TRINITY COLLEGE DUBLIN

School of Computer Science and Statistics

BA Moderatorship in Computer Science
Master in Computer Science

Course Handbook
2014–15
Contents

1 A Note on this Handbook 6

2 Introduction 7

3 General Information 7
   3.1 Trinity College Dublin ............................................. 7
   3.2 The School of Computer Science and Statistics .................. 7
   3.3 School Contact Details ............................................ 8
   3.4 Academic and Administrative Staff ............................... 8
   3.5 Student Contact .................................................. 9

4 The Degree Programme 9
   4.1 Aims and Objectives ............................................. 9
   4.2 Structure of the Programme ..................................... 10
      4.2.1 European Credit Transfer System .......................... 11

5 Year 1 – Junior Freshman Year 12
   5.1 CS1003 Mathematics ............................................. 12
   5.2 CS1010 Introduction to Programming ............................ 13
   5.3 CS1013 Programming Project I .................................... 14
   5.4 CS1021 Introduction to Computing I ............................. 14
   5.5 CS1022 Introduction to Computing II ............................ 14
   5.6 CS1025 Electrotechnology ......................................... 15
   5.7 CS1026 Digital Logic Design ..................................... 15
   5.8 CS1031 Telecommunications I ..................................... 15
   5.9 CS1081 Computers and Society ................................... 16

6 Year 2 – Senior Freshman Year 16
   6.1 MA2C03 Discrete Mathematics .................................... 16
   6.2 CS2010 Programming Techniques .................................. 16
   6.3 CS2013 Programming Project II ................................... 17
   6.4 CS2014 Systems Programming ................................... 17
   6.5 CS2016 Concurrent Systems and Operating Systems .......... 17
   6.6 CS2021 Microprocessor Systems .................................. 18
   6.7 CS2022 Computer Architecture I ................................ 18
   6.8 CS2031 Telecommunications II ................................... 19
   6.9 CS2041 Information Management I ............................... 19

7 Year 3 – Junior Sophister Year 20
   7.1 CS3011 Symbolic Programming ................................... 20
   7.2 CS3012 Software Engineering ................................... 20
   7.3 CS3013 Software Engineering Group Project ................... 21
| 7.4 | CS3014 Concurrent Systems I ........................................... 21 |
| 7.5 | CS3016 Introduction to Functional Programming .................... 21 |
| 7.6 | CS3017 Introduction to the Semantics of Formal Languages ......... 22 |
| 7.7 | CS3021 Computer Architecture II ....................................... 22 |
| 7.8 | CS3031 Advanced Telecommunications ................................... 22 |
| 7.9 | CS3041 Information Management II ..................................... 22 |
| 7.10| CS3061 Artificial Intelligence I ....................................... 23 |
| 7.11| CS3071 Compiler Design I ............................................... 23 |
| 7.12| CS3081 Computational Mathematics .................................... 23 |
| 7.13| ST3009 Statistical Methods for Computer Science .................... 24 |
| 8   | Year 4 – Senior Sophister Year ........................................ 24 |
| 8.1 | Year 4 – Moderatorship Only ........................................... 24 |
| 8.2 | Year 4 – MCS Programme ................................................ 24 |
| 8.3 | CS4051 Human Factors .................................................... 25 |
| 8.4 | CS4081 Technology Entrepreneurship ................................... 25 |
| 8.5 | CS4098 Group Computer Science Project ................................ 25 |
| 8.6 | CS4099 Final Year Project .............................................. 25 |
| 8.7 | CS7091 Industrial / Research Lab Internship ........................ 26 |
| 9   | Year 5 (MCS) ............................................................... 26 |
| 9.1 | CS7064 Research Methods ............................................... 26 |
| 9.2 | CS7092 MCS Dissertation ............................................... 26 |
| 10  | Year 4 and Year 5 Options ............................................... 27 |
| 10.1| CS4001 Fuzzy Logic ..................................................... 28 |
| 10.2| CS4004 Formal Verification Techniques ................................ 28 |
| 10.3| CS4012 Topics in Functional Programming ............................. 28 |
| 10.4| CS4021 Advanced Computer Architecture .............................. 28 |
| 10.5| CS4031 Mobile Communications ........................................ 28 |
| 10.6| CS4032 Distributed Systems ............................................ 29 |
| 10.7| CS4052 Computer Graphics .............................................. 29 |
| 10.8| CS4053 Computer Vision ................................................ 30 |
| 10.9| CS4061 Artificial Intelligence IIA ...................................... 30 |
| 10.10|CS4071 Compiler Design II ............................................. 30 |
| 10.11|CS7003 Middleware for Distributed Systems ........................... 30 |
| 10.12|CS7004 Embedded Systems .............................................. 31 |
| 10.13|CS7008 Vision Systems .................................................. 31 |
| 10.14|CS7009 Networked Applications I ...................................... 31 |
| 10.15|CS7012 Management of Networks and Distributed Systems .......... 32 |
| 10.16|CS7030/CS7058 Numerical Methods and Advanced Mathematical Modelling I/II ........................................... 32 |
| 10.17|CS7031 Graphics and Console Hardware ................................ 32 |
| 10.18|CS7032 Artificial Intelligence .......................................... 32 |
10.19CS7033 Real-time Animation ........................................ 33
10.20CS7034 Augmented Reality ..................................... 33
10.21CS7048 Data Communications and Wireless Networking ........ 33
10.22CS7052 Sustainable Computing .................................. 33
10.23CS7053 Security of Networks and Distributed Systems ......... 33
10.24CS7055 Real-time Rendering .................................... 34
10.25CS7068 Financial Informatics ................................... 34
10.26CS7069 Behavioural Finance .................................... 34
10.27CS7073 Specification and Analysis of Reactive Systems ....... 34

11 Prizes ........................................................................... 35
11.1 Prizes in the Computer Science Programme ..................... 35
11.1.1 The Professor John G. Byrne Prize ............................ 36
11.1.2 The Victor W. Graham Prize .................................. 36
11.1.3 The Ludgate Prize ................................................ 36
11.1.4 The William Nurock Prize .................................... 36
11.2 Scholarship ............................................................... 36

12 Regulations .................................................................. 37
12.1 College Regulations .................................................. 37
12.2 Attendance and Participation ...................................... 37
12.3 Non-Satisfactory Attendance or Performance .................. 37
12.4 Retaining Coursework ............................................... 38
12.5 Plagiarism .................................................................. 38
12.5.1 Identifying Plagiarism ............................................ 39
12.6 Examinations and Assessment ..................................... 39
12.6.1 Progression .......................................................... 40
12.6.2 Progression to Year 4 of the Masters Programme ......... 41
12.6.3 Progression to Year 5 of the Masters Programme ......... 41
12.6.4 Repeating the Year ............................................... 41
12.6.5 Module Assessment .............................................. 42
12.7 Awards .................................................................... 42
12.7.1 Ordinary BA Degree (exit only) .............................. 42
12.7.2 Moderatorship Degree ........................................... 42
12.7.3 Master in Computer Science Degree ......................... 43

13 Academic Year Structure .......................................... 44

14 Year 1 Timetable – Michaelmas Term .............................. 45

15 Year 1 Timetable – Hilary Term .................................... 46

16 Year 2 Timetable ........................................................ 47
17 Year 3 Timetable 48
18 Year 4 Timetable 49
19 Year 5 Timetable 50
20 Campus Map 51
21 First Year in University 52
22 Research Ethics 53

23 Student Supports 54
23.1 Programming Centre 54
23.2 Student Learning Development 54
23.3 Maths Help Room 54
23.4 Academic Concerns: Sources of Assistance 54
23.5 Personal Concerns: Sources of Assistance 55
23.6 Tutors 55

24 Health and Safety 56

1 A Note on this Handbook

The Computer Science degree programme leads to a BA Moderatorship in Computer Science degree after four years and a Master in Computer Science (MCS) degree after five. This handbook contains information and regulations for all Computer Science degree programme students in the 2014/2015 academic year. It provides a guide to what is expected of you on this programme, and the academic and personal support available to you. Please retain it for future reference.

Information provided in this handbook is accurate at time of preparation, except where noted. Any necessary revisions will be notified by college email. Please note that, in the event of any conflict or inconsistency between the General Regulations published in the University Calendar and information contained in course handbooks, the provisions of the General Regulations will prevail. The University Calendar is available at http://www.tcd.ie/calendar/.

This handbook is available from the School of Computer Science and Statistics website. A hard copy of this document is available from the School Reception office on request.
2 Introduction

Welcome to the Computer Science degree programme offered by the School of Computer Science and Statistics.

We are confident that you will find this programme challenging and demanding, and we hope that you will find your studies at Trinity College Dublin both stimulating and rewarding. Our courses have been designed to offer students a dynamic, structured and coherent learning experience. Our programme has several features which we believe will contribute to your studies being an effective and enjoyable period of personal and academic development.

If you are an incoming student, we invite you to read First Year in University, on page 52, which may help you understand what you need to do to have an enjoyable and productive time at college.

We wish you every success in the coming year.
Mike Brady, Jeremy Jones.

3 General Information

3.1 Trinity College Dublin

Trinity College Dublin (TCD)—the College of the Holy and Undivided Trinity of Queen Elizabeth near Dublin—was founded in 1592 by Queen Elizabeth I. Trinity is sometimes referred to as the University of Dublin or Dublin University. Today, Trinity has nearly 800 academics catering for more than 16,500 students (of whom over 25% are postgraduates). TCD is recognized internationally as Ireland’s premier university and is ranked in the top 100 world universities and amongst the top 50 European universities.

3.2 The School of Computer Science and Statistics

The School of Computer Science and Statistics (SCSS) was formed in 2005 by the amalgamation of the Department of Computer Science and the Department of Statistics. The School has more than 60 academic staff and more then 200 full-time postgraduate students and support staff. It comprises five academic disciplines:

- **Computer Systems** undertakes systems research at the hardware/software interface and has a particular research focus on telecommunications and networked computer systems.

- **Information Systems** studies the impact of Information and Communication Technology on society and business with a particular focus on the impact of technology on learning.
• **Intelligent Systems** has a research focus is on computational issues related to perception, cognition, decision and interaction by and between systems and their human users.

• **Software Systems** has significant strength in programming language and software technologies and the formal foundations that underlie them.

• **Statistics** provides the School’s research strength in statistical learning techniques and in modelling uncertainty.

### 3.3 School Contact Details

The School Reception office is located beside Room G.8 in the O’Reilly Institute. Opening hours during lecture terms are 9:00 am to 11:00 am, 11:30 am to 1:00 pm and 2:00 pm to 5:00 pm.

- **Tel:** (01) 896 1765
- **Fax:** (01) 677 2204
- **Email:** enquiries@scss.tcd.ie
- **Web:** [http://www.scss.tcd.ie/](http://www.scss.tcd.ie/)

The postal address of the school is: **School of Computer Science and Statistics, O’Reilly Institute, Trinity College Dublin, Dublin 2.**

The noticeboard for this programme is located beside School Reception in the O’Reilly Institute.

