Department of Computer Science: Courses for Visiting Students 2010/11

The Department is one of the UK's leading centres within the field, and has an international reputation for research in artificial intelligence, data mining, machine learning, fundamental areas of theoretical computer science, bioinformatics, computational biology, distributed architectures, and vision and signal processing.

A wide range of courses is available. Level 1 (denoted CS1***) courses cover the web and internet, databases, computer engineering, the theory of computer languages and Java programming. Level 2 courses (CS2***) include computer graphics, artificial intelligence, information security, operating systems, theoretical computer science and robotics. Level 3 (CS3***) courses are more specialised and include advanced courses in the main research areas previously mentioned, games technology, computational finance, and digital sound and music. These courses (listed below), are open to all Study Abroad, International Exchange, and Erasmus students, subject to fulfilment of pre-requisites as specified in the relevant course descriptions.

Entry Requirements:

Generally, for Level 1 courses, some mathematical background at university level is very advantageous. Level 2 and 3 courses normally require completion of CS1**** courses or an equivalent course successfully completed at an overseas university.

* Please note that whilst this course runs in term 1, it is only suitable for students studying for the full academic year, and not for just term 1 only.

The information contained in these course outlines is correct at the time of publication (July 2009) but may be subject to change. Every effort will be made to maintain accurate and up-to-date information.

Level 1 Courses:

CS1801	Object-oriented programming	Full Year
CS1820	Computing Laboratory (Robotics)	Term 1*
CS1830	Computing Laboratory (Games)	Term 2
CS1840	Internet Services	Term 1*
CS1850	Databases	Term 2
CS1860	Mathematical Structures	Term 1*
CS1870	Machine Fundamentals	Term 2

Level 2 Courses:

CS2800	Software Engineering	Term 1*
CS2810	Team Project	Term 2
CS2820	Programming Paradigms	Term 2
CS2830	Robotics	Term 1*
CS2840	Graphics and HCI	Term 1*
CS2850	Network Operating Systems	Term 2
CS2860	Algorithms and Complexity I	Term 1*
CS2870	Algorithms and Complexity II	Term 2

IY2760	Introduction to Information Security	Term 1*

Level 3 Courses:

CS3110	Bioinformatics	Term 2
CS3220	Fundamentals of Digital Sound and Music	Term 1*
CS3230	Computer Games Technology	Term 1*
CS3470	Compilers and code generation	Term 2
CS3490	Computational Optimisation	Term 2
CS3580	Advanced Data Communications	Term 2
CS3750	Concurrent and Parallel Programming	Term 1*
CS3760	Information Security	Term 1*
CS3920	Computer Learning	Term 1*
CS3930	Computational Finance	Term 2
CS3940	Intelligent agents and multi-agent systems	Term 1*
IY3770	Trusted Computing Platforms	Term 1*
IY3780	Secure Software Engineering	Term 2

Level 1 Courses

Course Code:	**CS1801**	**Availability**:	Full Year only	**Unit Value**:	1.0
Course Title:	**Object-oriented programming**				
Level:	1	**Admin. Fees:**	n/a		
Prerequisites:	University level mathematics				
Course Aims:	To teach fundamental programming skills.				
Learning Outcomes:	By the end of this course a student should be able to: • solve basic programming tasks • understand and use basic object-oriented concepts • appreciate the need for program documentation, testing, readability and modifiability				
Course Description:	*Program basics:* variables, types, scope, lifetimes *Control flow:* if-constructs, for-loops, while-loops *GUI:* event-based programming, callbacks, layout managers *Data structures:* strings, sets, lists, trees *Exceptions:* throwing and catching *File I/O:* streams, file reading, writing and copying *Applets:* applet life cycle, security managers				
Teaching and Learning:	Lecture based delivery, supported by laboratory classes and small group tutorials. Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	Herbert Schildt: Java: a Beginner's Guide, 4th Edition, McGraw-Hill, 2006, ISBN 0072263849 Cary S. Horstmann: Big Java, 3rd Edition, Wiley, 2007, ISBN 987 0470105542				
Assessment:	**Formative Assessment & Feedback:** Small group tutorials in which students work through programming exercises under supervision, getting verbal feedback. A zero-weighted test in the first term for which marks will be returned. Students who fail the mid-year test will be required to resit it. **Summative Assessment:** Exam (90%) 3 hours. Answer 2 out of 3 questions from each of Sections 1 and 2. No calculators. Coursework (10%) Five equally weighted assignments. First term test 0%.				
				Last updated:	15/02/10 JCW

Course Code:	**CS1820**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Computing Laboratory (Robotics)**				
Level:	1	**Admin. Fees:**	n/a		
Prerequisites:	University level mathematics				
Course Aims:	To provide a practical approach to programming and the building of computer systems. To introduce elementary robotics concepts To reinforce the earning of elementary programming				
Learning Outcomes:	By the end of this course a student should be able to: • Programme a mobile robot to execute pre-defined movements • Understand the operation of basic sensors • Understand the essentials of real time event-driven programming				
Course Description:	*Lego NXT system:* control brick capabilities, motors, sensor classes *Sensor operation:* light sensor, ultrasonic sensors, pushbuttons *Simple movement:* motors, gearing, idler wheels, tracked vehicles *Actuators:* grippers, extensible arms *Projects:* a sequence of scripted exercises, with opportunities for student led expansion				
Teaching and Learning:	Teacher led laboratory classes and self-driven, scripted group project work. Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	J. Kelly: LEGO Mindstorms NXT-G Programming Guide, Technology in Action Press, 2007, ISBN 10: 1590598717 Brian Bagnell: Maximum Lego NXT: Building Robots with Java Brains, Variant Press 2007, ISBN 10: 0973864915				
Assessment:	**Formative Assessment & Feedback:** Five formative worksheets. Verbal feedback on progress will be given during the laboratory sessions. **Summative Assessment:** Coursework (100%) Five formative worksheets (0%) Group project presentation (25%) Individual report on group project (75%)				
				Last updated:	15/02/10 JCW

Course Code:	**CS1830**	**Availability**:	Term 2	**Unit Value**:	0.5
Course Title:	**Computing Laboratory (Games)**				
Level:	1	**Admin. Fees:**	n/a		
Prerequisites:	University level mathematics				
Course Aims:	To provide a practical approach to programming and the building of computer applications To introduce elementary gaming concepts To reinforce the earning of elementary programming				
Learning Outcomes:	By the end of this course a student should be able to: • Write simple games programmes using C# and XNA • Load and run single-player games on an X-box • Understand the essentials of real world simulation				
Course Description:	*C# fundamentals:* relationship to Java, APIs *X-box capabilities:* the games console market, hardware block diagram, networking *Gaming graphics:* animation, shading, graphics engines *Gaming physics:* collision detection, ballistics *User experience:* classic game styles, scoring, statistics *Projects:* a sequence of scripted exercises, with opportunities for student led expansion				
Teaching and Learning:	Teacher led laboratory classes and self-driven, scripted project work. Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	Stephen Cawood: Microsoft XNA Game Studio Creators Guide, McGraw-Hill, 2007 ISBN 10: 007149071X Benjamin Nitschke: Professional XNA Game Programming: For Xbox 360 and Windows, Wrox, 2007 ISBN 10: 0470126779				
Assessment:	**Formative Assessment & Feedback:** Five formative worksheets. Verbal feedback on progress will be given during the laboratory sessions. **Summative Assessment:** Coursework (100%) Five formative worksheets (0%) Group project presentation (25%) Individual report on group project (75%)				
				Last updated:	17/02/10 JCW

