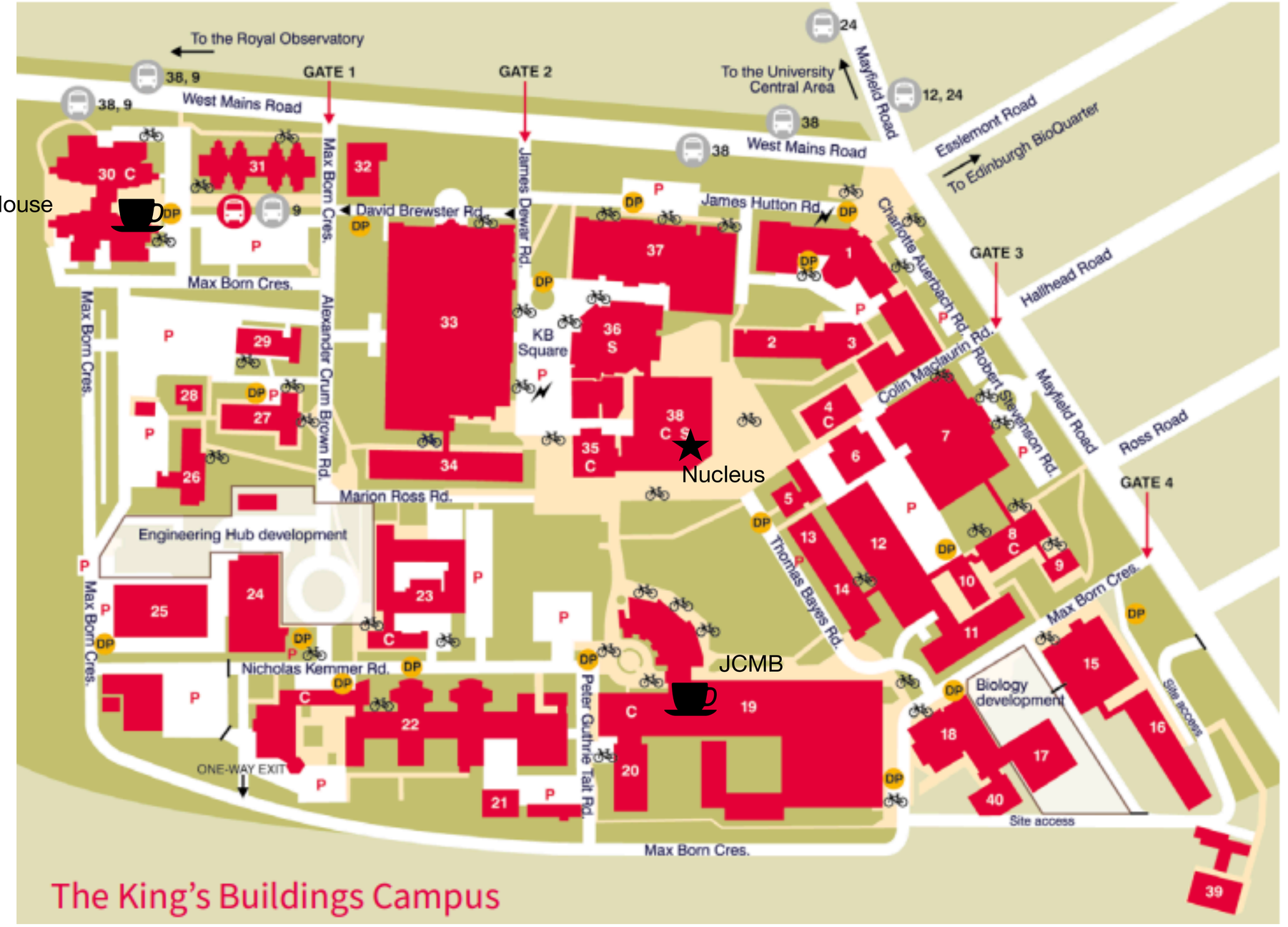
Student Voice

- Survey questions on the weekly quiz
- Ongoing feedback form where you can add your views at any time. I will check this weekly and report back on any comments and my response to them.
- Come and talk to me in my office hour or after lectures
- Use Piazza

Coping with KB

- We know that KB has drawbacks for our students
- Because of your feedback, we're hoping to change this for next year
- In the meantime:
  - All lectures will be live-streamed
  - If you need somewhere to hang out, try Murchison House or JCMB.

