# Languages and Computability Preliminaries

Adam Wyner

CS3518, Spring 2018

University of Aberdeen

# Times and Places

- <u>Lectures</u>
- Monday 14-15 New King's 14
- Tuesday 11-12 King's College T2

- <u>Tutorials/Practicals</u>
- Tuesday 13:00-15:00 Meston 311

# Staff

- Adam Wyner, Lecturer, course coordinator
- Kees van Deemter, Professor, lecturing
- Yuting Zhao, Post-doc, lecturing

# Aims of the course

- The main question of the course:
  - What problems can be solved on a computer?
    - which **problems** are **computable**?
- Different perspectives on computability exist. In most of these, **problems** are seen in terms of **formal languages**

# Course structure

1. Introductory material
2. Functional programming and infinite sets
3. Computability

# History of CS3518

- In previous years at Aberdeen, CS3518 started with an introduction to Formal Languages
- More maths was introduced into levels 1 and 2
  - CS1022 Computing Programming and Principles
  - CS2013 Mathematics for Computing Science

# History of CS3518

- CS3518 presupposes you understand what these level 1 and 2 courses cover, especially the basics of
  - Formal languages (problems in terms of languages; regular expressions; finite automata)
  - Formal logic (propositional logic, predicate logic)
  - Elementary set theory (union, intersection, power set, relations and functions as sets)
- We also assume you have some programming experience or exposure to programming languages (e.g. Java and Haskell)

# Course structure in more detail

1. Introductory material
   - Bijections, infinite sets, the Russell paradox
2. Functional programming and infinite sets
   - Lambda calculus
   - Haskell: recursion, types, list comprehension
3. Computability
   - Turing Machines, Halting problem, undecidability of predicate logic

# At the end of the course, you should ..

1. .. have a basic understanding of the reasons why some important problems are not computable/decidable (and what may nevertheless be done with them)

2. .. have an appreciation for the power of Functional Programming (Haskell), particularly for working with maths.

NB: Though the theory of computability bears similarities to the theory of Computational Complexity, the latter is not covered in this course

# MyAberdeen

- All course materials will be available on MyAberdeen

- A tour of what is there.