Course Code:	**CS1840**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Internet Services**				
Level:	1	**Admin. Fees:**	n/a		
Prerequisites:	University level mathematics				
Course Aims:	To provide an introduction to internet technologies and their use in an increasingly e-centric industry.				
Learning Outcomes:	By the end of this course a student should be able to: • understand and describe the client-server model • program both the client and server sides of a simple application • use XML to construct Web pages • understand the fundamentals of web services				
Course Description:	*Internet Basics:* client-server model, protocols *Internet Protocol Stack:* example protocols such as DNS, SMTP *Web Technologies:* HTTP, XML *Scripting languages:* JavaScript (client-side) and PHP (server-side) *Web Services:* SOAP message exchange, WSDL service description, UDDI service discovery, introduction to REST, ATOM publishing Protocol				
Teaching and Learning:	Lecture based delivery, supported by laboratory classes and tutorials. Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	C.D. Knuckles: Introduction To Interactive Programming On The Internet Using HTML And Javascript, Wiley, 2000, ISBN 9780471383666 J.F. Kurose and K.W. Ross: Computer Networking: A Top-Down Approach Using The Internet, Addison-Wesley, 2008, ISBN 0321497708 P.K Yuen and V. Lau: Practical Web Technologies, Addison Wesley, 2003, ISBN 0201750767				
Assessment:	**Formative Assessment & Feedback:** Immediate verbal feedback will be given on class exercises carried out in the laboratory sessions. Three mandatory zero-weighted assignments and a zero-weighted mid-term test will be marked and returned. **Summative Assessment:** Exam (100%) One and one half hours. Answer 3 out of 4 questions. No calculators. Coursework (0%) Three mandatory assignments and a mid-term test.				
				Last updated:	17/02/10 JCW

Course Code:	**CS1850**	**Availability**:	Term 2	**Unit Value**:	0.5
Course Title:	**Databases**				
Level:	1	**Admin. Fees:**	n/a		
Prerequisites:	University level mathematics				
Course Aims:	To provide the basic concepts of database technology. To describe the need for database integrity and robustness. To demonstrate the use of a modern database system in a web-based environment.				
Learning Outcomes:	By the end of this course a student should be able to: • explain the issues involved in database design and the theory of the relational view of data • describe the crucial issues concerning database integrity and recovery from failure • write SQL programs • use Java to access a database on the internet				
Course Description:	*Data modelling:* views, subschema, data dictionary, data independence, entity relationship model *The relational model:* relations, attributes, domains, relational algebra *Database design:* normalisation, normal forms, entities and attributes *SQL:* basic SQL, correspondence between the relational model and SQL commands, simple queries, combination and sub-queries *Administration and implementation:* integrity, recovery from failure, concurrency, MySQL data entry, deletion and updating, forms, report writing, JDBC technology and API				
Teaching and Learning:	Lecture based delivery, supported by laboratory classes and tutorials. Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	C.J. Date: An Introduction To Database Systems, 8th edition, Addison Wesley, 2003, ISBN 0321197844 P. Rob and C. Coronel: Database Systems: Design, Implementation, And Management, Course Technology Inc, 2002, ISBN 061906269X R. Elmasri and S. Navathe: Fundamentals Of Database Systems, 3rd edition, Addison Wesley, 2000, ISBN 0805317554 D.M. Kroenke: Database Processing, Prentice Hall, 1997, ISBN 0137378424				
Assessment:	**Formative Assessment & Feedback:** Immediate verbal feedback will be given on class exercises carried out in the laboratory sessions. The mid-term test will be graded to provide feedback. **Summative Assessment:** Exam (90%) One and one half hours. Answer 3 out of 4 questions. No calculators. Coursework (10%) 50 minute in-class written mid-term test				
				Last updated:	17/02/10 JCW

Course Code:	**CS1860**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Mathematical Structures**				
Level:	1	**Admin. Fees:**	n/a		
Prerequisites:	University level mathematics				
Course Aims:	To provide insights and skills in rigor and formal reasoning in a way that allows reasoning about behaviour, correctness and performance in a programming environment. To provide basic knowledge of the formal structures for program data representation.				
Learning Outcomes:	By the end of this course a student should be able to: • define and reason about sets, relations, functions and cardinality • write and reason about recursive definitions and prove results by induction and contradiction • represent and reason about problems using graphs • use vectors and transformations to define and manipulate graphical objects • have an understanding of basic probability and statistics suitable for use in studying artificial intelligence				
Course Description:	Sets: defining sets, logic notation, proofs by construction, counterexample and contradiction Relations: relations, orderings, functions, bisections Recursion: recursive definitions and induction, cardinality of infinite sets Graph theory: graphs, trees and spanning trees, directed graphs Vector spaces: vectors, transformations, bases, matrices, determinants Probability: elementary and conditional probability, binomial distribution, random variables and Bayes theorem				
Teaching and Learning:	Lecture based delivery, supported by small group tutorials. Normally 3 hours of lectures per week.				
Key Bibliography:	Rosen: Discrete Mathematics And Its Applications, McGraw Hill, 2006, ISBN 0071244743 Ross and Wright: Discrete Mathematics, Prentice Hall, 2003, ISBN 0130652474				
Assessment:	**Formative Assessment & Feedback:** Individual discussion of non-assessed tutorial exercises and discussion of assessed worksheets in small group tutorials. **Summative Assessment:** Exam (90%) One and one half hours. Answer 3 out of 4 questions. No calculators. Coursework (10%) Three equally weighted assessed worksheets.				
				Last updated:	17/02/10 JCW

Course Code:	**CS1870**	**Availability**:	Term 2	**Unit Value**:	0.5
Course Title:	**Machine Fundamentals**				
Level:	1	**Admin. Fees:**	n/a		
Prerequisites:	University level mathematics				
Course Aims:	To explain the theory and use of logic in the description, specification and behaviour of machine processes. To provide insights and skills for dealing with large and infinite objects in a way that allows them to be implemented in a programming environment.				
Learning Outcomes:	By the end of this course a student should be able to: • use formal logic to design, reason about and minimise switching circuits • write basic programs in assembly language • understand binary representations of signed and unsigned integers • write regular expressions to describe sets and build deterministic automata to recognise these sets • use automata to design and reason about sequential flow systems				
Course Description:	*Numbers:* binary number systems, two's complement notation. *Logic:* propositions, logical formulae, truth tables and logical equivalences. Predicates, proofs and logical inference. Normal forms. *Networks:* series and parallel switching circuits, network minimisation. *Automata:* regular expressions, automata, Thompson's construction and the subset construction. Push down automata. Turing machines and non-computability. *Low level languages:* assembly language programming.				
Teaching and Learning:	Lecture based delivery, supported by laboratory classes and group tutorials. Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	Rosen: Discrete Mathematics And Its Applications, McGraw Hill, 2006, ISBN 0071244743 Brookshear: Computer Science An Overview, Addison-Wesley, 2007, ISBN 0321524039				
Assessment:	**Formative Assessment & Feedback:** Individual discussion of non-assessed tutorial exercises and discussion of assessed worksheets in small group tutorials. **Summative Assessment:** **Exam** (90%) One and one half hours. Answer 3 out of 4 questions. No calculators. **Coursework** (10%) Three equally weighted assessed worksheets.				
				Last updated:	17/02/10 JCW

