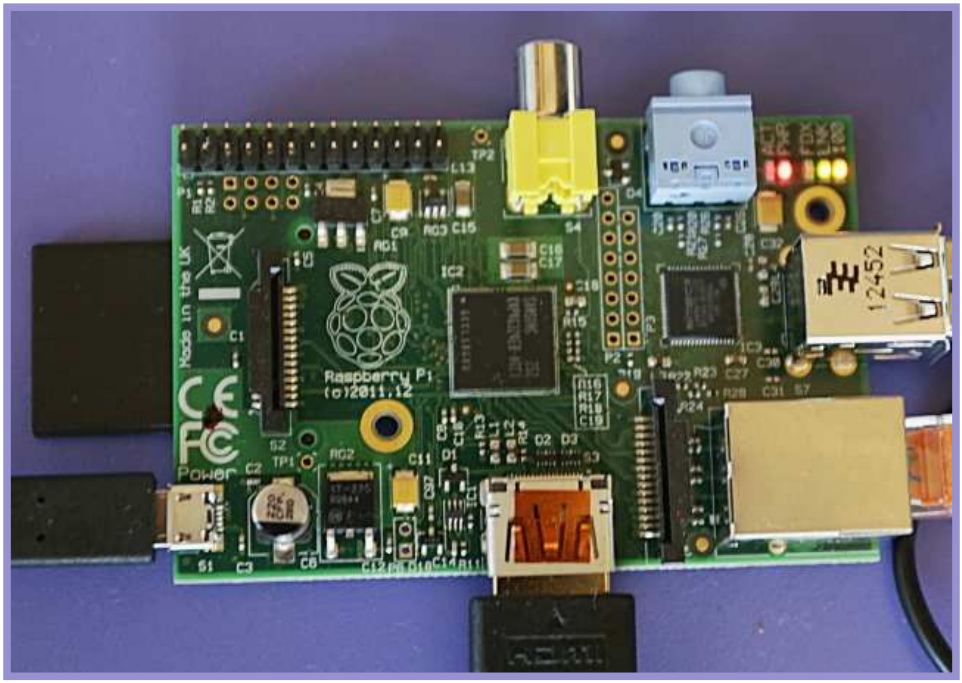
UNIVERSITY OF CAMBRIDGE
Computer Laboratory




# Example pathways for the Computer Science Tripos

### Programming languages

Ramanajan always enjoyed programming, and was interested in the wide variety of programming languages. In the first year he was introduced to functional programming and made a study of how and why this differs from procedural programming. The second year courses on Compiler Construction, Semantics and Comparative Languages thoroughly reinforced his understanding. For his third year project he wrote a compiler that targeted a novel, low-energy microprocessor developed in the Processor Architecture Group of the Computer Laboratory. He received many job offers from the local processor companies around Cambridge.

### Interaction design

Alex has always been fascinated with user interfaces and need to marry ease-of-use for beginners with access to advanced features and customisation by experienced users. The ways in which we interact with technology are always changing, from old-fashioned panels of buttons to voice, gesture and gaze operated systems. The first year Interaction Design course gave Alex the theoretical framework to understand what makes a good interface, which was developed in the second year Further HCI course. He took a major role in his second year group project, designing the web-based interface. In Part II Alex studied DSP and Music Production, which again involved placing advanced technology under human control. He stayed on for the fourth year, and greatly enjoyed the Affective Computing module.

### Databases

Harry wrote several social media apps during his sixth-form studies and one of them accrued many thousand interacting users. The simple data management system he had implemented would not scale up and the app was eclipsed by a rival. If only he had already formally studied some Computer Science! The first year courses on Databases and Real-World data soon showed him how he could have done things better. His second year group project required the development of a mobile app

that had similar data handling requirements, so his role was the design and development of the central data federating protocols. His influence on the group led to a peer-to-peer design being used from the outset, instead of the cloud-based implementation, envisaged by the others. In the third year, Harry learned most from the Bioinformatics course, where he found the algorithms taught to be applicable to all sorts of large data sets, such as social network graph traversal.

### Hardware and computer design

Nicole always loved electronics and robots. She learned to solder at the age of ten and had assembled many kits, before entering robots, based around the Raspberry Pi board, in the Pi Wars competition held each year at the Computer Laboratory. The first year Digital Electronics course gave her a thorough understanding of the hardware she had built. The second year ECAD and Architecture workshops taught her the Verilog RTL language leading her to re-implement her favourite collision avoidance algorithm in hardware on an FPGA, making her toy robot stop and turn much more quickly than when the algorithm was in software. She also greatly enjoyed the Logic and Proof and Model Checking courses because then she could prove that her robots would always stop safely under certain potentially dangerous conditions, even though the main body of control software was too complex to reason about. She went on to do a PhD in Reconfigurable Computing, where the low execution energy of FPGA-based software is increasingly leading to a greener planet.

### Theoretical computer science

At school, Suzanne's strongest subject was always Maths and she had already won a national competition. Reading Computer Science at Cambridge, she learnt so much new material from the Discrete Maths course, which seemed so different from school maths, yet equally as beautiful. The relationship between programming and discrete maths soon became apparent and she understood for herself the analogy between efficiency of an algorithm and conciseness of a proof. The first year Scientific Computing module and the second year Data Science

course taught her new forms of statistical analysis and inference checking. These would be invaluable in her future career as a government statistician. She also greatly enjoyed the second year courses on Computation Theory and Complexity Theory. The third year Quantum Computing course showed the importance of counting with the correct units; just as complex memory systems and multiprocessors have already altered the best approach in complexity terms, so too the quantum promises a revolution in computing.

## Business studies

Bartok's parents are both highly-successful engineering managers. On application, he intended to read only Part I of the Computer Science Tripos and then change to Management Studies. His parents believe he only needed a little in-depth technical knowledge and that technical work should be abandoned in favour of the management path at the appropriate point. But Bartok found that he loved Computer Science, while there were still opportunities to study management within the Computer Science Tripos. Ultimately he felt there was so much valuable Computer Science to learn at Cambridge that he should take an MBA later in life. In the first year, Bartok took most interest in the Software Engineering course, and he made a private study of software engineering tools, such as build systems, revision control systems, requirements capture packages and design methodologies. He took a minor leadership role in the first-year Interaction Design mini group project and was the designated project manager for his second-year group project. He found the second year course on Ethics and Professional Practice very interesting. During the summer break between his second and third years he worked at the Entrepreneurship Centre next to the Computer Laboratory and in the third year he enjoyed the Business Studies Seminars, exploring ideas with the guest speakers afterwards.