### 3.4 Academic and Administrative Staff

- **Dr Jeremy Jones** Head of School
- **Dr Mike Brady** Director of Undergraduate Teaching and Learning
- **Dr Lucy Hederman** Director of Postgraduate Teaching and Learning
- **Ms. Kaukab Fatima Naqvi** Executive Officer, Teaching Support Unit

The school’s Director of Undergraduate Teaching & Learning is Dr Mike Brady who can be contacted by telephone at (01) 896 1786, or by email at brady@scss.tcd.ie.

Please note that in the first instance all enquiries regarding modules, assignments, feedback and supervision should be directed to the administrative staff in the Teaching Support Unit who will then, where appropriate, inform the director and
coordinator. The Teaching Support Unit can be contacted by emailing undergraduate@scss.tcd.ie.

3.5 Student Contact

Communications from many College services will be sent to you via your online portal at https://my.tcd.ie, which will give you access to an ‘in tray’ of your messages. You can view your timetables online, both for your teaching and for your examinations. Fee invoices and payments, student levies and commencement fees will be issued online and all payments will be carried out online. You can view your personal details in the new system—some sections of which you may edit yourself.

The ‘TCD Blackboard’ online learning system is accessible via http://mymodule.tcd.ie.

Examination results are published on the noticeboard in the O’Reilly Institute and online by the Examinations Office via https://my.tcd.ie.

Lecturers, tutors and support staff may contact you using your College email address, which you can access through http://myzone.tcd.ie.

It is expected that you will check your College email regularly. The use of other email addresses for official communication is discouraged.

4 The Degree Programme

The Computer Science degree programme began in 1979 as a four-year course leading to an honors degree in Computer Science. Honors\(^1\) degrees are traditionally called Moderatorships in Trinity, thus the formal title of the degree is B.A. (Mod.) in Computer Science.

Since 2011, the programme also offers the degree of Master in Computer Science over a five-year course.

4.1 Aims and Objectives

The aim of the programme is to bring you to a thorough understanding of the principles and practice of computer science and to prepare you to contribute fully and effectively as a member of the professional computer science community, in industry or the academic world.

Developing this into a set of objectives, on successful completion of the programme, you should be able to:

---

\(^1\)This is the correct spelling of the word when applied to degrees awarded in TCD.
• Develop and apply computer systems from a broad base of knowledge in mathematics, computer science, computer technology and human factors.

• Identify and formulate advanced technical challenges and demonstrate judgement to design appropriate computer science solutions.

• Design systems, components or processes to meet specified functional objectives and to measure and analyse performance against these objectives.

• Understand and express the role of computer science in the community including the need for high standards of ethical behaviour and professional responsibility.

• Work effectively, independently and within multidisciplinary teams, and act as a mentor in team settings and engage in lifelong learning.

• Communicate effectively both professionally with other computing professionals and with the wider community.

• Participate in contemporary research activity as appropriate and demonstrate the knowledge and skills needed to undertake independent research.

4.2 Structure of the Programme

Figure 1 (p.11) shows the five-year composition of the programme. Students typically enter Year 1 from secondary school via the Central Applications Office (CAO) system. Suitably qualified students may also join the programme in Year 3 or Year 4.

Students normally exit the programme at the end of Year 4, with a B.A. (Mod.) in Computer Science degree, or at the end of Year 5, with a Master in Computer Science degree in addition to the B.A. (Mod.) in Computer Science degree. Provision is made for students to exit the programme with an ordinary B.A. degree at the end of Year 3.

The terms Junior Freshman, Senior Freshman, Junior Sophister and Senior Sophister are widely used in Trinity to refer to a first-year, second-year, third-year and fourth-year student respectively; thus, for example, Junior Freshman year, (or JF year), refers to Year 1.

In the first two years, instruction is given in the theoretical underpinnings of computer science along with courses in hardware and software. In the later years, students may select a number of options in addition to core courses. Students participating in the MCS course are required to engage in a one-semester internship in industry or in a university research laboratory in Year 4. In Year 5, students undertake a significant project with a substantial element of independent research leading to a dissertation. There are laboratory classes in each year of the programme.
The teaching year is divided into two twelve-week semesters. The first semester is the Michaelmas Term, the second is the Hilary Term\(^\text{2}\). The seventh week of each semester is a reading week, during which no lectures are held.

Subjects are taught in modules. Modules may be taught for one or two semesters, and consist of lectures, tutorials, seminars, and laboratory sessions. All students on the programme take the same modules in Year 1, Year 2 and the first semester of Year 3. From the second half of Year 3 onwards, students take some compulsory modules and a selection of elective modules. Each module is assigned an European Credit Transfer System (ECTS) rating. Modules in the first three years are each assigned five or ten ECTS points. Modules and other parts of the programme in Year 4 and Year 5 may have a rating of more than five ECTS points each.

4.2.1 European Credit Transfer System

The European Credit Transfer System (ECTS) is an academic credit transfer and accumulation system representing the student workload required to achieve the specified objectives of a study programme.

The ECTS weighting for a module is a measure of the student input or workload

\(^{2}\)The names come from traditional Christian feast days: Michaelmas Day is September 29 and St Hilary’s Day is January 13.
required for that module, based on factors such as the number of contact hours, the
number and length of written or verbally presented assessment exercises, class
preparation and private study time, laboratory classes, examinations, clinical
attendance, professional training placements, and so on as appropriate. There is no
intrinsic relationship between the credit volume of a module and its level of difficulty.

In College, one ECTS unit is defined as 20–25 hours of student input so a
five-credit module will be designed to require 100–125 hours of student input
including class contact time and independent or group work. Each year of the
programme is composed of modules worth a total of 60 credits. Where there is the
option to choose from a range of modules, it is the responsibility of the student to
ensure that they successfully complete modules worth 60 credits.

ECTS credits are awarded to a student only upon successful completion of the
course year. Progression from one year to the next is determined by the course
regulations. Students who fail a year of their course will not obtain credit for that
year even if they have passed certain component modules. Exceptions to this rule are
one-year and part-year visiting students, who are awarded credit for individual
modules successfully completed.

5 Year 1 – Junior Freshman Year

Year 1 is called Junior Freshman (JF) year in Trinity. In this section is a list of Year 1
subject modules and a brief description of each. Full details, including learning
outcomes, book recommendations and important evaluation and assessment criteria
are available via http://my.tcd.ie.

<table>
<thead>
<tr>
<th>Michaelmas Term</th>
<th>Hilary Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS1003 Mathematics</td>
<td>CS1010 Introduction to Programming</td>
</tr>
<tr>
<td>CS1021 Introduction to Computing I</td>
<td>CS1022 Introduction to Computing II</td>
</tr>
<tr>
<td>CS1026 Digital Logic Design</td>
<td></td>
</tr>
<tr>
<td>CS1025 Electrotechnology</td>
<td>CS1031 Telecommunications I</td>
</tr>
<tr>
<td>CS1081 Computers and Society</td>
<td>CS1013 Programming Project I</td>
</tr>
</tbody>
</table>

5.1 CS1003 Mathematics

This module\(^3\) aims to provide students with an introduction to the mathematics,
both continuous and discrete, which lies at the foundation of many real-world
applications in Computer Science, Engineering and the Social Sciences.

\(^3\)This module is coded CS1000 in the College Calendar
Mathematics is of interest to computer scientists due to the fact that it is both practical and theoretical in nature. Not only does it have a myriad of applications (e.g. in wireless communications and computer graphics), it is also of intrinsic interest to theoretical computer scientists. The mathematical techniques learned as part of this module have wider applications in areas as diverse as Business (e.g. for modelling volatility and risk), Economics and Engineering (e.g. for structural monitoring).

This module aims to develop the students skills and abilities in the mathematical methods necessary for solving practical problems. In the first semester students will encounter some of the key mathematical structures at the heart of computer science including the representation of data using matrices. They will gain a greater appreciation of the relationships between calculus and the graphs of functions, including the representation of functions using Taylor Series. During Semester 2 students will be introduced to discrete mathematics and mathematical logic along with their applications to computer science. In particular, the module will introduce set operations, discrete maths functions in Number Theory and Logic calculation. This part of the module is influenced by the approaches of Backhouse, Dijkstra and Gries.

One of the key objectives for this module is to introduce students to the learning styles needed for university level mathematics. Students will be encouraged to develop the independent, reflective learning skills needed for success at University level. It is expected that students will adapt their learning style to become more independent, self-motivated learners.

5.2 CS1010 Introduction to Programming

This module provides an introductory course in computer programming. This course takes a practical approach to teaching the fundamental concepts of computer programming with a strong emphasis on tutorial and laboratory work and is an important vehicle for developing students’ analytical and problem-solving skills.

This module aims to give students an understanding of how computers can be employed to solve real-world problems. Specifically, this course introduces students to the object-oriented approach to program design and teaches them how to write programs in an object-oriented language (in this case Java).

Students also have the opportunity to reinforce their problem solving and programming skills by developing solutions to programming problems and implementing those solutions as object-based programs.
5.3 CS1013 Programming Project I

This module concentrates on the development of practical programming ability through example-based lecturing coupled with intensive laboratory sessions. The emphasis throughout is on producing working programs, starting with interactive graphical applications and moving on to construction of a larger group project involving a data visualisation task.

5.4 CS1021 Introduction to Computing I

This module provides students with an introduction to the basic structure and operation of a computer system, focussing on the processor (CPU) and memory. Students gain an insight into the execution of programs on a computer system by designing, implementing and executing simple assembly language programs. Students are also introduced to concepts that are fundamental to the study of Computer Science, including the binary numeral system and the representation of basic information such as signed integers and strings (text).

Students are encouraged to consider the relationship between high-level programming language constructs from simple assignments and arithmetic expressions to conditional (if) and iterative (while) execution and the realisation of these constructs as sequences of machine instructions.

Students are also be given opportunities to develop their problem solving, programming and written communication skills by designing solutions to programming problems, implementing those solutions, first in the form of high-level pseudo-code programs and then as assembly language programs, which must be documented and tested.

5.5 CS1022 Introduction to Computing II

This module continues directly from CS1021 (which is a prerequisite) and examines the structure and behaviour of computer systems in greater depth. In particular, this module introduces students to the implementation of simple data structures (stacks, multi-dimensional arrays, composite data types), subroutines, exceptions, interrupts and basic I/O at the machine level.

The relationship between high-level programming language constructs and their realisation as executed instructions is explored further by developing pseudo-code solutions to programming problems in the first instance, before translating those solutions into assembly language programs. Particular attention is paid to the implementation of subroutines, the system stack and parameter passing conventions.

Students are given opportunities throughout the module to reinforce their problem solving, programming and written communication skills by designing, implementing,
documenting and testing solutions to programming problems of increasing complexity. Problem decomposition is strongly encouraged.

5.6 **CS1025 Electrotechnology**

The Electrotechnology course is intended as an introductory course. The course examines the structure of matter from an electrical perspective dealing with conductors, dielectrics and semiconductors. Concepts of electrical charge, Electric and Magnetic fields are also examined.