Level 2 Courses

Course Code:	**CS2800**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Software Engineering**				
Level:	2	**Admin. Fees:**	n/a		
Prerequisites:	CS1801 or equivalent				
Course Aims:	To teach advanced programming skills and the techniques and disciplines needed to be able develop software as part of a team. To raise awareness of professional, social and ethical issues involved in the sustainable exploitation of computer technology.				
Learning Outcomes:	By the end of this course a student should be able to: • understand the software engineering techniques and managerial discipline required to work as part of a team • understand and use basic object-oriented concepts • appreciate the need for program documentation, testing, readability and modifiability • understand social and ethical issues relating to the use of computer technology in society				
Course Description:	*Modularity:* objects, inheritance, encapsulation, polymorphism, mutability, collections, iterators, invariants, exceptions, design issues *Software engineering:* models of software development, planning, project documentation, cost and resource estimation, quality assurance, requirements analysis *Object oriented design:* Notation for design, identifying objects, classes, attributes and methods. Class relationships, design patterns *Programming methodologies:* choice of programming language, program structure, style, and layout. Coding standards *Testing:* Program analysis, black and white box testing, defensive programming, system, integration and acceptance testing.				
Teaching and Learning:	Lecture based delivery with class exercises and worksheets. Normally 3 hours of lectures per week.				
Key Bibliography:	Cary S. Horstmann: Big Java, 3rd Edition, Wiley, 2007, ISBN 978 0470105542 Cary S. Horstmann: Object Oriented Design And Patterns, 2nd Edition, Wiley, 2005, ISBN 978 0471744870 E. Freeman, E. Freeman: B. Bates and K. Sierra, Head First Design Patterns, O'Reilly, 2004, ISBN 978 0596007126				
Assessment:	**Formative assessment:** Two mandatory zero-weighted exercises will be marked and returned. **Summative assessment:** Exam (90%) 2 hours. Answer 4 out of 6 questions. No calculators. Coursework (10%) Two equally weighted assessed assignments 10%. Two mandatory exercises 0%.				
				Last updated:	17/02/10 JCW

Course Code:	**CS2810**	**Availability**:	Term 2	**Unit Value**:	0.5
Course Title:	**Team Project**				
Level:	2	**Admin. Fees:**	n/a		
Prerequisites:	CS1801, CS2800 or equivalent				
Course Aims:	To provide practical experience of developing software as part of team.				
Learning Outcomes:	By the end of this course a student should be able to: • work in a team to produce a substantial product using software engineering techniques • apply managerial discipline • use advanced object-oriented analysis procedures to solve complex problems				
Course Description:	The software lifecycle: Models of software development, planning, documentation, costing, quality assurance. Requirements extraction: working with customers, resolving conflicting demands Team structures: large teams, ownership, change control A large software development project, conducted by teams of between four and six students, based on a simplified real world problem				
Teaching and Learning:	Lecture based delivery together with directed student lead team work and team meetings with the team supervisor. Normally 1 hour of lectures per week.				
Key Bibliography:	Richard Whitehead: Leading a Software Development Team, Addison Wesley, 2001 ISBN 10: 0201675269 Cary S. Horstmann: Big Java, 3rd Edition, Wiley, 2007, ISBN 978 0470105542				
Assessment:	**Formative Assessment & Feedback:** Teams will provide scheduled project deliverables, and verbal feedback will be provided during the team supervisor meetings. **Summative Assessment:** **Coursework** (100%) There is a set of deliverables from which a team mark and an individual mark will be awarded. The details of the deliverables are given in the course documentation.				
				Last updated:	17/02/10 JCW

Course Code:	**CS2820**	**Availability**:	Term 2	**Unit Value**:	0.5
Course Title:	**Programming Paradigms**				
Level:	2	**Admin. Fees:**	n/a		
Prerequisites:	CS1801 or equivalent				
Course Aims:	To teach programming skills in the procedural, logic and functional programming environments. To enable students to understand the operation of dynamic memory and heap management				
Learning Outcomes:	By the end of this course a student should be able to: • demonstate a working understanding of program execution using a simplified model of the main memory hierarchy • explain the implementation of data structures at the level of memory references • implement search algorithms in Prolog • implement list processing algorithms in ML				
Course Description:	*Modelling the computer:* CPU/main memory backing store, the pigeon hole model of memory, indexing and pointer hopping, heap management *Performance*: abstracting performance, counting basic operations, impact of cache *Procedural languages:* dynamic memory handling in C++ and Java *Prolog:* clauses, predicates and queries, a simple interpreter, manipulation of symbolic expressions *ML:* type inference, eager evaluation, pattern matching, recursion, imperative aspects				
Teaching and Learning:	Lecture based delivery. Normally 3 hours of lectures per week.				
Key Bibliography:	Frantisek Franik: Memory as a Programming Concept in C and C++, Cambridge University Press, 2003 ISBN 10: 0521520436 Ivan Bratko: Prolog programming for artificial intelligence, Addison Wesley, 2001 ISBN 10: 0201403757 Matthias Felleisen et al: The Little MLer, MIT Press, 1998 ISBN-10: 026256114X				
Assessment:	**Formative Assessment & Feedback:** Two mandatory zero-weighted assignments will be marked and returned. Some model solutions will be discussed in class. **Summative Assessment:** Exam (100%) 2 hours. Answer 4 out of 6 questions. No calculators. Coursework (0%) Two mandatory assignments.				
				Last updated:	17/02/10 JCW

Course Code:	**CS2830**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Robotics**				
Level:	2	**Admin. Fees:**	n/a		
Prerequisites:	CS1801, CS1820				
Course Aims:	To introduce practical robotics To introduce simple dynamic control systems				
Learning Outcomes:	By the end of this course a student should be able to: • implement mobile robots that construct simple real world maps • understand the basics of PID feedback systems • program significant robotic application in Java on Lego NXT systems				
Course Description:	Motion control: line following, distance sensor balancing, dead reckoning Sensors: characterisation of sensors by dynamic range, sensitivity and repeatability PID control: proportional, integral and derivative terms, loop tuning Maze solving and search: wall following, Tremaux, heuristic search World mapping: blocks world, SLAM A project to implement a robot that can construct a world map				
Teaching and Learning:	Lecture based delivery, supported by laboratory classes. Self guided project work. Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	John L. Craig: Introduction to Robotics: Mechanics and Control, Pearson Education, 2003, ISBN 10: 0131236296 J. Kelly: LEGO Mindstorms NXT-G Programming Guide, Technology in Action Press, 2007, ISBN 10: 1590598717 Brian Bagnell: Maximum Lego NXT: Building Robots with Java Brains, Variant Press 2007, ISBN 10: 0973864915				
Assessment:	**Formative Assessment & Feedback:** Verbal feedback on progress will be given during the laboratory sessions. **Summative Assessment:** Exam (70%) 2 hours. Answer 4 out of 6 questions. No calculators. Coursework (30%) One assessed project				
				Last updated:	17/02/10 JCW

Course Code:	**CS2840**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Graphics and HCI**				
Level:	2	**Admin. Fees:**	n/a		
Prerequisites:	CS1801 or equivalent				
Course Aims:	To introduce students to the basic concepts and algorithms for producing computer graphics To introduce Human Computer Interfaces				
Learning Outcomes:	By the end of this course a student should be able to: • demonstrate a knowledge of basic computer graphics • use mathematical techniques including co-ordinate geometry, homogenous vectors and projective environments in graphics applications • design and implement a simple 3D animated scene • describe good HCI design principles				
Course Description:	*Geometry:* vectors, matrices, dot and cross product, transformations, finding intersections, homogeneous representations of transformations, rendering pipeline *Perspective:* projection, viewpoint, computation of viewing transformation from camera position *Appearance rendering:* construction of spline curves and surfaces, lighting, reflection, texture, Gouraud and Phong shading, ray tracing and photorealistic rendering, representation and reproduction of colour *Implementation:* practical implementation of a 3D animation *HCI*: ergonomics, user interfaces, user centred design				
Teaching and Learning:	Lecture based delivery, supported by practical sessions. Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	J. Foley, A. van Dam, S. Feiner, J. Hughes and R. Phillips: Introduction To Computer Graphics, Addison Wesley, 1994, ISBN 978 0201609219 Dave Shreiner et al: OpenGL(R) Programming Guide, Addison Wesley, 2006, ISBN 10: 0321335732				
Assessment:	**Formative Assessment & Feedback:** Non-assessed programming exercises and an assessed individual project. The non-assessed exercises will be discussed in class. **Summative Assessment:** Exam (80%) 2 hours. Answer 4 out of 6 questions. No calculators. Coursework (20%) Individual project				
				Last updated:	17/02/10 JCW

