Programme Specification

Bsc Computer Science

1. Programme title	Computer Science
2. Awarding institution	Middlesex University
3. Teaching institution	Middlesex University
4. Programme accredited by	N/A
5. Final qualification	BSc Honours
6. Academic year	
7. Language of study	English
8. Mode of study	Full Time/Thick Sandwich

9. Criteria for admission to the programme

Entry requirements are in accordance with the University regulations. We accept students from a range of backgrounds. Most students educated in the UK will have studied A-levels, AVCEs or an accredited Access Course. To enter a degree programme you would be expected to have achieved a specified number of UCAS tariff points, agreed annually and made available via the University Admissions web site or on application.

All candidates should normally possess at least grade C in GCSE maths and English Language, or equivalent.

Mature applicants with relevant work experience are also welcome to apply.

International students who have not been taught in English must show evidence of proven ability in English such as TOEFL grade 550 or IELTS grade 6.0. The University provides pre-sessional English language courses throughout the year for candidates who do not meet the English requirements. For further information, visit the learning resources web sit at: http://www.lr.mdx.ac.uk/lang/index/html.

University policies supporting students with disabilities apply, as described in the University

Regulations, ‘Information for students with disabilities’.

10. Aims of the programme

The programme aims to:

- Provide students with a thorough grounding in the practical and theoretical fundamentals of Computer Science.
- Develop knowledge and skills that are relevant to current requirements of industry.

11. Programme outcomes

A. Knowledge and understanding On completion of this programme the successful student will have knowledge and understanding of: 1. A range of underlying theories relevant to computer science. 2. The process of systems development. 3. The interaction between technology and society, and the role of computer professionals within this. 4. A range of specialised topics within Computer Science.	**Teaching/Learning Methods** Students gain knowledge and understanding through: • Lectures and tutorials. • Guided research. • Supervised lab work. • Case studies. **Assessment Methods** Students’ knowledge and understanding is assessed by: • Practical demonstrations. • Reports and essays. • Presentations. • Individual and group work. • On-line quizzes. • Unseen examinations.
B. Cognitive (thinking) skills On completion of this programme the successful student will be able to: 1. Apply analytical skills to the solution of computer-based problems. 2. Critically evaluate computer-based solutions using a range of techniques. 3. Construct abstract representations through the use of appropriate analysis and modelling techniques. 4. Apply design principles to practical problems.	**Teaching/Learning Methods** Students gain knowledge and understanding through: • Lectures and tutorials. • Guided research. • Supervised lab work. • Case studies. **Assessment Methods** Students’ knowledge and understanding is assessed by: • Practical demonstrations. • Reports and essays. • Presentations. • Individual and group work. • On-line quizzes. • Unseen examinations.

C. Practical skills On completion of the programme the successful student will be able to: 1. Apply sound principles to the construction and maintenance of computer-related artefacts. 2. Verify and validate computer-based systems. 3. Apply appropriate theories to the design and evaluation of systems. 4. Adapt and apply their knowledge and skills to mastering new technical areas.	**Teaching/Learning Methods** Students gain knowledge and understanding through: • Lectures and tutorials. • Guided research. • Supervised lab work. • Case studies. **Assessment Methods** Students' knowledge and understanding is assessed by: • Practical demonstrations. • Reports and essays. • Presentations. • Individual and group work. • On-line quizzes. • Unseen examinations.
D. Graduate Skills On completion of this programme the successful student will be able to: 1. Work effectively as a member of a software development team. 2. Communicate effectively in a variety of modes, including mathematics where appropriate. 3. Learn independently in a variety of situations, making use of available resources.	**Teaching/Learning Methods** Students gain knowledge and understanding through: • Lectures and tutorials. • Guided research. • Supervised lab work. • Specialist Workshops. • Case studies. **Assessment Methods** Students' knowledge and understanding is assessed by: • Practical demonstrations. • Reports and essays. • Presentations. • Individual and group work. • On-line quizzes. • Unseen examinations.