With this material, concepts of voltage, current, impedance in electrical circuits are explored. Circuit elements such as the resistor, capacitor, inductor, the semiconductor diode and transistors are examined as well as ideal and practical ac and dc supplies.

Methods of analyzing electrical circuits consisting of the above components are examined.

In the lab sessions students are expected to build electronic circuits and explain their operation.

5.7 **CS1026 Digital Logic Design**

Starting with the theoretical foundations of logic, students learn about combinatorial logic and how it can be used to construct logic functions that are useful in computing systems. They learn that feedback around combinatorial logic introduces asynchronous sequential behaviour that is the basis for latches and gated latches.

In the second semester, beginning with the study of edge-triggered flip-flops, students learn about synchronous sequential logic and algorithmic state machines.

The focus is on laying the groundwork for the hardware courses in the second year. Care is taken that the students realize the subject applies to both computer software and hardware. Laboratory experiments reinforce the concepts as well as adding variety and introducing practical elements.

5.8 **CS1031 Telecommunications I**

The Telecommunications course is intended as an introductory course. It begins with an overview of the concept of information and its significance in today's society. The aim of the course is to then shows how such information is transmitted over links and networks.

The course gives particular focus on the understanding of time domain, frequency domain and bandwidth of signals. It introduces some basic operations, such as modulation and multiplexing, and will introduce some general concept of data transmission and transmission media.
Course work will consist on laboratories that will put in practice most of the theory explained during the lectures.

5.9 CS1081 Computers and Society

Topics in this module include the following: IT and its “impact” on society; models for assessing technological “impact”; history of IT; ethics; writing, presenting and argumentation; other topics.

6 Year 2 – Senior Freshman Year

Year 2 is called Senior Freshman (SF) year in Trinity. In this section is a list of Year 2 subject modules and a brief description of each. Full details, including learning outcomes, book recommendations and important evaluation and assessment criteria are available via http://my.tcd.ie.

<table>
<thead>
<tr>
<th>Michaelmas Term</th>
<th>Hilary Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS2010 Algorithms &amp; Data Structures</td>
<td>CS2016 Concurrent Systems and Operating Systems</td>
</tr>
<tr>
<td>CS2014 Systems Programming</td>
<td>CS2021 Microprocessor Systems</td>
</tr>
<tr>
<td>CS2031 Telecommunications II</td>
<td>CS2022 Computer Architecture I</td>
</tr>
<tr>
<td>CS2041 Information Management I</td>
<td>MA2C03 Discrete Mathematics</td>
</tr>
<tr>
<td>CS2081 Broad Curriculum—Michaelmas Term or Hilary Term*</td>
<td>CS2013 Programming Project II</td>
</tr>
</tbody>
</table>

All modules have an ECTS weighting of 5 points.

*See http://www.tcd.ie/Broad_Curriculum

6.1 MA2C03 Discrete Mathematics

This module provides students with an introduction to Discrete Mathematics.

Students are exposed to diverse course material presented in the formal style and language that is commonplace in contemporary mathematics, with the aim that they should develop the skills required to engage effectively with such material.

Students get an introduction to a variety of topics arising from Calculus, Geometry and Discrete Mathematics, that are of relevance in fields such as acoustics, image processing, computer graphics and cryptology.

6.2 CS2010 Programming Techniques

The aim of the module is threefold: (1) to teach effective programming and problem solving, using a core toolset of classical algorithms and data structures, (2) to
introduce the methods for evaluating the performance and requirements of programs written by the students, and (3) to promote effective software engineering by using well-established techniques for code modularity, structuring, debugging and readability, such as Design by Contract, and unit testing.

6.3 CS2013 Programming Project II

In this module, students are introduced to the discipline of software engineering and must work in groups to complete a complex software project. Groups will be managed in conjunction with more senior students taking the CS3013 module. For the duration of this module, students are divided into groups, each of which is closely supervised by the module lecturer and a separate project customer. The module provides students with their first formal experience of group work. They are required to follow a rigorous process consisting initially of requirements gathering, analysis and system design. Thereafter they are required to implement a complex software product using industry standard software engineering tools and methodologies.

The principal aim of this module is to provide students with experience of working together in groups to complete a complex software project. Upon completion of the module students will have gained experience of analysing, specifying, designing and implementing a complete software system. They will also have been exposed to the challenges posed by working in teams and the need to communicate effectively both within their respective groups and to their project supervisor.

6.4 CS2014 Systems Programming

Students taking this module have already successfully completed modules in object-oriented Java programming and ARM assembly language programming. This module starts with a new perspective on program construction, with structured programming in C. This part of the module deals with the features of C, sound design principles for structured programming, and the necessary self-discipline required to program in a low-level language like C.

Students also learn the basics of how programs are commonly implemented, with special emphasis on the layout of program data in memory. In parallel with learning C programming, the students learn about the UNIX operating system, including program development tools, interaction between C programs and the operating system.

6.5 CS2016 Concurrent Systems and Operating Systems

The first part of this module introduces students to concurrency and concurrent programming. The aim is to provide students with the ability to develop concurrent.
software systems using standard techniques and constructs.

To achieve this aim, students must have a thorough understanding of common problems that arise in concurrent systems and how those problems can be avoided. This module will teach the use of tools and techniques for modelling and verifying the correctness of concurrent systems, applying this through practical laboratory exercises in which small concurrent software systems are developed.

The second part of the module addresses various aspects of the design of modern operating systems. The main aim is to explore how programmers can apply a knowledge of operating system features to the design of efficient applications. This is achieved by examining common algorithms and policies used by modern operating systems, as well as the facilities provided to application programmers. This knowledge is then applied in laboratory exercises.

### 6.6 CS2021 Microprocessor Systems

Students bring the knowledge and expertise of programming, digital logic and a small amount of digital electronics to the development of small system integration projects combining program design and implementation with simple interface interfacing and breadboarding.

Each team is provided with a small ARM-based computer, a PC-based integrated development system, some electronic components and breadboarding facilities.

Students have to design, build and test integrated hardware and software systems to provide a certain required functionality.

Students also study processor and memory architectural techniques such as pipelining, superscalar architectures and caches.

### 6.7 CS2022 Computer Architecture I

The aims of the module are to learn register-transfer specification and design and learn the fundamentals of an instruction processor.

The lectures and tutorials examine the detailed design and organisation of an instruction processor. Coursework consists of two design projects and students use VHDL and ModelSim to simulate and test their designs. The first project involves the design of a processor unit (ALU + shifter + fast registers) and the second project involves the design of an instruction processor.

Topics studied include digital logic, register transfer definitions, micro-operations, bus transfers, ALU design, shifter design, hardwired control design, microprogrammed processor control and the design of an instruction processor.

18
6.8 CS2031 Telecommunications II

The module is structured following the Open Systems Interconnect (OSI) model and consists of two parts.

The first part of the module focuses on the concepts and mechanisms that are employed in the second layer of the OSI model, the data link layer. This layer is concerned with the delivery of data between two immediately connected devices i.e. devices that share a common physical medium. The layer coordinates the access to the physical medium and attempts to detect and correct errors introduced by the transfer of signals over the physical medium. The concepts that are employed in this layer are discussed and the students exercises that demonstrate the application of these concepts.

The second part of the course focuses on the third and fourth layer of the OSI model. The third layer – the network layer – focuses on the connection of local area networks (LANs). This layer employs concepts that hide the communication through a LANs and provide an abstraction that allows the communication across various interconnected LANs. This abstraction forms the foundation for todays internet and represents essential knowledge for todays computer science graduates. The fourth layer – the transport layer – provides services such as reliable transport to applications. The understanding of the mechanisms employed in this layer is essential to the understanding of the implementations and performance of current network technology.

6.9 CS2041 Information Management I

This module focuses on the methods and techniques for efficient management (storage, manipulation and retrieval) of data and information in a computer and on the world wide web. It provides a foundation for later modules in database management and advanced information management.

The module explores fundamental issues related to information management in a system: how information and its exchange is modeled through UML; how data is organised on storage devices, transferred between storage and computer, structured within files, and effectively searched through indexing. Concepts are exercised through the examination of XML as an example file format used to both store and process information. Accompanying technologies for structuring, manipulating and querying XML will be studied and practiced.

The module also explores information on the web, ranging from traditional information retrieval techniques through to emerging semantic web techniques.
7 Year 3 – Junior Sophister Year

Year 3 is called Junior Sophister (JS) year in Trinity. In this section is a list of Year 3 subject modules and a brief description of each. In the Hilary semester, students take two of the three optional modules. Full details, including learning outcomes, book recommendations and important evaluation and assessment criteria are available via http://my.tcd.ie.

<table>
<thead>
<tr>
<th>Michaelmas Term</th>
<th>Hilary Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS3011 Symbolic Programming</td>
<td>CS3081 Computational Mathematics</td>
</tr>
<tr>
<td>CS3012 Software Engineering</td>
<td>CS3013 Software Engineering Group Project</td>
</tr>
<tr>
<td>CS3021 Computer Architecture II</td>
<td>CS3014 Concurrent Systems I</td>
</tr>
<tr>
<td>CS3071 Compiler Design I</td>
<td>ST3009 Statistical Methods for Computer Science</td>
</tr>
<tr>
<td>CS3016 Introduction to Functional Programming</td>
<td>CS3017 Introduction to the Semantics of Formal Languages*</td>
</tr>
<tr>
<td>CS3041 Information Management II</td>
<td>CS3031 Advanced Telecommunications*</td>
</tr>
<tr>
<td></td>
<td>CS3061 Artificial Intelligence*</td>
</tr>
</tbody>
</table>

*denotes an elective module; two elective modules must be taken.
All modules have an ECTS weighting of 5 points.

7.1 CS3011 Symbolic Programming

This module introduces Prolog, a symbolic programming language, including recursion, Definite Clause Grammars, cuts and negation.

7.2 CS3012 Software Engineering

This module provides students with a solid grounding in various aspects related to building large, important software systems.

The overall aim of this module is for students to learn the fundamental skills for building large, important software systems. This entails (i) recognising the general software lifecycle and its stages from domain analysis to maintenance, (ii) analysing software in the problem domain, (iii) identifying the fundamental approaches to managing software projects and teams, (iv) distinguishing the roles of stakeholders in a software project in general and in software teams in particular, (v) recognising architectures for building large-scale distributed software systems.

The module covers various aspects related to building software systems ranging from the use of software lifecycle models, to project management, to large-scale software architectures. Specifically, software lifecycle models, including variations of the waterfall and spiral models as well as extreme programming and agile software
development methods, are introduced along with concepts relevant to the specific model stages. These concepts include UML-based O-O, domain analysis, requirements and specification analysis, testing and debugging and version control. Strategies for managing large software projects and their contracts as well as project teams are presented and contrasted.