Murchison House



Office Hour

- TBD - we will do a poll - in AT_6.06.
- Weekly biscuit vote
- this week: ***???***

Lectures

Lectures

- Mondays and Thursdays at 1210- Larch Theatre at KB

- Introduces core concepts
- Plenty of live coding
- Quizzes and interaction
- Updates and discussions

Learn Concepts and Techniques

Labs

Labs

- Starting this week
- Drop in

Regular Practice

- Regular exercises to improve your skills - these should be the heart of your Inf1B life.

Day	Time	Room
Monday	1610	AT_6.06
Tuesday	1610	AT_6.06
Wednesday	1310	AT_6.06
Wednesday	1610	AT_6.06
Friday	1610	AT_6.06

Labs

Labs

- Starting this week
- Drop in

Regular Practice

- Each week, there are **warmup**, **core** and **optional** exercises. You should be doing the first two types, and try to do at least some optional exercises.
- There are also **advanced** exercises. These are for people with experience in Java, who want to stretch themselves. Don’t worry about these unless this is you.

Labs

Labs

- Starting this week
- Drop in

Regular Practice

- Feedback on lab exercises:
  - Use automated JUnit tests
  - Solutions are provided online (**don't peek!**)
  - Help from demonstrators in lab sessions
  - Discussion with peers, on Piazza, during tutorial (**initiated by you!**)
- Lab sessions are not compulsory but are very useful

Tutorials

Tutorials

- Starting in week 3

Basic SWE Techniques

- Tutorials are held in large groups and focus on *cooperative learning*
- Practice basic software engineering techniques, e.g. pair programming, debugging, testing, etc.
- No intensive prep required - different model to many other courses!
- Tutorials are all available on the Learn page
- Solutions afterwards

Tutorials

- Tutorials are held in large groups and focus on *cooperative learning*

Tutorials

- Starting in week 3

Day	Time	Room
Wednesday	1130-1300	40GS_LG.07
Wednesday	1410-1530	AT_M2
Wednesday	1530-1700	AT_M2
Thursday	1610-1730	AT_M2
Friday	1110-1230	MH_LG.15
Friday	1230-1400	40GS_LG.07

Assessment

- No exam
- Three pieces of coursework worth 20%, 20% and 40%
  - The final piece of coursework is done during the exam period
- Continuous assessment in the form of weekly quizzes (20% in total)

Quizzes

Quizzes/surveys

- Weekly in weeks 1-10 plus getting started badge
- Helps you reflect on your learning

- Quizzes help you reflect on what you have learned each week and help us check on progress
- They also help us gauge how the course is going so we can respond appropriately
- They should not be difficult or time consuming - if you have covered the material, you should be able to do them.

Quizzes

Quizzes/surveys

- Weekly in weeks 1-10 plus getting started badge
- Helps you reflect on your learning

- Worth 2% if you get more than 80%
- Worth 1% if you get 50-79%
- Quizzes are not supposed to be difficult or time consuming - they are a check for you that you have understood the key points of the lecture
- Some questions in the quiz are not assessed - these are about gathering feedback
- Allow you to claim badges to track your progress

How to do well in this course

- Try to stick to the hours indicated for each week as much as you can
- Prioritise your mental and physical health
  - Don’t work very long hours*
  - Don’t give yourself a hard time if you sometimes get behind
  - Take time to exercise and sleep properly
- The most important thing to do is practice - do your lab exercises!
- Remember that doing well in this course is about laying the foundations for future learning - it’s not all about grades.

*it’s ok to put in extra hours around deadlines, but don’t do this as a regular thing, and take proper breaks after you’ve done this.

How do I know I’m doing well?