12. Programme structure (levels, modules, credits and progression requirement

12. 1 Overall structure of the programme

Year 1 (Level 4)	Year 2 (Level 5)	Placement Year (optional)	Final Year (Level 6)
Systems and Architecture CSD1004	Distributed Systems and Networking CSD2600		Option 1
Programming CSD1000	Web Applications and Databases CSD2550		Option 2
Foundations of Computer Science CSD1002	Software Development		Option 3
First Year Projects CSD1001	and Projects CSD2221		Final Year Computer Science Project CSD3999

12.2 Levels and modules

Starting in academic year 2010/11 the University is changing the way it references modules to state the level of study in which these are delivered. This is to comply with the national Framework for Higher Education Qualifications. This implementation will be a gradual process whilst records are updated. Therefore the old coding is bracketed below.

Level 4 (1)

COMPULSORY	OPTIONAL	PROGRESSION REQUIREMENTS
Students must take all of the following: 1. Systems and Architecture. CSD1004 2. Programming. CSD1000 3. Foundations of Computer Science. CSD1002 4. First Year Projects. CSD1001		Students are normally expected to achieve 120 credits at level 4 to progress to level 5.
Level 5 (2)		
COMPULSORY	OPTIONAL	PROGRESSION REQUIREMENTS
Students must take all of the following: 1. Distributed Systems and Networking. CSD2600 2. Web Applications and Databases. CSD2550 3. Software Development and Projects. CSD2221		Students are normally expected to achieve 240 credits at levels 4 & 5 to progress to level 6.
Level 6 (3)		
COMPULSORY	OPTIONAL	PROGRESSION REQUIREMENTS

Students must take all of the following: 1. Final Year Computer Science Project. CSD3999	Students must also choose at least 3 from the following list of indicative options: 1. Graphics and Visualization. CSD3340 2. Novel Interaction Technologies CSD3810 3. Quantum Information Processing, Verification, and Security. CSD3401 4. Social Network Analysis and Visual Analytics. CSD3335 5. Social Professional and Ethical Issues in Information Systems. BIS3400 6. Open Source Software. CSD3600 7. Human Factors in Design. CSD3820	Students are required to complete 360 credits to complete the programme and qualify for BSc (Hons) Computer Science.

	8. Artificial Intelligence. CSD3939 9. Correctness in Computer Systems. CSD3202 10. History and Philosophy and Computing. CSD3203 11. Computation, Algorithms and Complexity. CSD3201 12. Systems Engineering for Robotics PDE3413 13. Introduction to Ubiquitous Computing 14. Image Processing with MATLAB CSD3301 15. Advance Web Development with Big Data CSD3205 16. Teaching Computer Science in the Secondary School CSD3207	

12.3 Non-compensatable modules (note statement in 12.2 regarding FHEQ levels)	
Module level	***Module code***
6	*Final Year Computer Science Project*

13. A curriculum map relating learning outcomes to modules

See Curriculum Map attached

14. Information about assessment regulations

Assessment for the First Year is described in detail in the associated Computer Science Staff Handbook. Students will be provided with details of the First Year assessment scheme during the first week of the programme and continually thereafter. Assessment for Years 2 and 3 will be subject to the standard University module-based assessment regulations and the assessment details for each module are given in the module narratives***.***

15. Placement opportunities, requirements and support (if applicable)

Students will be encouraged to apply for placements in Year 3 of the programme. This is not compulsory, however they will be supported in terms of the searching for placements, generating CVs, interview technique. The School of Science and Technology works with a number of employers to run specific information sessions describing the opportunities and application procedures.

16. Future careers (if applicable)

Students who graduate with a good honours degree in Computer Science from Middlesex University will be well placed to follow a career path in a computer based industry or to go on to further study. Industrial careers include: IT management; software engineering; software architecture; hardware and software designer; web-developer; database management and administration.

17. Particular support for learning (if applicable)

Students will be supported throughout their programme of study in Computer Science by academic experts in the appropriate fields. In addition, students will be supported by a Learning Resource Centre that works closely with academics in order to offer the most up-to-date resources. All of the modules on Computer Science are supported by a team of Graduate Teaching Assistants and Technical Tutors who work with academic colleagues to ensure that labs are resourced, materials are available and feedback is provided. In the case of the First Year, the GTAs and TTs will be used to ensure that feedback is available throughout the lab sessions for all students without unreasonable delay.