7.3 CS3013 Software Engineering Group Project

This module follows on directly from CS3012 (Software Engineering) and focuses on the practical application of the various concepts and tools related to building software systems studied in third year software engineering. This practical application takes the form of a large “hands-on” group project that covers numerous aspects of building object-oriented software systems including problem analysis, usage of development environments, project management, team management, design, implementation, testing and documentation. Students will take a leadership role within these groups which are combined with students taking module CS2013.

7.4 CS3014 Concurrent Systems I

The goal of this module is to provide students with a deep understanding of parallel and multi-core architectures and to provide students with necessary architecture background for careers in professional software development and/or further research on these emerging platforms.

7.5 CS3016 Introduction to Functional Programming

Functional programming languages present a powerful, abstract, and important direction in programming languages. The high level of abstraction and the expressive syntax makes program decomposition and composition unusually easy, while the close connections to the underlying semantics make formal reasoning tractable. Systems such as Google’s “Map/Reduce” framework demonstrate the influence of this approach, and the importance to a computer scientist of understanding it.

In this module students will learn to apply the techniques of functional programming in a practical context. The focus is on software design and programming in the functional style, and students will “learn by doing”, through regular weekly programming assignments and case studies.

The module draws on the programming and mathematics background the students have acquired in the first two years of the degree and extends it by teaching new approaches to program design and implementation.
7.6 CS3017 Introduction to the Semantics of Formal Languages

This module introduces students to formal methods for specifying the semantics of programming languages and formal techniques for verifying the behaviour of programs.

7.7 CS3021 Computer Architecture II

This module focuses on the architecture of modern high performance microprocessor systems. Topics covered are basic IA32 and x64 assembly language, procedure calling conventions, the architecture of RISC CPUs, instruction level pipelining, techniques to overcome data, load and control hazards, virtual memory, caches, multiprocessors and cache coherency.

This module pays particular attention to issues that improve performance and the close relationship between the hardware and the needs of the software.

7.8 CS3031 Advanced Telecommunications

This option concentrates on building upon the students JF and SF years knowledge and introduces them to advanced topics in the areas of data communications and telecoms networks. In particular the area of high speed local area networks operating at speeds of 100 megabits and higher are examined.

A more detailed study is made of the TCP/IP protocols in terms of addressing, routing and subnetting. Topics in the area of telecommunications protocols such as ATM, MPLS, GSM/UMTS are studied.

Students are also introduced to the areas of Network Security, Mobile Communications and Multimedia Networking.

7.9 CS3041 Information Management II

This module is focused on the modelling of information and database system technology. More specifically, it focuses on state-of-the-art database technology, from both the user and system perspectives.

From a system engineering perspective, the module examines the concepts and algorithms for: transaction processing, concurrency control, metadata representation, semantic representation and active databases, recovery, database security policies, integration of databases on the web and emerging database technologies.

From an information designer’s perspective, the module examines the theoretical model underpinning relational databases, functional dependency theory and
normalisation (for information modelling), functional dependency modelling, object relational modelling, implementation of databases and database applications.

Thus the module is intended to enable the students to design information models and implement these models in state of the art databases (relational and native web databases), as well as be able to analyse and evaluate approaches to information organisation, storage, transaction support and management.

7.10 CS3061 Artificial Intelligence I

This module introduces classic topics in AI, including search, constraint satisfaction, knowledge representation, abduction and meta-interpretation.

7.11 CS3071 Compiler Design I

The aim of the module is to teach the principles of compiler design, including finite-state automata and push-down machines, lexical analysers, top-down parsers, L-attributed translation grammars, and recursive-descent parsers.

Students will learn how to use Coco/R to construct high-level language analysers and parsers.

After a brief introduction to the subject, the first few classes will be devoted to the study of finite-state automata and their use in the design of efficient lexical analysers based on reduced (minimal) machines. This will be followed by an in-depth study of syntax analysis (with an emphasis on top-down parsing methods) based primarily on principles of attributed translation. The final couple of weeks will be devoted to the study of basic code generation techniques along with a selection of associated run-time implementation issues.

Tutorial exercises will be incorporated into lecture periods throughout the course, and students will be introduced to the more practical side of compiler construction during the weekly laboratory sessions.

7.12 CS3081 Computational Mathematics

The aim of the module is to teach, in sufficient detail for practical implementation, the mathematical concepts and methods appropriate to writing computer programs for science and engineering applications in general, and in particular: computer graphics, computer vision, image processing, robotics, physical simulation, and control.
7.13 ST3009 Statistical Methods for Computer Science

The aim of this new module is to introduce computer students to basic statistical concepts. More details TBA.

8 Year 4 – Senior Sophister Year

Year 4 is called Senior Sophister (SS) year in Trinity. In this section is a list of Year 4 subject modules and a brief description of each. Full details, including learning outcomes, book recommendations and important evaluation and assessment criteria are available via http://my.tcd.ie.

8.1 Year 4 – Moderatorship Only

Year 4 students who have elected to follow the BA (Mod) programme with the intention of graduating with a BA (Mod) in Computer Science after successfully completing four years of study must take the following modules:

<table>
<thead>
<tr>
<th>Michaelmas Term</th>
<th>Hilary Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS4051 Human Factors (5 credits)</td>
<td>CS4098 Group Computer Science Project (10 credits)</td>
</tr>
<tr>
<td>CS4081 Technology Entrepreneurship (5 credits)</td>
<td>CS4099 Final Year Project (20 credits)</td>
</tr>
</tbody>
</table>

In addition, students must select four options from the Year 4 Options in the Options Table in section 10. Full details are linked to the Year 4 web pages at http://www.scss.tcd.ie/undergraduate/computer-science/ss/.

8.2 Year 4 – MCS Programme

Year 4 students who have elected to follow the MCS programme with the intention of graduating with a Master in Computer Science after successfully completing five years of study must take the following modules:

<table>
<thead>
<tr>
<th>Michaelmas Term</th>
<th>Hilary Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS4051 Human Factors (5 credits)</td>
<td>CS7091 Industrial / Research Lab Internship (30 credits)</td>
</tr>
<tr>
<td>CS4081 Technology Entrepreneurship (5 credits)</td>
<td></td>
</tr>
</tbody>
</table>

In addition, student must select four options from the Year 4 Options in the Options Table in section 10. Full details are linked to the Year 4 web pages at http://www.scss.tcd.ie/undergraduate/computer-science/ss/mcs.php.
8.3 CS4051 Human Factors
This module aims to provide students with an understanding of the main issues underlying the usability of systems, and the main techniques and processes for interface design and evaluation. They will also gain a basic understanding of the theories which account for human performance.

8.4 CS4081 Technology Entrepreneurship
This module introduces the fundamentals of technology entrepreneurship. It will cover the process technology entrepreneurs use to start companies. This involves taking a technology idea and finding a high-potential commercial opportunity, gathering resources such as talent and capital, figuring out how to sell and market the idea and managing rapid growth.

8.5 CS4098 Group Computer Science Project
The Group Computer Science Project is undertaken in Hilary Term by students who have elected to follow the BA (Mod) programme. The goal of this project is to promote teamwork and also to allow students to use their individual skills and experiences within the context of developing defined projects from specification to delivery. A key aim is to develop students abilities in framing design problems and working iteratively to achieve a working solution.

Instruction will be provided in Agile development methodologies and facilities will be provided in order to promote close collaboration between team members.

A project supervisor will meet with each team weekly and will advise on setting up the team structure including the assignment of roles and responsibilities within the team and on reporting systems both internally and externally.

Weekly “peer” code and design reviews are a core component of the delivery of the module. These are to encourage a team approach to leaning and introduce the practicalities of software quality control.

8.6 CS4099 Final Year Project
The Final Year Project is undertaken by students who have elected to follow the BA (Mod) programme. The aim of the project is to integrate the theoretical and practical knowledge of the student across all of the years of their study and provide a practical demonstration of their capability in executing a challenging project.

Students may either select a project from the list of project proposals put forward by the lecturing staff, or alternatively propose their own projects (in consultation with the course director).
8.7 CS7091 Industrial / Research Lab Internship

The Internship is undertaken in the Hilary term by students who have elected to follow the MCS programme. The aim is to enable students to further develop an understanding of how design aspects and theoretical aspects of computer science are applied to practical problems in a real world context.

For more information, please visit http://www.scss.tcd.ie/internships.

9 Year 5 (MCS)

In this section is a list of Year 5 subject modules and a brief description of each. Full details, including learning outcomes, book recommendations and important evaluation and assessment criteria are available via http://my.tcd.ie.

Students must take the following modules:

<table>
<thead>
<tr>
<th>Michaelmas Term</th>
<th>Hilary Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS7064 Research Methods (5 credits)</td>
<td>CS7092 MCS Dissertation (30 credits)</td>
</tr>
</tbody>
</table>

In addition, student must select five options from the Year 5 Options in the Options Table in section 10.

9.1 CS7064 Research Methods

The module is designed to foster constructive reflection on methods of scientific research in computer science and statistics as appropriate to a five year integrated masters course in computer science.

9.2 CS7092 MCS Dissertation

Students will select and carry out an in-depth research project which is expected to yield publishable results. Students must select the project, carry out required investigations and submit their dissertation within the academic year.
10 Year 4 and Year 5 Options

In this section is a list of optional modules that may be taken by students in Years 4 and 5, as set out in sections 8 and 9. A brief description of each module is also provided.

<table>
<thead>
<tr>
<th>Year 4 Options</th>
<th>Year 5 Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS4001 Fuzzy Logic</td>
<td>CS7003 Middleware for Distributed Systems</td>
</tr>
<tr>
<td>CS4004 Formal Verification Techniques</td>
<td>CS7004 Embedded Systems</td>
</tr>
<tr>
<td>CS4012 Topics in Functional Programming</td>
<td>CS7008 Vision Systems</td>
</tr>
<tr>
<td>CS4021 Advanced Computer Architecture</td>
<td>CS7009 Networked Applications I</td>
</tr>
<tr>
<td>CS4031 Mobile Communications</td>
<td>CS7012 Management of Networks and Distributed Systems</td>
</tr>
<tr>
<td>CS4032 Distributed Systems</td>
<td>CS7030 Numerical Methods and Advanced Mathematical Modelling I</td>
</tr>
<tr>
<td>CS4052 Computer Graphics</td>
<td>CS7031 Graphics and Console Hardware</td>
</tr>
<tr>
<td>CS4053 Computer Vision</td>
<td>CS7032 Artificial Intelligence</td>
</tr>
<tr>
<td>CS4061 Artificial Intelligence Ia</td>
<td>CS7033 Real-time Animation</td>
</tr>
<tr>
<td>CS4071 Compiler Design II</td>
<td>CS7034 Augmented Reality</td>
</tr>
<tr>
<td></td>
<td>CS7048 Data Communications and Wireless Networking</td>
</tr>
<tr>
<td></td>
<td>CS7052 Sustainable Computing</td>
</tr>
<tr>
<td></td>
<td>CS7053 Security of Networks and Distributed Systems</td>
</tr>
<tr>
<td></td>
<td>CS7055 Real-time Rendering</td>
</tr>
<tr>
<td></td>
<td>CS7058 Numerical Methods and Advanced Mathematical Modelling II</td>
</tr>
<tr>
<td></td>
<td>CS7068 Financial Informatics for Computer Science</td>
</tr>
<tr>
<td></td>
<td>CS7069 Behavioural Finance for Computer Science</td>
</tr>
<tr>
<td></td>
<td>CS7073 Specification and Analysis of Reactive Systems</td>
</tr>
</tbody>
</table>

Options Table

Notes:

- It may not be possible to offer all the options each year and some modules have prerequisites.
- Additional modules may be added to this list.
- All modules have an ECTS weighting of 5 points.
10.1 CS4001 Fuzzy Logic

This module will introduce you to the exciting new field of fuzzy systems. Fuzzy systems are in almost daily use: your washing machine has fuzzy controls and many of the refrigerators do too. In both cases, the machines can sense the amount of stuff (clothes or food) and adjust their operations accordingly.