Course Code:	**CS2850**	**Availability**:	Term 2	**Unit Value**:	0.5
Course Title:	**Network Operating Systems**				
Level:	2	**Admin. Fees:**	n/a		
Prerequisites:	CS1801 or equivalent				
Course Aims:	To introduce students to the principles of the function and architecture of network operating systems.				
Learning Outcomes:	By the end of this course a student should be able to: • demonstrate an understanding of the principles of computer operating systems • evaluate the theory and practice of an existing operating system • write basic shell scripts • understand the use of network services at operating systems level				
Course Description:	*Introductory topics:* role of an operating system, historical perspectives, computer architecture *Processes:* process management and scheduling, inter-process interaction, UNIX and Windows *Memory:* fixed and dynamic partitioning, swapping and paging, virtual memory, page replacement algorithms, implementation issues *File systems:* implementation and maintenance, file systems in UNIX and Windows, access control, security issues, distributed file systems, network management issues *UNIX shell:* starting programs, input and output streams, pipes, filters, UNIX utilities				
Teaching and Learning:	Lecture based delivery, supported by practical sessions. Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	A.S. Tanenbaum: Modern Operating Systems , 2nd Edition, Prentice Hall, 2001 , ISBN 01336006639 W. Stallings: Operating Systems: Internals And Design Principles, 4th Edition, Prentice Hall, 2001, ISBN 0136006329 G. Nutt, Operating Systems: A Modern Perspective, 2nd Edition, Addison Wesley, 2002, ISBN 0201773449				
Assessment:	**Formative Assessment & Feedback:** Written grades will be given for the assessed assignments. **Summative Assessment:** Exam (80%) 2 hours. Answer 4 out of 6 questions. No calculators. Coursework (20%) Two equally weighted assignment sheets				
				Last updated:	17/02/10 JCW

Course Code:	**CS2860**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Algorithms and Complexity I**				
Level:	2	**Admin. Fees:**	n/a		
Prerequisites:	CS1801, CS1860 or equivalent				
Course Aims:	To teach the design of algorithms and data structures from the point of view of time and space complexity.				
Learning Outcomes:	By the end of this course a student should be able to: • understand and reason about alternative data structure representations • implement and reason about alternative implementations for basic algorithms • calculate the complexity of basic algorithms				
Course Description:	*Complexity:* counting, big-O notation, best-case, worst-case and average-case analysis *Sorting algorithms:* implementation and analysis of bubble sort, insertion sort, merge sort, quick sort, heap sort *Searching algorithms:* implementation and analysis of linear search, binary search, binary search trees, hash coding *Game theory:* Min-max trees, alpha-beta trees *String matching:* naive and Rabin-Karp string matching algorithms				
Teaching and Learning:	Lecture based delivery, supported by laboratory classes and problems sessions. Normally 3 hours of lectures per week.				
Key Bibliography:	T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein: Introduction To Algorithms, 2nd Edition, MIT Press, Cambridge MA, 2001, ISBN 0262531968.				
Assessment:	**Formative Assessment & Feedback:** Two mandatory zero-weighted assignments and a zero-weighted mid-term test will be marked and returned. Students who fail the mid-term test will be required to resit it. **Summative Assessment:** **Exam** (100%) 2 hours. Answer 4 out of 6 questions. No calculators. **Coursework** (0%) Two assignments and a mid-term test.				
				Last updated:	17/02/10 JCW

Course Code:	**CS2870**	**Availability**:	Term 2	**Unit Value**:	0.5
Course Title:	**Algorithms and Complexity II**				
Level:	2	**Admin. Fees:**	n/a		
Prerequisites:	CS1801, CS1860, CS2860 or equivalent				
Course Aims:	To teach the design of fundamental graph theoretic algorithms, and to introduce simple notations and tools for this task.				
Learning Outcomes:	By the end of this course a student should be able to: • implement and reason about fundamental graph-theoretic algorithms • describe several applications of graph-theoretic algorithms • understand the significance of NP-hard problems and use heuristic approaches				
Course Description:	Basic graph theory: graph models, basic notions, paths and walks, shortest path problem Graphs and trees: directed graphs, cycles, trees, minimum spanning trees algorithm, connectivity Algorithms and applications: matching in graphs, network flows, Hamilton cycles, travelling salesman problem, acyclic and strongly connected graphs, graph colourings, planar graphs NP-hardness: examples, simple transformations, exact and approximation algorithms, heuristics Spectral graph theory: Markov transition matrices and random walks, spectral decomposition.				
Teaching and Learning:	Lecture based delivery, supported by laboratory classes and problems sessions. Normally 3 hours of lectures per week.				
Key Bibliography:	W. Kocay and D.L. Kreher: Graphs, Algorithms And Optimization, Chapman & Hall, 2005, ISBN 1584883960				
Assessment:	**Formative Assessment & Feedback:** Two mandatory zero-weighted assignments and a zero-weighted mid-term test will be marked and returned. **Summative Assessment:** Exam (100%) 2 hours. Answer 4 out of 6 questions. No calculators. Coursework (0%) Two assignments and a mid-term test.				
Timetable:				Last updated:	17/02/10 JCW

Course Code:	**IY2760**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Introduction to Information Security**				
Level:	2	**Admin. Fees:**	n/a		
Prerequisites:	None				
Course Aims:	Information storage, communication, processing and management is probably the core application of computer technology, and the successful management of information resources is fundamental to the success of business both now and in the future. The significance of information to commercial enterprises means that securing this information is of fundamental importance. This course is concerned with providing a general introduction to the subject of Information Security as a whole. The key technologies are introduced, and, after taking this course, the student will be equipped with the necessary background knowledge to go on and gain an in-depth understanding of specialised topics within the subject area.				
Learning Outcomes:	On successful completion of this course, students will be able to: • explain the issues involved in the successful and secure management of information resources; • apply the background knowledge gained to further study of the specialised topics within the subject area.				
Course Description:	The course will cover the following topics: • *Introduction*: What is security (covering notions of Confidentiality, Integrity, and Availability)? Security threats and risks. Security management (ISO/IEC 17799). Data Protection legislation. • *Elements of cryptography*: Ciphers (DES/AES). Message Authentication codes (MACs). Public key ciphers and digital signatures (RSA). • *Access control*: Access Control Lists, capabilities, security labels (MAC and DAC), and role-based access control. • *Database security*: The access path, views for security, integrity controls. • *Personal computer security*: Viruses, spyware, restricting access. • *Identity verification:* Use and storage of conventional passwords. Dynamic password schemes. Biometric techniques. Use of tokens (dumb and intelligent), including the use of smart cards. • *CASE STUDY I*: Security of a modern Operating System (e.g. Windows or Linux). • *Network security concepts:* The concepts of security services and security mechanisms (as in ISO 7498-2). Firewalls. • *Authentication and key distribution:* The importance and relatedness of the concepts of key management and entity authentication in a network. Objectives of an entity authentication protocol. Some fundamental protocols (e.g. Kerberos). Using authentication protocols for key distribution, and other approaches to key establishment (including public key certificates and X.509). • *Security standards bodies:* Introduction to roles of ISO (notably ISO/IEC SC27), ITU and IETF. A brief introduction to (security related) standards developed and under development. • *CASE STUDY II*: Security for a networked application e.g. SSL/TLS.				
Teaching and Learning:	150 hours, made up of 33 hours of lectures and 117 hours non-assessed coursework and private study. The coursework is designed to allow students to apply their knowledge to applications with which they are familiar.				