- Don’t depend on grades to tell you this.
  - Grades represent lots of things, like your past experience, the amount of support you’ve been getting, your general health, etc.
  - Good grades are a good thing, but bad grades don’t mean you’re not progressing
  - If you get bad grades, the important thing is to look into what went wrong and understand how to improve. Make sure you get support with this.
- If you are putting in the hours required and attending sessions then you are building up the skills you need. This won’t always be reflected in your grades.

How important are good grades?

- Getting good grades is nice, and we always encourage you to work towards these
- But lots of students who do very well in their degrees and careers don't do that well in first year. Why not?
  - Very mixed backgrounds - some have a lot more to learn than others
  - Many students take a while to adapt to uni studying
  - Sometimes you just have a bad year, for all sorts of reasons
- Focus on the process of learning - grades are just an indication.

Marking Criteria

- In your assessment, we will be looking for:
  - Completion
  - Readability and Code Structure
  - Correctness and Robustness
  - Use of the Java Language
  - Code Review

Marking Criteria

- Marks are assigned following the Universities Common Marking Scheme
  - <40% - **Fail**
  - 40 - 49% - **Pass**
  - 50 - 59% - **Good**
  - 60 - 69% - **Very Good**
  - 70 - 79% - **Excellent**
  - >80% - **Outstanding** (and exceptionally rare)

Understanding Grades

- “70 is the new 100”
- A grade of 70 means you have done everything expected of you
- Over 70 requires you to go beyond what is in the course
- In Inf1B this is common, as many students have a lot of prior knowledge
- If you are beginning in Java, you should be aiming for 60-70 as a great grade - getting more than that is rare and may lead to overworking.

Resit

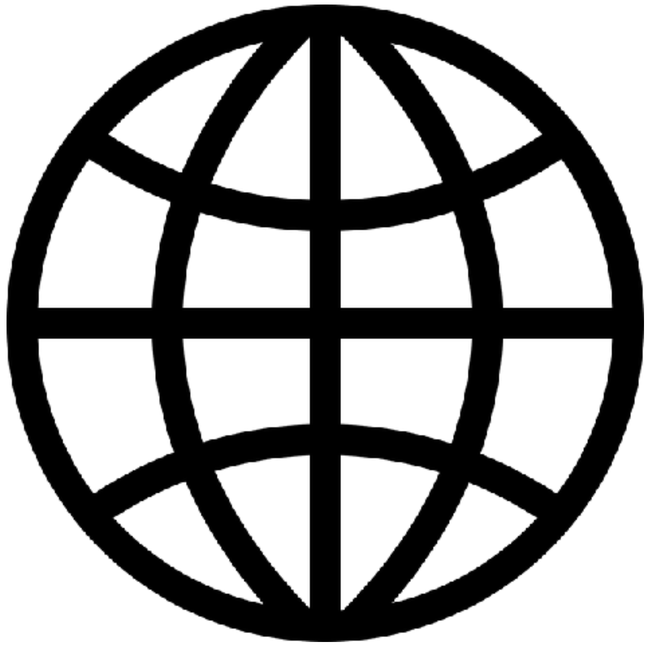
- Inf1B is a **Core Course**
  - If you fail this course and the resit, you will have to repeat year 1
- A summer resit will be offered, likely in the form of a take-home assignment

Good Scholarly Practice

Please remember the University requirement as regards all assessed work for credit. Details about this can be found at:

http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct

Resources



To get you started:

- Oracle Java tutorials
- Java Language Spec
- API Spec
- Tutorials Point
- Lynda
- Stackoverflow

but there are many many sources: feel free to browse and find what suits your own style

Who to contact for help?

- **Lecturer:** Fiona McNeill
- **Assessment Lead:** James Cheney and Adriana Sejfia
- **TA:** Colton Botta and Christopher Dalziel
- **Course Page**: **OpenCourse** & **Learn**
- **Piazza**: see Discussions link on Learn
- **Tutors and Demonstrators**
- **ITO: Kendal Reid** AT level 6; source of all admin knowledge

Who to contact for more help?

- **Fellow Students:** feel free to work in groups
- **InfPals:** student-to-student study groups (Link)
- **Societies:** CompSoc or Hoppers
- **Better Informatics:** https://betterinformatics.com