Fuzzy logic has been developed by computer scientists and control engineers over the last 30 odd years. Fuzzy logic is now being used in a range of critical systems ranging from image processing to the processing of financial time series.

10.2 CS4004 Formal Verification Techniques

Increasingly complex computer systems are becoming ever more important in every aspect of our lives. There are numerous examples where software bugs had extremely serious consequences, financially or to human well-being. This course will explore techniques to verify that computer systems satisfy their specifications; i.e. that they have no bugs and are thus safe to use. The focus will be mainly on software correctness but also on high-level system correctness.

We will learn the mathematical underpinnings of these techniques, logics and formal languages to specify correct system behaviour, and proof systems and some of the available tools to assist us with the verification process.

Students of this course will learn to mathematically prove the absence of bugs in software and high-level descriptions of computer systems. They will also gain experience with using verification tools to achieve this aim.

10.3 CS4012 Topics in Functional Programming

This module builds on CS3016 which introduced the fundamental concepts of functional programming. In CS4012 we will take an in-depth look at more advanced topics in functional programming and discuss some current research directions in the field.

10.4 CS4021 Advanced Computer Architecture

The aim of this module is to provide students with theoretical and practical experience of concurrent programming with and without locks.

10.5 CS4031 Mobile Communications

Effective wireless communication is the key enabling technology for realising the emerging ubiquitous computing vision. Mobile Communications is a final year option
module which imbues the next generation of graduates with a cognisance and awareness of both the capabilities and limitations of modern mobile devices. In this module students begin by learning about the underlying principles of wireless transmission and how these underpin the design of wireless communication networks. This material forms a platform for the subsequent analysis, assessment and implementation of a wide variety of modern wireless communication systems.

Critical appraisal of recent publications in IEEE and ACM journals is used to enhance each student’s ability to communicate effectively through the written medium.

10.6 CS4032 Distributed Systems

Building distributed applications is a difficult task due to the concurrency, communication latency, and possibility of partial failure that is inherent in distributed systems. As in other areas of computer science, the trend in providing support for building distributed applications has been towards presenting the application developer with ever higher levels of abstraction and, in the particular case of distributed programming, of location transparency. This course takes a critical look at some of the paradigms and architectural issues involved in distributed programming and their likely evolution.

Students will be given opportunities to develop their problem solving, programming and written communication skills by designing solutions to programming problems, implementing those solutions as fully networked distributed systems.

10.7 CS4052 Computer Graphics

The objective of this module is to equip the students with the fundamental understanding of the major elements of Computer Graphics and explore related areas including geometric modelling, rendering and animation. The main focus of the module is on the mathematics and algorithms used in the synthesis of computer graphics imagery and animation, and their practical application. Students are introduced to the standard architectures of modern graphical applications including details on the underlying hardware and low-level software components common to all such systems. The module is intended to enable students to bridge the gap between these low-level fundamental, components common to all computer applications, and the high-level abstract output in most interactive graphical applications.

Students are also introduced to OpenGL, a modern high-level graphics API which is widely used for 3D Design and Visualisation, along with the industry standard modelling software, 3D Studio Max, and this software is used throughout the course to demonstrate concepts and to allow the students to develop their own 3D models, scenes and applications.
10.8 CS4053 Computer Vision

The aim of this module is to give students a firm understanding of the theory underlying the processing and interpretation of visual information and the ability to apply that understanding to ubiquitous computing and entertainment related problems. It provides them with an opportunity to apply their problem-solving skills to an area which, while it is firmly part of computer science/engineering, draws strongly from other disciplines (physics, optics, psychology).

The module is based around problems so that the technology is always presented in context and during some tutorials students work in groups to design solutions to real world problems using the techniques that they have been taught. In addition, the module has a significant practical component so that students can appreciate how difficult it can be to apply the technology.

10.9 CS4061 Artificial Intelligence IIa

This is an in-depth initiation into some topics in AI, including the use of simple description logics and the application of finite-state methods to natural language processing. The syllabus includes topics such as knowledge representation, description logics, finite-state methods and reasoning about change in simple domains. Students will be able to appreciate the computational possibilities opened up by automata-theoretic approaches to reasoning.

10.10 CS4071 Compiler Design II

To teach the principles of compiler optimization.

10.11 CS7003 Middleware for Distributed Systems

To expose students to the complexities involved in designing and building distributed applications and to develop students analytical skills. To gain in-depth understanding of the principle paradigms used in the area. To gain an appreciation of the open research issues in the area. The course covers the underlying theory of distributed computing. A significant feature of the course is the use of an interactive teaching style in which students are encouraged to discover the fundamental principles of distributed computing by considering challenge problems collectively or in small groups before being presented with the relevant course material.
10.12 CS7004 Embedded Systems

The module will give students the opportunity to gain the knowledge and skills necessary to develop embedded systems. Students taking the module will study a real embedded hardware platform in depth (based on the widely used ARM7TDMI microcontroller) and will use this platform in conjunction with industry-standard software tools to develop embedded systems of varying complexity. Topics covered will be in the broad areas of computer architecture, systems software and I/O. Throughout the module, students will be given opportunities to consider issues of particular relevance in embedded systems design (e.g. development cost, power, performance and reliability).

10.13 CS7008 Vision Systems

The aim of this module is to give students a firm understanding of the theory underlying the processing and interpretation of visual information and the ability to apply that understanding to ubiquitous computing and entertainment related problems. It provides them with an opportunity to apply their problem-solving skills to an area which, while it is firmly part of computer science/engineering, draws strongly from other disciplines (physics, optics, psychology). The course is based around problems so that the technology is always presented in context and during some tutorials students work in groups to design solutions to real world problems using the techniques that they have been taught. In addition, the course has a significant practical component so that students can appreciate how difficult it can be to apply the technology.

10.14 CS7009 Networked Applications I

This module aims to provide an understanding of the world-wide web as an application platform that is becoming increasingly important economically and socially. It covers the fundamental content, social and meta)data structures that make up the web and how they can be represented, analysed and manipulated. It addresses the practical tools and techniques of web application programming, including client and server side programming languages, XML and semantic web information representation and analysis of application usage. It will encourage critical analysis of the impact of web applications on business and social concerns.
10.15 **CS7012 Management of Networks and Distributed Systems**

The aim of this module is to identify the issues and design approaches involved in managing networks & Services. To be capable of designing management solutions for various management application areas and organisations. Recognise and analyse the current management standards and technology trends in management of networks and distributed systems.

10.16 **CS7030/CS7058 Numerical Methods and Advanced Mathematical Modelling I/II**

The aim of this module is to encourage and foster the development of independent critical thinking. In particular students should be able to: (a) model problems using mathematics and statistics, (b) formulate and propose solutions, (c) infer from observation and interpret results.

The students should be able to use, critique, and edit (where relevant) Internet resources such as Wikipedia, Wolfram Research, etc.

10.17 **CS7031 Graphics and Console Hardware**

This module will give students a thorough overview of modern graphics hardware and multi-core systems. Each of the current generation of consoles will be analysed and compared in detail. The course will cover general purpose computer architecture e.g. memory hierarchies, SIMD & VLIW architectures, Vector units, multi-core, hyperthreading architectures and I/O busses. Students will become familiar with GPU pipeline architectures e.g. geometry, rasterisation, texture, fragment pixel and vertex shaders and newer Physics Processing Unit (PPU) and multi-GPU technology. Students will become familiar with the challenges of developing for these architectures through optimising compilers, compiler intrinsics and graphics card drivers.

10.18 **CS7032 Artificial Intelligence**

This module will present students with the state of the art in representing autonomous agents, decision making and learning. The students will develop a thorough understanding of the development of computer controlled characters that are aware of their environment, can react to external stimuli, behave according to sets of rules defined by a game designer and learn by interacting with the environment. The core of the module is reinforcement learning, presented within the autonomous agents framework described above. Other advanced topics, such as natural language generation and supervised learning, will also be presented.
10.19 CS7033 Real-time Animation

The aim of this module is to provide students with a deep understanding of the theory and techniques behind real time animation. We will explore computer animation and advanced issues such as behavioural animation and motion capture and also look at specific fundamental concepts such as interpolation.

10.20 CS7034 Augmented Reality

The aim of this module is to provide students with a solid background in alternative 3D compositing techniques using computer vision with applications in interactive interfaces—most notably augmented reality (AR) interfaces on mobile devices. Students will develop a comprehensive knowledge of 3D vision and will develop skills in the design and development of interactive augmented reality games.

Specific topics addressed in this module include: 3D vision; approaches to AR; alternative interface paradigms; spatial AR; lighting and illumination issues in AR.

10.21 CS7048 Data Communications and Wireless Networking

Students will acquire and demonstrate competence and capability in the areas of: Teamwork Time management Research Methods Project Planning Literature Review Project Specification Project and Experiment Design Project Execution Project Outcomes Clear, concise, appropriate and articulate dissemination of project outputs in the form of an IEEE/ACM calibre paper Students will be assessed and graded on all these elements.

10.22 CS7052 Sustainable Computing

This module introduces the foundations of sustainability and gives an appreciation for how energy is currently used in ICT and the problems created by the continuous growth of the ICT industry. The course will then delve into some detail on where power is consumed in current networks and how new techniques and trends will affect this. It will examine initiatives that have been taken to date, the impact that they have had and the prospects for future initiatives that will shape the industry.

10.23 CS7053 Security of Networks and Distributed Systems

The objectives of this module are: to gain a realistic understanding of risk, as it applies in distributed systems; to understand the main tools available to control risk from and how to use those whether as a designer, operator, user or security analyst;
and to enable the student to form opinions about issues such as full disclosure, some aspects of IPR (e.g. parts of DRM).

10.24 CS7055 Real-time Rendering

This module deals with programming for GPU pipeline architectures e.g. geometry, rasterisation, texturing, fragment / pixel and vertex shaders. Students will be introduced to shader systems and shader coding and will learn about modern game graphics engine architectures and developing real-time graphics applications, both for desktop PC and Xbox360. The module will explore advanced rendering concepts presented at leading international conferences such as SIGGRAPH and GDC.