Key Bibliography:	The main recommended text for this course is: D. Gollmann, Computer Security, John Wiley & Sons, 2005 (2nd edition). Useful background: • C.P. Pfleeger and S. L. Pfleeger, Security in Computing, Prentice Hall, 2006 (4th edition). • W. Ford, Computer Communications Security, 1994. • Ross Anderson, Security Engineering, John Wiley and Sons, 2001. • B. Schneier, Applied cryptography, 1996 (2nd edition). • W. Stallings, Cryptography and network security - principles and practice, Prentice Hall, 2006 (3rd edition).		
Assessment:	**Formative Assessment & Feedback:** This course will contain mandatory non-assessed coursework. Answers will be marked and returned with feedback. **Summative Assessment:** Exam 100%: 2 hours, 4 questions to be answered out of a choice of 6. This course will be assessed solely by written examination.		
Timetable:		Last updated:	17/02/10 JCW

The information contained in these course outlines is correct at the time of publication but may be subject to change. Every effort will be made to maintain accurate and up-to-date information.

Level 3 Courses:

Course Code:	**CS3110**	**Availability**:	Term 2	**Unit Value**:	0.5
Course Title:	**Bioinformatics**				
Level:	3	**Admin. Fees:**	n/a		
Prerequisites:	None				
Course Aims:	To introduce the main approaches currently in use in bioinformatics, with special emphasis on the analysis of DNA and protein sequences emerging from genome sequencing projects.				
Learning Outcomes:	By the end of this course a student should be able to: • assess the main approaches currently in use in bioinformatics; • demonstrate an understanding of the analysis of DNA and protein sequences.				
Course Description:	*Basic molecular biology:* introduction to the basic components of living cells, their functions and interactions, and to other concepts essential to understanding the use of computers in biology *Gene prediction:* ab initio and by homology; *Gene regulation:* finding promoters and regulatory regions: protein binding sites; evolutionary approaches to sequence analysis; evolutionary genetics; *Phylogenetic trees* *Dynamic Programming and HMMs*				
Teaching and Learning:	Lecture based delivery. Normally 3 hours of lectures per week.				
Key Bibliography:	R.Durbin, S.Eddy, A.Krogh, and G.Mitchinson: Biological sequence analysis, Cambridge University Press, 1998 : ISBN-10: 0521629713 R.Duda,P.Hart,D.Stork: Pattern classification, John Wiley & Sons, 2001:ISBN-10: 0471056693 B.Levin: Genes VIII, Prentice Hall, 2004: ISBN-10: 0131239244 A.M.Lesk: Introduction to Bioinformatics, Oxford University Press, 2005: ISBN-10: 0199277877				
Assessment:	**Formative Assessment & Feedback:** Return of marked zero-weighted exercises. **Summative Assessment:** Exam (90%) 2 hours. Answer 2 out of 5 questions. No calculators. Coursework (10%) Two 2 equally weighted assignments 10%. Class exercises 0%.				
				Last updated:	17/02/10 JCW

Course Code:	**CS3220**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Fundamentals of Digital Sound and Music**				
Level:	3	**Admin. Fees:**	n/a		
Prerequisites:	CS2800 or equivalent				
Course Aims:	To give students an understanding of the fundamental computational ideas relating to the processing of sound and music on a computer, from the basic mathematics underlying signal processing through to the use of up-to-date technological solutions to the problems of storing, synthesizing and analyzing speech and music.				
Learning Outcomes:	By the end of this course a student should be able to: • demonstrate an understanding of lossy compression algorithms for sound and music • use Fourier analysis to synthesize and transform sounds in the frequency domain • understand the MIDI format, related JAVA APIs, and write a synthesizer from scratch • understand the basic acoustic properties of speech and standard techniques for speech synthesis and recognition				
Course Description:	*Digital Signal Processing for Sound:* spectral analysis, Fourier transforms, Shannon-Nyquist theorem, phone, CD, and SACD bandwidth. *Compression algorithms:* lossless compression, MP3, Ogg-vorbis. perceptual coding. *Music Technology*: sequencing and synthesis, MIDI format, Java API, sequences, digital instruments and synthesis of instrumental sounds, interfaces *Signal Processing*: digital post-processing and effects. *Harmony:* pythagorean harmony, temperaments, harmony, Helmholtz theory of consonance. *Speech:* acoustic properties of speech, spectrograms, phonetics, vowel formants, stress assignment, text to speech systems, automatic speech recognition using HMMs, language models, decoders and acoustic models. *Labs:* frequency domain representations of sounds, digital signal processing, sound-effects				
Teaching and Learning:	Lecture based delivery, supported by laboratory classes. Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	Curtis Roads: The Computer Music Tutorial, MIT Press, 2000: ISBN-10: 0262680823 Eduardo Miranda: Computer Sound Synthesis for the Electronic Musician,1998: ISBN-10: 024051517X Craig Lindley: Digital Audio with Java, Prentice Hall, 2000: ISBN-10: 0130876763				
Assessment:	**Formative Assessment & Feedback:** Return of marked zero-weighted assignments, verbal feedback on project work during the laboratory sessions. **Summative Assessment:** Exam (80%) 2 hours. Answer 3 out of 5 questions. No calculators. Coursework (20%) Digital audio programming project 20%, 4 assignments 0%				
				Last updated:	17/02/10 JCW

Course Code:	**CS3230**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Computer Games Technology**				
Level:	3	**Admin. Fees:**	n/a		
Prerequisites:	CS2800, CS2820, CS2840 or equivalent				
Course Aims:	To introduce the range of techniques employed in computer games; to provide a motivation for the study of some technically very challenging topics in AI and computational geometry; to provide an analytical framework within which students can analyse computer games.				
Learning Outcomes:	By the end of this course a student should be able to: • design and implement a simple 2d computer game from scratch • demonstrate an understanding of core algorithms for 3d computational geometry • understand the software engineering problems involved in implementing games • understand the use of physical laws and violations of them in simulated environments • analyse the use of artifcial intelligence in computer games.				
Course Description:	*Game design*: flow, progressive reinforcement, basic game genres, games on mobile phones, consoles and PCs, serious games. *2D and 3D graphics*: Sprites, animation with threads, double buffered display, hit detection, first person views, axonometric projections, particle systems, OpenGL, DirectX and Java 3d, rendering algorithms, textures, shading algorithms. *Software engineering*: content pipeline, graphics engines, physics engines, open source game libraries, Microsoft XNA. *Computational geometry*: data structures and algorithms for 2d and 3d virtual environements, convexity, BSP algorithms, Voronoi tesselations, stochastic generation of 3d landscapes. *Sound*: music and sound effects, audio libraries, the role of sound in games; networking in games, client-server and peer-to-peer networking, TCP/UDP design tradeoffs *Game physics*: 3d dynamics, rigid objects, friction, elastic and inelastic collisions, kinematics. *AI*: route-finding, goals and planning, cheating; incomplete information; dynamic environments, flocking and teamwork. *Game theory*: game theoretic optimality, pure and impure strategies.				
Teaching and Learning:	Lecture based delivery. Normally 3 hours of lectures per week.				
Key Bibliography:	Raph Koster: Theory of fun in game design: Paraglyph Press 2005 ISBN-10: 1932111972 Ian Millington : Artificial Intelligence for Games by :Morgan Kaufmann: ISBN-10: 0124977820 Ian Millington: Game Physics Engine Development, Morgan Kaufmann : ISBN-10:12369471X				
Assessment:	**Formative Assessment & Feedback:** Return of marked zero-weighted exercises. Solutions will be discussed in class. **Summative Assessment:** **Exam** (80%) 2 hours. Answer 3 out of 5 questions. No calculators. **Coursework** (20%) Case study 5%, programming assignment 5%, project 10%, weekly exercises 0%				
				Last updated:	17/02/10 JCW