18. JACS code (or other relevant coding system)	G400
19. Relevant QAA subject benchmark group(s)	Computing

20. Reference points

QAA Subject Benchmark – Computing.

BCS Guidelines.

ACM Guidelines.

21. Other information

Please note programme specifications provide a concise summary of the main features of the programme and the learning outcomes that a typical student might reasonably be expected to achieve if s/he takes full advantage of the learning opportunities that are provided. More detailed information about the programme can be found in the student programme handbook and the University Regulations.

Curriculum map for BSc Computer Science

This section shows the highest level at which programme outcomes are to be achieved by all graduates, and maps programme learning outcomes against the modules in which they are assessed.

Programme learning outcomes

Knowledge and understanding		Practical skills	
A1	A range of underlying theories relevant to computer science.	C1	Apply sound principles to the construction and maintenance of computer-related artifacts.
A2	The process of systems development.	C2	Verify and validate computer-related systems.
A3	The interaction between technology and society and the role of computer professionals within this.	C3	Apply appropriate theories to the design and evaluation of systems.
A4	A range of specialized topics within Computer Science.	C4	Adapt and apply their knowledge and skills to mastering new technical areas.
Cognitive skills		**Graduate Skills**	
B1	Apply analytical skills to the solution of computer-based problems.	D1	Work effectively as a member of a software development team.
B2	Critically evaluate computer-based solutions using a range of techniques.	D2	Communicate effectively in a variety of modes, including mathematics where appropriate.
B3	Construct abstract representations through the use of appropriate analysis and modelling techniques.	D3	Learn independently in a variety of situations, making use of available resources.
B4	Apply design principles to practical problems.		

Programme outcomes														
A1	A2	A3	A4	B1	B2	B3	B4	C1	C2	C3	C4	D1	D2	D3
Highest level achieved by all graduates														
6	6	6	6	6	6	6	6	6	6	6	6	5	6	6

Module Title	**Module Code and Level**	**Programme outcomes**														
		A1	A2	A3	A4	B1	B2	B3	B4	C1	C2	C3	C4	D1	D2	D3
Systems and Architecture	CSD1004 (4)				✓			✓								
Programming	CSD1000 (4)		✓			✓				✓						
Foundations of CS	CSD1002 (4)	✓						✓				✓			✓	
First Year Projects	CSD1001 (4)			✓		✓	✓		✓		✓		✓	✓	✓	✓
Distributed Systems and Networking	CSD2600 (5)	✓			✓			✓			✓					
Web Applications and Databases	CSD2550 (5)			✓	✓				✓	✓			✓			
Software Development and Projects	CSD2221 (5)		✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Graphics and Visualization	CSD3340 (6)	✓			✓								✓			
Novel Interaction Technologies	CSD3810 (6)	✓			✓								✓			
Quantum Information Processing, Verification and Security	CSD3401 (6)	✓			✓								✓			
Social Network Analysis and Visual Analytics	CSD3335 (6)	✓			✓								✓			
Social Professional and Ethical Issues in Information Systems	BIS3400 (6)	✓			✓								✓			
Open Source Software	CSD3600 (6)	✓			✓								✓			
Human Factors in Design	CSD3820 (6)	✓			✓								✓			
Artificial Intelligence	CSD3939 (6)	✓			✓								✓			
Final Year CS Project	CSD3999 (6)		✓	✓		✓	✓	✓	✓	✓	✓	✓			✓	✓
Correctness in Computer Systems	CSD3202 (6)	✓			✓								✓			
History & Philosophy of Computing	CSD3203 (6)	✓			✓								✓			
Computation, Algorithms, andComplexity	CSD3201(6)	✓			✓								✓			
Systems Engineering for Robotics	PDE3413 (6)	✓			✓								✓			
Introduction to Ubiquitous Computing	CSD3302 (6)	✓			✓								✓			
Image Processing with MATLAB	CSD3301 (6)	✓			✓	✓	✓						✓	✓	✓	✓

Advanced Web Development with Big Data	CSD3205 (6)	✓	✓	✓		✓	✓					✓	✓	✓		✓
Teaching Computer Science in the Secondary School	CSD3207 (6)			✓									✓			

Module Narratives

In this section you will find details of all the modules associated with your programme so that you can see what is involved in your programme and make any choices over option modules (if applicable).