10.25 CS7068 Financial Informatics

Problems in finance and business are amongst the hardest problems to be solved on computer systems. This module deals with two branches of computing that aim explicitly to deal with such problems: soft computing and grid computing. These problems include the known complex phenomena, including evolutionary growth and catastrophic failures, self-organisation and closely-coupled systems, viral attacks and contagion, and self-repair and learning.

10.26 CS7069 Behavioural Finance

This module will help you in understanding why stakeholders in a market sometimes over-react, tend to ignore factual information, and rely on rumours and on anecdotal evidence—the so-called irrational behaviour, or the rather exaggerated irrational exuberance. The stakeholders, it has to be said, have to take risky decisions and typically work with partial information. In many ways, behavioural finance studies the limits of mathematical and statistical description of economic and financial activities. The current literature on behavioural finance focuses on the limits of efficient market theory and on what motivates individuals to under- or over-react.

10.27 CS7073 Specification and Analysis of Reactive Systems

Computing systems are everywhere in modern society, becoming ever more sophisticated and controlling key aspects of our lives. They permeate our homes and offices, they control our transportation systems and underlie our security infrastructures. With this increasing dependency on computing systems and with their increasing complexity, it is crucial to have robust methods for their design and validation.
To meet this challenge computer scientists are increasingly using model-based approaches. This means that prior to any implementation effort, following the time-honoured engineering approach, a model of the system is first constructed and subjected to exhaustive analysis, the ultimate aim being to certify that the design embodied in the model meets its intended specification.

The aim of this module is to provide sufficient foundational expertise to enable students to effectively use modern *model checking* and *automatic verification* tools. Mathematical and logical based models will be introduced for the description and analysis of programs, with special emphasis on concurrent, reactive and possibly real-time systems. The core of the course will centre on formal languages for describing the intended behaviour of systems, logic systems for describing their properties, and algorithms for checking these properties. Lectures will be augmented by tutorials and workshops in which students will use a range of *model checking* and *automatic verification* tools.

## 11 Prizes

Gold medals are awarded by the Board to candidates of the first class who have shown exceptional merit at the annual degree examination in honor or professional courses (see http://www.tcd.ie/vpcao/administration/examinations/criteria-for-gold-medal.php).

Various studentships, scholarships, exhibitions, and other prizes are awarded to students on the results of honor and other examinations, provided that sufficient merit is shown. Monetary awards are sent direct to prize-winners unless otherwise stated under the regulations for the particular prize. For details please refer to the University Calendar.

At the annual examinations, a book prize (under review) is awarded to each candidate obtaining an overall first class honors grade in an honor or professional course. These prizes are not awarded in Year 4. These prizes, which are issued in the form of vouchers, can be exchanged by the student at designated booksellers. Book prizes are issued by the Examinations Office and are posted to recipient students at their home address.

### 11.1 Prizes in the Computer Science Programme

The following prizes are listed in the University Calendar for the Computer Science programme.
11.1.1 The Professor John G. Byrne Prize

This prize was established in 2014 with funds provided by Alumni of the School in honour of Professor John G. Byrne, Chair of Computer Science 1973–2003, and Head of the Department of Computer Science from its founding in 1969 to 1987 and from 1990 to 2001. In celebration of excellence, the prize is awarded annually to the student who achieves the highest overall result in the Masters Year of the Computer Science course provided the result is at Distinction level. Value, €1,024.

11.1.2 The Victor W. Graham Prize

This prize, founded in 1986 from funds subscribed by friends and pupils to mark Mr V. W. Graham’s retirement, is awarded to the Year 1 student in the moderatorship in computer science course who obtains the highest mark in the summer examination in pure mathematics. Value, €750.

11.1.3 The Ludgate Prize

This prize was instituted in 1991 in memory of Percy E. Ludgate, an Irish designer of an analytical engine. It is awarded to the student who submits the best project in Year 4 of the moderatorship in computer science. Value, €127.

11.1.4 The William Nurock Prize

This prize was founded in 1938 by a bequest from William Nurock. The conditions for the award of the prize were changed in 1984. It is now awarded annually to the best student in the final year examinations of the moderatorship in computer science, providing that such student also attains gold medal standard. Value, €1,000.

11.2 Scholarship

Foundation scholarship—("Schol")—is a College institution with a long history and high prestige. The examination for the award of scholarship is set and assessed so as to select students of outstanding ability. The objective of the foundation scholarship examination is to identify students who, at a level of evaluation appropriate to Year 2, can consistently demonstrate exceptional knowledge and understanding of their subjects.

The examination requires candidates to demonstrate skill in synthesising and integrating knowledge across the full range of the set examination materials; to demonstrate rigorous and informed critical thought; and, in appropriate disciplines, to demonstrate a highly-developed ability to solve problems and apply knowledge.

Attempting the scholarship examination is highly recommended.
For more information, please visit the University Calendar entry at http://www.tcd.ie/calendar/assets/pdf/foundation-scholarships.pdf.

12 Regulations

12.1 College Regulations

College regulations are set out in the University Calendar, which may be consulted in any College Library, the Enquiries Office, any academic or administrative office or online at http://www.tcd.ie/calendar/. The two most relevant extracts of the Calendar, entitled General Regulations and Information and Faculty of Engineering, Mathematics and Science, are handed out at registration at the beginning of the year. You are expected to be aware of the various regulations. Ignorance of the regulations is not a valid reason for failure to comply.

12.2 Attendance and Participation

Students are required to attend all lectures, laboratory and tutorial sessions associated with their programme of study and to participate fully in the academic work of their class.

Students must notify the lecturer concerned or their tutor as early as possible if they are unable to attend lectures, laboratories or tutorials or to submit coursework for any reason. Students who are absent for medical reasons should notify their tutor and will usually be required to provide a medical certificate.

12.3 Non-Satisfactory Attendance or Performance

At the end of each teaching term, students whose attendance or performance in coursework has not been satisfactory may be reported to the Senior Lecturer’s Office as non-satisfactory for that term (see University Calendar, General Regulations and Information). Normally, where students are non-satisfactory in a course for two terms in the year they may be refused permission to take their annual examinations and may be required to repeat the year.

Unless otherwise specified for an individual module, a student’s attendance and participation will be deemed to be non-satisfactory if they fail to attend more than one third of the scheduled lectures, laboratories or tutorials or make a serious attempt to complete more than one third of the coursework for any individual module.
12.4 Retaining Coursework

It is the responsibility of each student to retain a copy of any coursework that they submit.

12.5 Plagiarism

Students should be aware of the University’s policy regarding plagiarism.

Plagiarism is interpreted by the University as the act of presenting the work of others as one’s own work, without acknowledgement. Plagiarism is considered as academically fraudulent, and an offence against University discipline. The University considers plagiarism to be a major offence, and subject to the disciplinary procedures of the University. Plagiarism can arise from deliberate actions and also through careless thinking and/or methodology. The offence lies not in the attitude or intention of the perpetrator, but in the action and in its consequences. Plagiarism can arise from actions such as:

(a) copying another student’s work;
(b) enlisting another person or persons to complete an assignment on the student’s behalf;
(c) quoting directly, without acknowledgement, from books, articles or other sources, either in printed, recorded or electronic format;
(d) paraphrasing, without acknowledgement, the writings of other authors.

Examples (c) and (d) in particular can arise through careless thinking and/or methodology where students:

(i) fail to distinguish between their own ideas and those of others;
(ii) fail to take proper notes during preliminary research and therefore lose track of the sources from which the notes were drawn;
(iii) fail to distinguish between information which needs no acknowledgement because it is firmly in the public domain, and information which might be widely known, but which nevertheless requires some sort of acknowledgement;
(iv) come across a distinctive methodology or idea and fail to record its source.

All the above serve only as examples and are not exhaustive. Students should submit work done in co-operation with other students only when it is done with the full knowledge and permission of the lecturer concerned. Without this, work submitted
which is the product of collusion with other students may be considered to be plagiarism.

It is clearly understood that all members of the academic community use and build on the work of others. It is commonly accepted also, however, that we build on the work of others in an open and explicit manner, and with due acknowledgement. Many cases of plagiarism that arise could be avoided by following some simple guidelines:

(i) Any material used in a piece of work, of any form, that is not the original thought of the author should be fully referenced in the work and attributed to its source. The material should either be quoted directly or paraphrased. Either way, an explicit citation of the work referred to should be provided, in the text, in a footnote, or both. Not to do so is to commit plagiarism.

(ii) When taking notes from any source it is very important to record the precise words or ideas that are being used and their precise sources.

(iii) While the Internet often offers a wider range of possibilities for researching particular themes, it also requires particular attention to be paid to the distinction between one’s own work and the work of others.

Particular care should be taken to keep track of the source of the electronic information obtained from the Internet or other electronic sources and ensure that it is explicitly and correctly acknowledged. It is the responsibility of the author of any work to ensure that he/she does not commit plagiarism. Students should ensure the integrity of their work by seeking advice from their lecturers, tutor or supervisor on avoiding plagiarism.

If plagiarism as referred to above is suspected, procedures defined in the University Calendar, General Regulations and Information, will be followed.

12.5.1 Identifying Plagiarism

The School reserves the right to use plagiarism detection services, such as “Turnitin”, to identify potential cases of plagiarism.

12.6 Examinations and Assessment

Students are examined in the work of each year at the annual examinations. Supplemental examinations are held in Michaelmas term each year, except for the fifth year. Permission to take supplemental examinations will not normally be granted to students whom the court of examiners considers not to have made a serious attempt at the annual examinations unless an adequate explanation is furnished. Students must submit satisfactory course work in each year. Students who fail to do
so, or whose attendance is unsatisfactory, may be refused permission to take all or part of the annual examinations for the year.

Students who have not passed in its entirety any examination within eighteen months from the date on which they first became eligible for it, will be reported to the University Council as unsatisfactory with a recommendation for their exclusion from the course.

12.6.1 Progression

To progress to the next year of the programme, students must be successful at the annual examinations. Students who have failed the annual examination are required to take a supplemental examination or assessment in all modules in which they have not satisfied the examiners, as specified in the examination results. The method of assessment of modules varies between annual and supplemental examinations.

Annual Examinations

In order to be successful in the annual examinations, students must pass all modules. Alternatively, students may pass by compensation if they achieve an overall result of at least 40% and either:

- pass modules totalling 55 credits, and get a minimum mark of 30% in the failed module, or

- pass modules totalling 50 credits, and get a minimum mark of 35% in the failed module(s) – either one 10-credit module or two 5-credit modules.

A student’s overall mark will be calculated as the average of each module’s mark weighted by its ECTS rating. Students who pass the annual examinations will be awarded an overall grade according to the scale below.

*Important Note:* A module or modules may be identified as “non-compensatable” – i.e. a student must pass the module, and may not fail it and pass by compensation. In addition, a module or modules may be identified as “non-supplementable” – i.e. supplemental evaluation will not be available, requiring the student to repeat the year if the module is failed. Non-compensatable modules and non-supplementable modules are so identified in the College calendar and on the Student Information System (http://my.tcd.ie).