DEPARTMENT OF: Computer Science				Academic Session:	
Course Code:	**CS3470**	**Course Value:**	**0.5**	**Status:**	Option
Course Title:	Compilers and code generation			**Availability:**	Term 2
Prerequisites:	CS1801, CS1870			**Recommended:**	CS1860
Co-ordinator:					
Course Staff					
Aims:	To describe how to construct and implement interpreters and compilers for modern processors				
Learning Outcomes:	By the end of this course a student should be able to: • explain the role and structure of a compiler and the standard stages of compilation • to build a DFA based lexical analyser for a set of tokens specified using regular expressions • Advanced students should be able to construct regular expressions, which define specified set of strings • use grammars to define context free languages and to build parsers for them • describe syntax directed translators and use them to construct intermediate code • describe various types of error detection and recovery • generate three address code from source code.				
Course Content:	*Lexical analysis:* input buffering, regular expressions, finite state automata, hash coded symbol tables, LEX *Syntax analysis:* context free grammars, derivations, recursive descent and table-based LR parsing, YACC*Semantics:* abstract parse trees, syntax directed translation, intermediate forms and three address code, attributes, annotated parse trees, semantic rules and attribute grammars, translation schemes and L-attribute grammars *Error detection and recovery*: classes of error and error recovery in top down parsers *Intermediate code generation*: using attribute grammars, three address code *Code improvement*: basic blocks and code improvement techniques, flow graphs, loop improvement and fusion, directed a cyclic graphs for identifying code improvements *RDP:* parser generation with **rdp,** using **rdp** promotion operators, using **rdp** to generate a full compiler for a small language				
Teaching & Learning Methods:	Lecture based delivery, supported by practical classes Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	Aho, Lam, Sethi, Ullman: Compilers: Principles, Techniques, and Tools, Addison Wesley, 2006 ISBN: 10: 0321486811 J. Tremblay and P.G. Sorenson: The theory and practice of compiler writing, McGraw Hill, 1985, ISBN 0-07-065161-2 N. Wirth: Algorithms + Data Structures = Programs, Prentice Hall, 1976.				
Formative Assessment & Feedback:	Return of coursework grades and comments, in-class discussion of coursework solutions				
Summative Assessment:	**Exam** (80%) 2 hours. Answer 3 out of 5 questions. No calculators. **Coursework** (20%) 4 equally weighted assignments **Deadlines: As shown on departmental coursework deadlines sheet**				

Course Code:	**CS3490**	**Availability**:	Term 2	**Unit Value**:	0.5
Course Title:	**Computational Optimisation**				
Level:	3	**Admin. Fees:**	n/a		
Prerequisites:	CS2860, CS2870 or equivalent				
Course Aims:	To introduce the basic models of computational optimisation and the basic algorithms for solving computational optimisation problems. To demonstrate the theoretical and computational methods of analysing computational optimisation algorithms and will discuss available software packages for solving problems.				
Learning Outcomes:	By the end of this course a student should be able to: • demonstrate an understanding of the basic models of computational optimisation • apply the basic algorithms for solving computational optimisation problems • evaluate theoretical and computational methods of analysing computational optimisation algorithms • use enhanced algorithmic and mathematical skills.				
Course Description:	*Introduction:* algorithm efficiency and problem complexity *Linear programming (LP)*: LP model, formulating problems as LP problems, graphical solution, simplex method, duality in LP;, decomposition of LP problems, LP software *Integer Programming (IP)*: IP models, branch-and-bound algorithm *Computational optimisation problems*: greedy-type algorithms, construction heuristics and local search for the TSP *Heuristics*: DMERN problem and signed graphs; heuristics for DMERN, experimental analysis of embedded network and TSP heuristics, theoretical analysis of heuristics, meta-heuristics.				
Teaching and Learning:	Lecture based delivery, Normally 3 hours of lectures per week.				
Key Bibliography:	M.W. Carter and C.C. Price: Operations Research: A practical Introduction, CRC, 2001 (001.424 CAR): ISBN-10: 0849322561 Z. Michalewicz and D.B. Fogel: How to Solve It: Modern Heuristics, Springer, 2000: ISBN-10: 3540224947 F. Glover and M. Laguna: Tabu Search, Kluwer, 1997: ISBN-10: 079239965X J. Bang-Jensen and G. Gutin: Digraphs: Theory, Algorithms and Applications, Springer, 2000: ISBN-10: 1852336110				
Assessment:	**Formative Assessment & Feedback:** Return of marked zero-weighted assignments. Some solutions discussed in class. **Summative Assessment:** **Exam** (100%) 2 hours. Answer 3 out of 5 questions. No calculators. **Coursework** (0%) 3 assignments 0%.				
				Last updated:	17/02/10 JCW

Course Code:	**CS3580**	**Availability**:	Term 2	**Unit Value**:	0.5
Course Title:	**Advanced Data Communications**				
Level:	3	**Admin. Fees:**	n/a		
Prerequisites:	CS1840 or equivalent				
Course Aims:	To cover a range of important topics in modern data communications including text, image, audio and video transfer over networking infrastructures and recent trends in network management and network security.				
Learning Outcomes:	By the end of this course a student should be able to: • explain how multimedia communications work • describe coding and compressing algorithms for text, image and video • describe Internet technologies in terms of supporting QoS • demonstrate an understanding of most common security terms and concepts • describe modern network security mechanisms and their applications • explain Internet network management issues and possible solutions				
Course Description:	*Introduction:* layering, abstraction, and standards. The ISO/OSI reference model. The Internet reference model. *Multimedia communications:* multimedia information representation, coding and compression, applications and standards, quality of service (QoS) requirements. *Internet:* IP addressing, routeing algorithms and routeing protocols, RIP, OSPF, the Internet multicast model, scheduling and queue management. *Security in communication networks:* security issues, security mechanisms, secure protocols. *Network management:* network management issues, infrastructure and framework for Internet management.				
Teaching and Learning:	Lecture based delivery. Normally 3 hours of lectures per week.				
Key Bibliography:	James F. Kurose and Keith W. Ross: Computer Networking: a top-down approach featuring the Internet, Addison-Wesley: ISBN-10: 0321227352 Fred Halsall: Multimedia Communications: applications, networks, protocols and standards, Addison-Wesley: ISBN-10: 0321227352 William Stallings: Cryptography and Network Security: principles and practice, Prentice Hall: ISBN-10: 0131873164				
Assessment:	**Formative Assessment & Feedback:** Return of marked zero-weighted assignments. **Summative Assessment:** **Exam** (100%) 2 hours. Answer 3 out of 5 questions. No calculators. **Coursework** (0%) 3 assignments 0%.				
				Last updated:	17/02/10 JCW