The narratives are correct at the time of this handbook went to print, but details change over time (especially reading lists) and therefore you should always refer to the latest version available on the My Study area of myUniHub:
https://myunihub.mdx.ac.uk/web/home-community/mystudy

Module Code	**CSD1004**
Module Title	**Systems and Architecture**
Credit	**30**

Aims

To provide students with an understanding of the organization of computer-based systems from the small-scale (for example gates, processors and memory), through the medium-scale (for example personal computer organization), to the large-scale (for example the Internet).

Learning Outcomes

On completing this module a successful student will be able to:

Knowledge:

- Describe a range of typical computer based systems in terms of their internal organization of components and their system interfaces.

Skills:

- Construct, test and analyse systems involving both hardware and software.

Syllabus

- Data representation: binary; structures.
- Circuits: logic gates; arithmetic.
- Machine organization: memory; registers.
- Programming: instructions; interrupts; low-level vs. high-level. I
- nput-output: devices; interfaces.
- Distributed Systems: networking; communication; concurrency.
- Operating Systems.

Learning, Teaching and Assessment Strategy

This module is taught in an integrated fashion concurrently with all other first-year modules. Each first-year module contributes individual learning outcomes that are demonstrated by students typically through practical sessions throughout the first-year.

The opportunities for students to demonstrate their learning for assessment purposes are continuous and integrated with all other first-year modules. There will be multiple opportunities for students to demonstrate each learning outcome.

Each new module cohort will provided with an assessment handbook detailing the specific requirements for that year.

Assessment Weighting

This module is assessed exclusively through a student profiling mechanism as described above. The module is assessed as pass or fail based on a student's profile.

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD1000**
Module Title	**Programming**
Credit	**30**

Aims
To provide students with an understanding of the foundations of programming.

Learning Outcomes

On completing this module a successful student will be able to:

Knowledge

1. Understand the key foundational concepts used to construct, analyse and debug programs.

Skills

2. Construct, analyse and debug small-to medium sized programs for a variety of computing applications.

Syllabus

- Definitions, names, variables, interfaces.
- Data structures, lists, functions.
- Syntax, value, denotation.
- Recursive definitions, processing recursive data structures, control structures.
- Debugging, step-by-step execution.

Learning, Teaching and Assessment Strategy
This module is taught in an integrated fashion concurrently with all other first-year modules. Each first-year module contributes individual learning outcomes that are demonstrated by students typically through practical sessions throughout the first-year. The opportunities for students to demonstrate their learning for assessment purposes are continuous and integrated with all other first-year modules. There will be multiple opportunities for students to demonstrate each learning outcome.

Each new module cohort will provided with an assessment handbook detailing the specific requirements for that year.

Assessment Weighting

This module is assessed exclusively through a student profiling mechanism as described above. The module is assessed as pass or fail based on a student’s profile.

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD1002**
Module Title	**Foundations of Computer Science**
Credit	**30**

Aims

To provide students with an understanding of the foundations of Computer Science.

Learning Outcomes

On completing this module a successful student will be able to:

Knowledge:

1. Understand the key concepts underpinning the discipline of Computer Science.

Skills:

2. Be able to apply key concepts to computer systems development.

Syllabus

- Execution: state transitions; sequencing; recursion; concurrency.
- Algebraic structures: sets; graphs; sequences; trees; functions; relations.
- Logic: predicate logic; propositional logic; reasoning and proof.
- Languages: regular expressions; automata; syntax; semantics.
- Specification: ADT; invariants; pre-post conditions.

Learning, Teaching and Assessment Strategy

This module is taught in an integrated fashion concurrently with all other first-year modules. Each first-year module contributes individual learning outcomes that are demonstrated by students typically through practical sessions throughout the first-year. The opportunities for students to demonstrate their learning for assessment purposes are continuous and integrated with all other first-year modules. There will be multiple opportunities for students to demonstrate each learning outcome.

