1st Year Course Content

Informatics Courses (40 credits)

- INF1a
  - Functional Programming
  - Computation and Logic  
- Object-oriented Programming  
- Data and Analysis  

Maths Courses (40 credits)

- Introduction to linear algebra  
- Calculus and its applications  

Outside Courses (40 credits)

- Chosen from anywhere in the University (timetable permitting)

See the handbook for details - look for the course web page
The basics of functional programming using the language Haskell

(Pure) functional programs have no state and no “side-effects”

- the output from a function depends only on the input
- this makes it much easier to predict the behaviour

The course introduces fundamental concepts such as recursion, abstraction, higher-order functions and data types

And emphasises the practical use of these constructs ...

Functional languages are becoming more popular in commercial applications

- Barclays, Deutsche Bank, Facebook, Google, Intel (Haskell)
- Netflix, Twitter, Foursquare, LinkedIn (Scala)

Don Sannella & Phil Wadler
Computation & Logic

Communication

- Our natural languages are often verbose and ambiguous. This course introduces some tools that we use to talk and reason, clearly and unambiguously, about computational systems. We will study two fundamental topics:

Finite State Systems

- Finite state machines provide a simple model of computation that is widely used. We will study their structure and behaviour. They are used as basic computational models across the whole of Informatics and at the same time underpin many widely used applications and components.

Propositional Logic

- Propositional logic is the first step in understanding logic. We use it to build computational tools for solving puzzles and reasoning about Informatics systems and their properties.

Michael Fourman
General principles of programming in imperative and object-oriented frameworks using the language Java

- learn to program in a major real-world programming language, Java
  - Many libraries and existing applications are written in Java

- Uses an imperative, object-oriented paradigm in contrast to Haskell
  - “Object-oriented” languages encapsulate the data, together with the code which manipulates them into “objects”

- This course tries to be very flexible to cater for very wide range of prior experiences, from no previous experience (or failing Inf1A…) to lots
  - to make that work, it's essential to go in ready to be in charge of your own learning and pick appropriate challenges
An introduction to ways of **thinking** about data from areas across Informatics: what it is, how much there is, how to represent it, process it, and analyse it to find out new things.

- **Structured Data**
  - The highly-structured records and files in relational databases of all sizes: how they fit together, and how to use SQL to extract the information within.

- **Semistructured Data**
  - Flexible and adaptive trees of self-describing data, using XML to represent them and XPath to process them.

- **Unstructured Data**
  - Massive text repositories, acres of numerical data from scientific experiments: ways to build up these giant datastores and ways to analyse their content.
Course Activities

Lectures

- Check the web pages for the times & dates
  - Some weeks there may be no lecture
  - Some lectures start on the hour, some at 10 past

- Check the web pages for any recommended textbooks and notes

Tutorials

- Tutorials are a valuable resource - these are compulsory
  - you will be allocated a group/time by the ITO
  - the tutor is there to facilitate your discussions and answer your questions
  - to take advantage of this, you must work on the exercises beforehand

Lab Sessions

- Demonstrators are available during lab sessions to help you

Exams

- All courses are assessed by an exam at the end of the semester
“Programming - and in general software development - is not a purely theoretical skill, nor is it something you can do well without learning some fundamental concepts. Unfortunately, far too often, teaching fails to maintain a balance between theory/principles and practicalities/techniques.

Consequently, we see people who basically despise programming (“mere coding”) and think that software can be developed from first principles without any practical skills.

Conversely, we see people who are convinced that “good code” is everything and can be achieved with little more than a quick look at an online manual and a lot of cutting and pasting; ...

My opinion is that both attitudes are far too extreme and lead to poorly structured, inefficient, and unmaintainable messes even when they do manage to produce minimally functioning code.”

Bjarne Stroustrup (designer of the C++ programming language)
All Informatics students are expected to be able to program well.

Programming is not just “code hacking”
- “real” programs need to be correct, reliable, maintainable & efficient
- they are often large & complex, involving many people
- this requires a solid understanding of the underlying principles and theory

Programming is a practical craft
- you can’t learn by just attending lectures & reading books
- you need to practice steadily outside of the formal course activities
- the programming club provides one opportunity
  - http://progclub.inf.ed.ac.uk/

Different students will have vastly different previous experience
- there will be a choice towards the end of the first semester …
  - an introduction to imperative programming in Java, for those with less or no previous experience
  - some more advanced functional programming, for those who already have sufficient experience
Computing facilities for Informatics 1 are available in the Computer Laboratories in Appleton Tower

- DICE: “Distributed Informatics Computing Environment”
  - Linux based. Provides all of the software that you need for your courses
  - please use the systems responsibly!

- You may use your own machines for your coursework
  - but DICE machines are used for the exams, so you must learn to use them

- There may be some introductory sessions
  - this covers “local” information, such as the location of printers etc.
Getting Help

- **Course/content questions**
  - talk to the lecturer, tutor in your tutorial, or lab demonstrator
  - InfBase provides additional support “when you need it”

- **Academic difficulties**
  - talk to your personal tutor

- **Personal difficulties**
  - talk to the SST (student support team)
    in person, or email to inf-sst@inf.ed.ac.uk

- **Problems with Informatics computing facilities or accounts**
  - talk to the computing support staff, or fill in the online support form

- **General enquiries**
  - talk to the ITO (Informatics teaching Office)
    in person (AT 6.05), or email ito@inf.ed.ac.uk
Class representatives are appointed to represent the views of students

- By conveying student opinion about the courses, or by alerting staff to student issues that they are unaware of, class reps can make a real difference to how Informatics 1 functions
  - gather feedback from students on all aspects of courses and facilities
  - alert staff to issues
  - eligible to be selected as a member of Board of Studies or Teaching Committee
  - attend weekly lunchtime meetings during semester

- Volunteer as a class rep! This is also a great opportunity for your CV
  - see http://tinyurl.com/inf1-reps for details
    - I will mail round for volunteers
    - you will need to provide a short blurb
    - all students will be able to vote
    - we will take diversity into account, as well as the vote results
    - please mail, or come and see me if you have any questions
My Recommendations

Take responsibility for your own learning.
› This may be different from the model that you are used to
› All of the activities are designed to help you

Take advantage of the tutorials & labs
› Turn up & be prepared
› Participate. ask questions. make the tutorial work for you

Don’t give up if it gets hard
› Some things just are hard & it takes a while - try again

If you are having difficulties, don’t wait until it is too late
› Discuss it with someone as soon as possible

Have fun!