Course Code:	**CS3750**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Concurrent and Parallel Programming**				
Level:	3	**Admin. Fees:**	n/a		
Prerequisites:	CS1801, CS2800 or equivalent				
Course Aims:	To introduce the principles and practical implementations of concurrent programming, to implement algorithms in a concurrent fashion, and to give an understanding of the variety of different concurrent architectures available.				
Learning Outcomes:	By the end of this course a student should be able to: • understand the theoretical underpinnings of concurrent programming • understand deadlock and race conditions in shared states • understand threaded and message passing paradigms and be familiar with writing software using these paradigms • implement a variety of different algorithms concurrently and understand how their performance varies • describe wide variety of concurrent platforms, ranging from multi-core processors to PC farms to more widely distributed computing				
Course Description:	*CSP*: events, processes, choice, synchronisation, traces, transition diagrams, interleaving, specification. *Behaviour of processes*: explosion of possible states in concurrent processes in comparison to sequential programming, unexpected traces, deadlock. *Java and CSP*: JCSP, implementing basic ideas of CSP in JCSP. *Threading paradigm*: implementation in Java, launching threads, shared states and race conditions, locking, conditional locking and deadlock. *Message Passing paradigm*: implementation in MPI, synchronization, broadcasting, one-toone communication, synchronous and asynchronous communication. *Algorithms*: search, sort, graphs and dynamic programming implemented concurrently, limitations of concurrency. *Architectures*: SIMD and MIMD, shared and distributed memory. Hyper-Threading, multi-core processors and supercomputing. PC farms and Beowulf clusters. Distributed computing.				
Teaching and Learning:	Lecture based delivery. Normally 3 hours of lectures per week.				
Key Bibliography:	Cary S. Horstmann: Big Java, 3rd Edition, Wiley, 2007, ISBN 987 0470105542 A.Grama,A.Gupta,G.Karypis,V.Kumar: Introduction to Parallel Computing:ISBN-10: 0201648652				
Assessment:	**Formative Assessment & Feedback:** Return of marked zero-weighted exercises. Some solutions will be discussed in class. **Summative Assessment:** Exam (80%) 2 hours. Answer 3 out of 5 questions. No calculators. Coursework (20%) equally weighted assignments 20%, exercises 0%				
				Last updated:	17/02/10 JCW

Course Code:	**CS3760**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Information Security**				
Level:	3	**Admin. Fees:**	n/a		
Prerequisites:	n/a				
Course Aims:	To providing a general introduction to Information Security. To provide the background knowledge needed to study the subject area at an advanced level.				
Learning Outcomes:	By the end of this course a student should be able to: • demonstrate a thorough understanding of the successful and secure management of information resources • apply the background knowledge gained to more advanced study of the specialised topics within the subject area				
Course Description:	*Introduction:* What is security (covering notions of Confidentiality, Integrity, and Availability)? Security threats and risks. Security management (the DTI code of practice). The Data Protection Act. *Elements of cryptography:* Ciphers (DES). Authentication codes (MACs). Public key ciphers and digital signatures (RSA). *Access control:* Access Control Lists, capabilities, security label, role-based access control. *Database security:* The access path, views for security, integrity controls. *Personal computer security:* Viruses, restricting access. *Identity verification:* Use and storage of conventional passwords. Dynamic password schemes. Biometric techniques. Use of tokens (dumb and intelligent), smart cards. *CASE STUDY I:* Unix security. *Network security concepts:* Security services and security mechanisms (as in ISO 7498--2). *Authentication and key distribution:* Key management and entity authentication in a network. Objectives of an entity authentication protocol. Some fundamental protocols (e.g. Kerberos). Using authentication protocols for key distribution, and other approaches to key establishment (including public key certificates and X.509). Firewalls. *Security standards bodies:* Introduction to roles of ISO, ITU, CEN, ETSI and BSI. The main roles of ISO/IEC SC21 and SC27 and a brief introduction to (security related) standards. *CASE STUDY II:* Security for a LAN. Internet Privacy Enhanced Mail (PEM).				
Teaching and Learning:	Lecture based delivery. Normally 3 hours of lectures per week.				
Key Bibliography:	D. Gollmann: Computer Security, John Wiley & Sons, 2nd Edition (Nov 2005) ISBN-10: 0470862939 C.P. Pfleeger: Security in Computing, 2nd edition, 1997: ISBN-10: 0133374866 Ross Anderson: Security Engineering, 2000: ISBN-10: 0470068523 B. Schneier: Applied cryptography, 2nd edition, 1996: ISBN-10: 0471117099 W. Stallings: Cryptography and network security - principles and practice, 3rd edition, 2002, ISBN10: 0130914290 / 4th edition, 2006, ISBN-10: 0131873164				
Assessment:	**Formative Assessment & Feedback:** Return of marked zero-weighted assignments. **Summative Assessment:** Exam (100%) 2 hours. Answer 3 out of 5 questions. No calculators. Coursework (0%)				

Timetable:		Last updated:	17/02/10 JCW

Course Code:	**CS3920**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Computer Learning**				
Level:	3	**Admin. Fees:**	n/a		
Prerequisites:	n/a				
Course Aims:	To introduce modern machine learning techniques, and to develop the theoretical background in learning methods and to give practical experience in developing computer learning systems. To understand a variety of applications such as image recognition and face recognition.				
Learning Outcomes:	By the end of this course a student should be able to: • demonstrate knowledge of the theoretical background in machine learning methods • demonstrate an understanding of how to solve problems involving billions of parameters • describe the successful application of the methods used in image recognition, face recognition, and other areas				
Course Description:	*Introduction to Machine Learning:* basic ideas of learning theory; and applications. *Support Vector Machines:* notion of a learning machine; pattern recognition. *Kernel techniques:* Lagrangian methods of constrained optimization; kernel techniques in pattern recognition; kernel techniques in regression estimation; kernels as a general approach to solving extremely high-dimensional learning problems; application to the ridge regression procedure. *Induction and Transduction:* different types of inference. *On-line and off-line learning*.				
Teaching and Learning:	Lecture based delivery. Normally 3 hours of lectures per week.				
Key Bibliography:	Vladimir N. Vapnik: Statistical Learning Theory, Wiley, 1997: ISBN-10: 0471030031 John Shawe-Taylor and Nello Cristianini: Kernel Methods for Pattern Analysis, Cambridge University Press, 2004: ISBN-10: 0521813972				
Assessment:	**Formative Assessment & Feedback:** Return of grades and written comments on assessed coursework **Summative Assessment:** Exam (90%) 2 hours. Answer 3 out of 5 questions. No calculators. Coursework (10%) 1 assignment				
				Last updated:	17/02/10 JCW