<table>
<thead>
<tr>
<th>Grade</th>
<th>Mark</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>70%–100%</td>
</tr>
<tr>
<td>II.1</td>
<td>60%–69%</td>
</tr>
<tr>
<td>II.2</td>
<td>50%–59%</td>
</tr>
<tr>
<td>III</td>
<td>40%–49%</td>
</tr>
</tbody>
</table>
Supplemental Examinations
In order to be successful in the supplemental examinations, students must pass all modules. Alternatively, students may pass by compensation if they achieve an overall result of at least 40% and either:

- pass modules totalling 55 credits, and get a minimum mark of 30% in the failed module, or

- pass modules totalling 50 credits, and get a minimum mark of 35% in the failed module(s) – either one 10-credit module or two 5-credit modules.

Any module(s) may be identified as “non-compensatable” – i.e. a student must pass the module, and may not fail it and pass by compensation.

A student’s overall mark will be calculated as the average of each module’s mark weighted by its ECTS rating. Where a module has been examined more than once, the mark achieved in the most recent examination will be used.

A student who does not pass by either of the methods above is required to repeat the year in full.

12.6.2 Progression to Year 4 of the Masters Programme
In order to progress to Year 4 of the Masters programme, students must achieve an overall mark of 50% or better in their Year 3 examinations. Students may take supplemental examinations in those modules in which they obtained less than 50% in the annual examinations, or may repeat Year 3 in its entirety to achieve the required standard.

12.6.3 Progression to Year 5 of the Masters Programme
In order to progress to Year 5 of the Masters programme, students must be on Year 4 of the Masters programme, must qualify for the award of BA (Mod.) in Computer Science and must achieve an overall mark of 60% or better in their Year 4 examinations. Students may repeat Year 4 in its entirety to achieve the required overall mark of 60% or better. There are no supplemental examinations in Year 4.

12.6.4 Repeating the Year
When a student must repeat the year, this must be completed in full. This includes completing all assessment elements of all modules (e.g. all continuous assessment requirements and laboratory experiments).
12.6.5 Module Assessment

The form of assessment at annual and supplemental examination stages varies between modules and may include a combination of coursework, written examination or other forms of assessment. The method of assessment and criteria for passing each module is set out the module descriptor, which may be found on the Student Information System (http://my.tcd.ie).

12.7 Awards

12.7.1 Ordinary BA Degree (exit only)

Students who have passed their Year 3 examinations may have an ordinary BA degree conferred if they do not choose, or are not allowed, to proceed to Year 4 of the programme or if they fail to complete satisfactorily Year 4 of the course. Except by permission of the University Council, on the recommendation of the Executive Committee of the School of Computer Science and Statistics, an ordinary BA degree may be conferred only on candidates who have spent at least two years in the University.

12.7.2 Moderatorship Degree

The BA (Moderatorship) degree result is awarded based on a combined mark from the annual sitting of the Year 3 examinations (which count for 20% of the moderatorship result) and Year 4 examinations (which count for 80% of the moderatorship result).

An honors degree is awarded to candidates who achieve a weighted average mark of 40% or more and, if one or more Year 4 modules has been failed, those modules account for not more than 10 ECTS credits. Where students are awarded an honors degree, the class of degree awarded is based on the weighted average mark achieved as follows: First Class Honors: 70%–100%, Second Class Honors, First Division: 60%–69%, Second Class Honours, Second Division: 50%–59%, Third Class Honors: 40%–49%. If the weighted average mark is below 40% or if modules accounting for more than 10 ECTS credits are failed, then students may be awarded an ordinary BA degree if they are clear of Year 3.

Students who have been successful in their Year 4 examinations may have the BA (Mod) degree conferred if they do not choose, or are not allowed, to proceed to the fifth year of the programme. Students exiting the programme after Year 4 obtain a moderatorship based on their results at their first attempt at the Year 4 examinations.
12.7.3 Master in Computer Science Degree

Successful candidates at the Year 5 examinations will be awarded a classified BA (Moderatorship) based on their results in Years 3 and 4, as set out above, and a Master in Computer Science or a Master in Computer Science with Distinction. A distinction shall require at least 70 per cent in the dissertation and at least 70 per cent in the final credit-weighted average mark.
### Outline Structure of Academic Year 2014/15

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25-Aug-14</td>
<td></td>
<td>43</td>
</tr>
<tr>
<td>2</td>
<td>01-Sep-14</td>
<td></td>
<td>27</td>
</tr>
<tr>
<td>3</td>
<td>08-Sep-14</td>
<td></td>
<td>34</td>
</tr>
<tr>
<td>4</td>
<td>15-Sep-14</td>
<td></td>
<td>35</td>
</tr>
<tr>
<td>5</td>
<td>22-Sep-14</td>
<td></td>
<td>36</td>
</tr>
<tr>
<td>6</td>
<td>29-Sep-14</td>
<td></td>
<td>37</td>
</tr>
<tr>
<td>7</td>
<td>06-Oct-14</td>
<td></td>
<td>38</td>
</tr>
<tr>
<td>8</td>
<td>13-Oct-14</td>
<td></td>
<td>39</td>
</tr>
<tr>
<td>9</td>
<td>20-Oct-14</td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>10</td>
<td>27-Oct-14</td>
<td></td>
<td>41</td>
</tr>
<tr>
<td>11</td>
<td>03-Nov-14</td>
<td></td>
<td>42</td>
</tr>
<tr>
<td>12</td>
<td>10-Nov-14</td>
<td></td>
<td>43</td>
</tr>
<tr>
<td>13</td>
<td>17-Nov-14</td>
<td></td>
<td>44</td>
</tr>
<tr>
<td>15</td>
<td>01-Dec-14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>08-Dec-14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>15-Dec-14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>22-Dec-14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>29-Dec-14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>05-Jan-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>12-Jan-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>19-Jan-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>26-Jan-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>02-Feb-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>09-Feb-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>16-Feb-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>23-Feb-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>02-Mar-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>09-Mar-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>16-Mar-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>23-Mar-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>30-Mar-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>06-Apr-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>13-Apr-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>20-Apr-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>27-Apr-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>04-May-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>11-May-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>18-May-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>25-May-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>01-Jun-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>08-Jun-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>15-Jun-15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>22-Jun-15</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Statutory Term (Michaelmas) begins**

**Standard Week**

- **Orientation Week/Freshers’ Week**
- **Teaching Week 1**
- **Teaching Week 2**
- **Teaching Week 3**
- **Teaching Week 4**
- **Teaching Week 5**
- **Teaching Week 6** (Monday, Public Holiday)
- **Teaching Week 7** - **Study Week**
- **Teaching Week 8**
- **Teaching Week 9**
- **Teaching Week 10**
- **Teaching Week 11**
- **Teaching Week 12**

**Special Weeks**

- **Christmas Period** (College closed)
- **Foundation Scholarship Examinations**
- **Revision (Monday, Easter Monday)**
- **Revision Trinity Week (Monday, Trinity Monday)**
- **Annual Examinations 1**
- **Annual Examinations 2** (Monday, Public Holiday)
- **Annual Examinations 3**
- **Annual Examinations 4**
- **Marking/Examiners/Results**
- **Marking/Examiners/Results** (Monday, Public Holiday)
- **Marking/Examiners/Results/Examiners/Courts of First Appeal**
- **Courses of First Appeal/Academic Appeals**
- **Postgraduate dissertations/theses/Research 1-8**
- **Nineweeks between end of statutory (Trinity) term and commencement of statutory (Michaelmas) term. This period is also used for writing up Masters dissertations and research theses due for submission in September.**
- **Ends Friday 28 August 2015**

**Notes:**

- It may be necessary to hold some exams in the preceding week.
- Michaelmas term ends Friday 12 December 2014
- Hilary Term begins
- Hilary Term ends Friday 3 April 2015
- Trinity Term begins
- Annual Examination period: Four weeks followed by five weeks for marking, examiners’ meetings, publication of results, Courts of First Appeal and Academic Appeals.
- Statutory (Trinity) Term ends Friday 26 June 2015
<table>
<thead>
<tr>
<th>Time</th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>09.00 – 10.00</td>
<td>MT: CS1003: Lect LB04</td>
<td>MT: CS1025: Lect LB04</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15.00 – 16.00</td>
<td>MT: CS1003: Lect LB08 (2 hours)</td>
<td>MT: CS1026: Lab LG35/36</td>
<td>MT: CS1021: Tutorial DO</td>
<td>MT: CS1026: Lab LG35/36</td>
<td>MT: CS1026: Lab LG35/36</td>
</tr>
<tr>
<td>Time</td>
<td>Monday</td>
<td>Tuesday</td>
<td>Wednesday</td>
<td>Thursday</td>
<td>Friday</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------</td>
<td>--------------------------</td>
<td>--------------------------</td>
<td>--------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>09.00 - 10.00</td>
<td>CS1003: Lect LB08</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.00 - 11.00</td>
<td>CS1010: Lect MacNeil</td>
<td>CS1003: Lect LB04</td>
<td>CS1022: Lect LB08</td>
<td>CS1031: Lab LG36</td>
<td>CS1010/CS1022: Lab ICT Lab 1/2, LG35/36</td>
</tr>
<tr>
<td>11.00 - 12.00</td>
<td>CS1013: Lect LB04</td>
<td>CS1031: Lect LB01</td>
<td>CS1010: Lect LB01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.00 - 13.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13.00 - 14.00</td>
<td>CS1026: Lect LB04</td>
<td>CS1026: Lect LB08</td>
<td></td>
<td>CS1026 Lab LG35/36</td>
<td></td>
</tr>
<tr>
<td>14.00 - 15.00</td>
<td>CS1022: Lect LB01</td>
<td>CS1022: Lect Joly</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15.00 - 16.00</td>
<td>CS1031: Lab LB08 (2 hours)</td>
<td>CS1003: Lect LB08</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16.00 - 17.00</td>
<td></td>
<td>CS1010: Lect Joly</td>
<td>CS1013: Lect ICT Lab 1/2, LG36, LB04</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17.00 - 18.00</td>
<td>CS1026: Tut LB08</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


Code: Module, ECTS, Lecturer, Time/Location

CS1003: Mathematics I, 10 ECTS, Dr M Huggard/Dr H Gibbons
CS1010: Introduction to Programming, 10 ECTS, Dr K Dawson/Howe, Dr A Hughes
CS1013: Programming Project 1, 5 ECTS, Dr N Cardoz
CS1022: Introduction to Computing II, 5 ECTS, Dr J Dukes
CS1026: Electrotechnology, 5 ECTS, Dr E O’Nuallain

Locations:
- LB04: Lloyd Institute, Lecture Theatre 01/04/08
- LB01/02: Lloyd Institute, Lecture Theatre 01/04/08
- LG35/LG36: O'Reilly Institute
- LG 35 & LG 36: O'Reilly Institute
- Goldsmith, Goldsmith Hall