Course Code:	**CS3930**	**Availability**:	Term 2	**Unit Value**:	0.5
Course Title:	**Computational Finance**				
Level:	3	**Admin. Fees:**	n/a		
Prerequisites:	University level mathematics				
Course Aims:	To understand the key role played by the advent of derivatives, financial instruments which facilitate managing financial risks. To understand pricing derivatives (and associated strategies of dynamic hedging) using advanced computational models are required.				
Learning Outcomes:	By the end of this course a student should be able to: • demonstrate an understanding of mathematical and computational models of underlying and derivative securities; • master techniques for pricing derivatives and for dynamic hedging; • apply these models and techniques for creating computer programs.				
Course Description:	*Introduction:* financial markets; the rules of the game. *Taxonomy of securities:* main kinds of derivative securities and underlying markets. *Mathematical techniques:* Wiener process; diffusion processes as mathematical models of price dynamics; stochastic differential equations; computer simulations. *Pricing and hedging in the Black-Scholes world:* risk-neutral valuation; the Black-Scholes equation and analytic formulae; the “Greeks" and their use. *Beyond the Black-Scholes world:* application issues; computational models; fractals and their use in finance. *Efficient markets hypothesis:* theory vs empirical evidence. *Risk management:* Value at Risk. *Coursework Project:* implementing valuation algorithms for different derivatives (e.g., in MatLab).				
Teaching and Learning:	Lecture based delivery, supported by practical classes. Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	John Hull: Options, Futures and Other Derivatives, 6th edition, Prentice-Hall, 2007 : ISBN-10: 0131977059 (earlier editions starting from the 2nd are acceptable) : Paul Wilmott: Paul Wilmott Introduces Quantitative Finance, John Wiley, 2001: ISBN-10: 0471498629 Jeff Dewynne: The Mathematics of Financial Derivatives: A Student Introduction, Cambridge University Press, 1995 : ISBN-10: 0521497892				
Assessment:	**Formative Assessment & Feedback:** Return of grades and written comments on coursework project **Summative Assessment:** Exam (80%) 2 hours. Answer 3 out of 5 questions. No calculators. Coursework (20%) 1 project				
				Last updated:	17/02/10 JCW

DEPARTMENT OF: Computer Science				Academic Session: Term 1	
Course Code:	CS3940	**Course Value:**	0.5	**Status:**	Option
Course Title:	Intelligent agents and multi-agent systems			**Availability:**	
Prerequisites:	CS2820, CS3750			**Recommended:**	
Co-ordinator:					
Course Staff					
Aims:	To introduce the student to the concept and design of an agent and multi-agent system, and the main applications for which they are appropriate. To introduce a contemporary platform for implementing agents and multi-agent systems.				
Learning Outcomes:	By the end of this course a student should be able to: • understand the notion of an agent and understand the characteristics of applications that lend themselves to an agent-oriented solution • understand the key issues associated with constructing agents capable of intelligent autonomous action • understand the key issues in designing societies of agents that can effectively cooperate in order to solve problems • understand the main application areas of agent-based solutions, and be able to develop a meaningful agent-based system				
Course Content:	*Introduction*: agents and objects, expert systems, distributed systems; typical application areas for agent systems. *Intelligent Agents*: abstract architectures for agents; tasks for agents, the design of intelligent agents - reasoning agents, agents as reactive systems; hybrid agents (e.g., PRS); layered agents (e.g., Interrap). *Multi-Agent Systems*: classifying multi-agent interactions - cooperative versus non-cooperative; zero-sum and other interactions; cooperation - the Prisoner's dilemma and Axelrod's experiments; interactions between self-interested agents: auctions systems; negotiation; argumentation; interaction languages and protocols: speech acts, KQML/KIF, the FIPA framework, ontologies, coordination languages; interactions between benevolent agents: cooperative distributed problem solving (CDPS), partial global planning; coherence and coordination; applications of intelligent agents and multi-agent systems.				
Teaching & Learning Methods:	Lecture based delivery, supported by tutorial sessions. Normally 3 hours of lectures per week.				
Key Bibliography:	M. Wooldridge: An Introduction to MultiAgent Systems. John Wiley & Sons, 2002: ISBN-10: 047149691X G. Weiss, editor: Multi-Agent Systems, A Modern Approach to Distributed Artificial Intelligence. The MIT Press, 1999**:** ISBN-10: 0262232030 J. Ferber: Multi-Agent Systems. Addison-Wesley, 1999: ISBN-10: 0201360489 M. Singh and M. Huhns: Readings in Agents. Morgan-Kaufmann Publishers, 1997: ISBN-10: 1558604952				
Formative Assessment & Feedback:	Verbal feedback provided through the tutorial sessions.				
Summative Assessment:	**Exam** (70%) 2 hours. Answer 3 out of 5 questions. No calculators. **Coursework** (30%) Assignment 1 10%, Assignment 2 20% **Deadlines: As shown on departmental coursework deadlines sheet**				

Course Code:	**IY3770**	**Availability**:	Runs in term 1 but only available to students studying for the full year.	**Unit Value**:	0.5
Course Title:	**Trusted Computing Platforms**				
Level:	3	**Admin. Fees:**	n/a		
Prerequisites:	IY2760, CS2850 or equivalent				
Course Aims:	To provide a basic introduction to trusted computing. To describe the basic hardware and software technologies underlying trusted computing. To give an appreciation of how virtualisation and isolation, with appropriate hardware support, can provide a trusted environment in which to run sensitive applications.				
Learning Outcomes:	By the end of this course a student should be able to: • understand the fundamental principles behind trusted computing • work with the TCG specifications • have an appreciation of the range of possible trusted hardware and software • understand what trusted computing can do to improve system and user security • analyse and assess possible applications of trusted computing				
Course Description:	*Introduction:* What is trusted computing? An introduction to the fundamental ideas underlying trusted computing technology. The Trusted Computing Group (TCG) specifications: history, objectives, the v1.2 specifications, and profiles for various platform types. This will include a discussion of both the hardware base for the TCG specifications, namely the TPM, and also the supporting software (TSS etc.) *Trusted computing hardware architectures*: the Intel ARM and AMD processor extensions, AEGIS, XOM *Software architectures*: exploiting trusted computing hardware, including coverage of isolation kernels, virtualisation, secure boot *Examples of TCG software architectures:* NGSCB, Xen and L4. *Applications*: application security support through trusted computing, content protection, mobile device security functions, other selected applications.				
Teaching and Learning:	Lecture based delivery, Normally 3 hours of lectures per week.				
Key Bibliography:	C. J. Mitchell (ed.): Trusted Computing, IEE Press, 2005: ISBN-10: 0863415253				
Assessment:	**Formative Assessment & Feedback:** Return of marked zero-weighted assignments. **Summative Assessment:** Exam (100%) 2 hours. Answer 3 out of 5 questions. No calculators. Coursework (0%) 4 assignments 0%.				
				Last updated:	17/02/10 JCW

Course Code:	**IY3780**	**Availability**:	Term 2	**Unit Value**:	0.5
Course Title:	**Secure Software Engineering**				
Level:	3	**Admin. Fees:**	n/a		
Prerequisites:	IY2760, CS2850 or equivalent				
Course Aims:	To identify the vulnerabilities that can be introduced into programs through language features and poor programming practice. To discuss the generic techniques that can be applied to improve the security of programs and applications. To consider the specific support provided for developing secure applications in the .NET Framework and Java.				
Learning Outcomes:	By the end of this course a student should be able to: • explain the importance of security in the development of applications, particularly in the context of distributed software and web services • be able to identify poor programming practice • appreciate the support for secure software development that has been made available to programmers in the .NET Framework and Java				
Course Description:	Background • Vulnerabilities and attacks • Countermeasures • Mobile code • Case study: Java Technology • Case Study: The .NET Framework • Web Application Programming • Web Services Security • Deployment and Configuration				
Teaching and Learning:	Lecture based delivery, supported by laboratory classes. Normally 3 hours of lectures and laboratory classes per week.				
Key Bibliography:	To be confirmed				
Assessment:	**Formative Assessment & Feedback:** Return of marked zero-weighted assignments. **Summative Assessment:** Exam (100%) 2 hours. Answer 3 out of 5 questions. No calculators. Coursework (0%) 4 assignments 0%.				
				Last updated:	17/02/10 JCW