Term Dates:
- MT 2/9/14 – 12/12/14 (Reading Wk 3-7 Nov)
- 12/1/15 – 3/4/15 (Reading Wk 23-27 Feb)
### School of Computer Science and Statistics
#### Integrated Computer Science: Year 2 Timetable 2014-15

<table>
<thead>
<tr>
<th>Time</th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>09.00 – 10.00</td>
<td>HT: CS2022: Lect M20</td>
<td></td>
<td>MT: CS2010: Lab LG12 (2 hours)</td>
<td>MT: CS2010: Lect LB01</td>
<td></td>
</tr>
<tr>
<td>10.00 – 11.00</td>
<td>MT: CS2041: Lab LB01 (2 hours)</td>
<td>MT: CS2010: Lab LB01</td>
<td>MT: CS2010: Lab LB01</td>
<td>HT: CS2010: Lect LB01</td>
<td>MT: CS2010: Lab LB01</td>
</tr>
</tbody>
</table>

**Code:** Module: ECTS: Lecturer  
CS2010 Programming Techniques: 5 ECTS: Dr H Gibbons, Dr V Koutavas  
CS2013 Programming Project II: 5 ECTS: Mr G Strong/Dr C Harris  
CS2014 Systems Programming I: 5 ECTS: Dr D Gregg  
CS2016 Concurrent Systems and Operating Systems: 5 ECTS: Dr M Brady  
CS2021 Microprocessor Systems: 5 ECTS: Dr M MacNeil  
CS2022 Computer Architecture I: 5 ECTS: Dr M MacNeil  

**Locations:**  
LB101/102: Lloyd Institute: First Floor, Room 1.07/1.20  
M21: M21b, Museum Building, Room M21  
ICT Lab 1/2: ICT Hut Labs  
JOLY: Hamilton Building  
LG35/36: O'Reilly Institute  

**Term Dates:**  
MT: 22/10 – 11/12/14 (Reading: Wk 3-7 Nov)  
HT: 12/10 – 3/1/15 (Reading: Wk 23-27 Feb)  

<table>
<thead>
<tr>
<th>Time</th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>09:00</td>
<td></td>
<td>MT: CS3012: Lect LB04</td>
<td></td>
<td>HT: ST3009: Lect LB01</td>
<td></td>
</tr>
<tr>
<td>11:00</td>
<td>MT: CS3021: Lect M20</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17:00</td>
<td>MT: CS3041: Lect LB04</td>
<td></td>
<td></td>
<td></td>
<td>MT: CS3041: Lect LB08</td>
</tr>
</tbody>
</table>
## School of Computer Science and Statistics

### Integrated Computer Science: Year 4 Timetable 2014-15

<table>
<thead>
<tr>
<th>Time</th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>09:00</td>
<td>CS4001: Lect LB01</td>
<td>MT: CS4051: Lect LB01</td>
<td>MT: CS4012: Lect LB120</td>
<td>MT: CS4004: Lect LB120</td>
<td>MT: CS4081: Lect LB01</td>
</tr>
<tr>
<td>14:00</td>
<td>CS4001: Lect LB01</td>
<td>MT: CS4004: Lect LB120</td>
<td>MT: CS4052: Lect LB120</td>
<td>MT: CS4081: Lect LB01</td>
<td>MT: CS4004: Lect LB120</td>
</tr>
<tr>
<td>16:00</td>
<td>CS4001: Lect LB01</td>
<td>MT: CS4004: Lect LB120</td>
<td>MT: CS4052: Lect LB120</td>
<td>MT: CS4081: Lect LB01</td>
<td>MT: CS4004: Lect LB120</td>
</tr>
<tr>
<td>17:00</td>
<td>CS4001: Lect LB01</td>
<td>MT: CS4004: Lect LB120</td>
<td>MT: CS4052: Lect LB120</td>
<td>MT: CS4081: Lect LB01</td>
<td>MT: CS4004: Lect LB120</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time</th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
</table>

See http://www.tcd.ie/Maps/ for maps and directions for the campus.
21 First Year in University

Everybody says college is different from school. Of course, in lots of obvious ways it is different, and no doubt you’ll enjoy finding out just what those differences are. In not-so-obvious ways though, college is very different from school, and in this section we concentrate on how the academic side of university life is different and what you need to do about it.

1. You are not at school. We want you to do more than simply reproduce what you are told in a lecture. You need to get a good command of the material. In computing-related disciplines, the best way to do this—and the best way to know that you have really learned something—is to apply your new knowledge to solving new problems; not just the examples done in class, but to similar problems you’ll find in textbooks or elsewhere (later on, as a professional computer scientist, you will have to apply your knowledge to problems you have never seen before—now is the time to start).

2. Expect the material to be covered much faster than at school. Lecture time is at a premium, so it must be used efficiently. You cannot be taught everything in lectures and tutorials. It is your responsibility to learn the material. Most of this learning will take place outside the classroom, and you must be willing to put in the study time necessary to ensure that this learning takes place. If you do fall behind in a course—that is, if you can’t continue to understand the lectures as they are given—then you really need to make the effort to catch up right away. Don’t be tempted to think that you can somehow catch up at the end of the year—it’s almost impossible.

3. A lecturer’s job is primarily to provide you with a framework, with some of the particulars, to guide you in doing your learning of the concepts and methods that comprise the material of the course. It is not to ‘programme’ you with isolated facts and problem types or to monitor your progress. Your job is to fill out that framework with a thorough understanding of the material.

4. You are expected to read the textbook for comprehension. It gives the detailed account of the material of the course. It also contains many examples of problems worked out, and these should be used to supplement those you see in the lecture. The textbook is not a novel; you cannot simply skim through it from start to finish. Reading the textbook must often be slow-going and careful; frequently you’ll need to use pencil and paper to work through the material, but you can work at your own pace.

5. As for when to read the textbook, it’s a good idea to read the appropriate section ahead of the lecture. This way, although you may not understand it
fully, you’ll be prepared for the lecture, and you’ll have a good idea what areas to ask questions about. If you haven’t looked at the book beforehand, pick up what you can from the lecture (absorb the general idea and/or take thorough notes) and count on sorting it out later while studying the book and transcribing your notes.

6. Laboratories and tutorials are far more important than the marks you might get for them, because they give you a chance to develop your understanding of the subject. They are also a good ‘reality check’ for you to see just how much you really do understand. Use them wisely.

7. In examinations, the examiners set out to probe your mastery of the material in the course. Primarily, they’ll be looking for your command of the material, as noted above. You’ll probably have to solve problems you’ve never seen before. (To be sure, you’ll have encountered similar problems, but they won’t be the same.) Hence, preparing for examinations simply by remembering lots of answers without understanding them simply won’t work; examinations test your understanding of the material as well.

This section is adapted from Teaching at the University Level by Steven Zucker in Notices of the AMS August 1996.

22 Research Ethics

Any research project that involves human participation conducted through this course (for example, a questionnaire or survey, or system user-evaluation, etc.) must have independent review by a Research Ethics Committee before its commencement.

A basic principle is that prospective participants should be fully informed about the research and its implications for them as participants, with time to reflect on the possibility for participation prior to being asked to sign an informed consent form.

Application forms, with further information and guidelines, can be found here: http://www.scss.tcd.ie/undergraduate/ethics/

It takes time to prepare an application for research ethics approval, to have the application considered, and to respond to feedback on the application where issues are raised. You should plan in your work for the time it takes to obtain research ethics approval.

Retrospective approval will not be granted.

Please also note, research conducted in the School of Computer Science and Statistics should be undertaken with cognisance of the TCD Guidelines for Good Research Practice; see http://www.tcd.ie/about/policies/assets/pdf/TCDGoodResearchPractice.pdf.
23 Student Supports

23.1 Programming Centre

The Programming Centre is available to all Computer Science students free of charge. The centre operates as a drop-in service where you can get help with any problems you might have with programming in your courses. For further information, please visit http://www.scss.tcd.ie/misc/psc/.

23.2 Student Learning Development

Student Learning Development provides learning support to help students reach their academic potential. We run workshops, have extensive online resources and provide individual consultations. The service is offered by the College’s Student Counselling Service. To find out more, visit their website at http://www.tcd.ie/Student_Counselling/student-learning/.

23.3 Maths Help Room

The Maths Help Room offers free assistance to students who are having difficulty with Mathematics, Statistics or related courses. It runs every week of term and at certain times out of term. The Maths helproom is a drop in centre, where you can bring in a maths or stats question and get some help. It is run by the School of Mathematics and further information is available at http://www.maths.tcd.ie/~mathshelp/.

23.4 Academic Concerns: Sources of Assistance

- Other students in the class.
- The course lecturer.
- Engineering class representatives.
- Your tutor (or any other tutor if you cannot find yours), or the Senior Tutor.
- The Director of Undergraduate Teaching & Learning, Dr Mike Brady.
- The Students’ Union Education Officer, email education@tcdsu.org), web http://www.tcdsu.org.
• Peer Mentors. Junior Freshmen are introduced to their Peer Mentors during Freshers' Week. The Student to Student Service runs also provides peer mentoring for the other years. For information about all Student to Student services, please email student2student@tcd.ie or phone 8962438.

23.5 Personal Concerns: Sources of Assistance

• Your tutor (or any other tutor if you cannot find yours), or the Senior Tutor, phone 8962251.

• The Student Counselling Service, 3rd Floor, 7–9 South Leinster Street, College. Opening hours: 9:15 am to 5:10 pm Monday to Friday during lecture term. Phone: 8961407. Email: student-counselling@tcd.ie Web: http://www.tcd.ie/Student_Counselling.

• Niteline (Thursday to Tuesday during term time only, 9 pm–2.30 am) Phone: 1800 793 793. Web: http://www.niteline.ie/.

• The College Health Service, House 47, College. Medical Director: Dr David McGrath. Phone: 8961591 or 8961556.

• The Welfare Officer, Students' Union, House 6, College. Email: welfare@tcdsu.org;

• The Chaplains, House 27, College.  
  Paddy Gleeson and Peter Sexton SJ (Catholic) 8961260  
  Darren McCallig (Church of Ireland) 8961402  
  Julian Hamilton (Methodist) 8961901

• Any student, member of staff or other person with whom you feel able to discuss your concerns;

• Disability Services Coordinator, Mr Declan Treanor, Room 3055, Arts Building, phone: 8963475, email: dtreanor@tcd.ie

NOTE: IF YOU HAVE A CONCERN OF ANY SORT, PLEASE TALK TO SOMEONE STRAIGHT AWAY

23.6 Tutors

A tutor is a member of the academic staff who is appointed to look after the general welfare and development of the students in his or her care. Whilst your tutor may be one of your lecturers, the role of tutor is quite separate from the teaching role.
Tutors are a first point of contact and a source of support, both on arrival in college and at any time during your time in college. They provide confidential help and advice on personal as well as academic issues or on anything that has an impact on your life. They will also, if necessary, support and defend your point of view in your relations with the college.

24 Health and Safety

The College Emergency Number, for safety or personal security concerns, is extension 1999 on the internal College telephone system, +353-1-8961999 from a mobile phone or an external landline.