Each new module cohort will provided with an assessment handbook detailing the specific requirements for that year.

Assessment Weighting

This module is assessed exclusively through a student profiling mechanism as described above. The module is assessed as pass or fail based on a student’s profile.

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD1001**
Module Title	**First Year Project**
Credit	**30**

Aims
To provide students with experience of a range of Computer Science projects that reinforce First Year teaching and learning and provide opportunities to apply this learning in a supported and structured environment.

Learning Outcomes

On completing this module a successful student will be able to:

Knowledge:

- An appreciation of how Computer Science concepts provide a framework for understanding real-world systems.

Skills:

- To be able to successfully undertake simple development projects.
- Successfully demonstrate appropriate information searching, critical evaluation, writing and communication skills necessary to enable effective documentation and communication.

Syllabus

- Individual and group project management.
- Presentation and documentation skills.
- Professional and ethical responsibilities.
- Requirements, specification and verification.
- Design and computational thinking.

Learning, Teaching and Assessment Strategy

This module is taught in an integrated fashion concurrently with all other first-year modules. Each first-year module contributes individual learning outcomes that are demonstrated by students typically through practical sessions throughout the first-year. The opportunities for students to demonstrate their learning for assessment purposes are continuous and integrated with all other first-year modules. There will be multiple opportunities for students to demonstrate each learning outcome.

Each new module cohort will provided with an assessment handbook detailing the specific requirements for that year.

Assessment Weighting

This module is assessed exclusively through a student profiling mechanism as described above. The module is assessed as pass or fail based on a student’s profile.

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD2600**
Module Title	**Distributed Systems and Networking**
Credit	**30**

Aims
The module provides a firm basis for planning, programming and running distributed systems: how to design communicating processes using simple specification and graphical animation and how to program concurrent programs. Learners on satisfactory completion of the module will have knowledge of the main concepts of distributed systems like processes, communication, and synchronization and their central properties like fault tolerance and security. They will understand typical problems with distributed applications like deadlocks and mutual exclusion and will know the principles of networks and how distributed systems run on them.

Learning Outcomes

On completion of this module students will be able to:

Knowledge:

1. Analyse distributed applications, identify concurrency issues, and devise practical solutions

Skills:

2. Design solutions as labelled transition systems, textualise them as finite state processes, demonstrate them and implement them in Erlang on a Network.

Syllabus

- Processes and Threads
- Concurrent Execution
- Shared Objects & Mutual Exclusion
- Synchronization and Monitors
- Deadlock Safety and Liveness Properties
- Model-based Design
- Communication between remote objects
- Distributed Object Communication
- Networking Principles

Learning, Teaching and Assessment Strategy

All learning materials will be presented by using suitable multimedia presentation, such as power point presentation. There are weekly one-hour lectures. Two-hour weekly seminar and two hours lab will alternate to enforce topics covered in lectures and to encourage both individual and group activity. Discussion, weekly activities, and project work will underpin the achievement of the relevant learning outcomes. The project report needs to set out the ethical/legal/professional issues issues related to the project (raised by the tech and the context of application) and how they have been addressed in the project.

Assessment Weighting

Coursework (no examination): 100 %

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD2550**
Module Title	**Web Applications and Databases**
Credit	**30**

Aims

There are now a wide range of programming technologies associated with the Web creating a more interactive experience with cross platform capabilities. These applications allow access to backend resources from databases to media content. As the internet has increased in speed and accessibility, the software technologies have been developed to enable dynamic applications to be deployed. This module provides experience in the development of such software artefacts and an understanding of the technologies and model behind the Web.

Learning Outcomes

Knowledge:

1. Understand key technologies for creating Web applications.
2. Analyse and design a client/server Web project and evaluate between alternative technological solutions.
3. Critically evaluate the security requirements and potential vulnerabilities of a Web application.

Skills:

4. Manage the installation of tools and support software suitable for a Web project's development and final deployment.
5. Implement scripts on both the client and server side capable of communicating with databases and other information storage and exchange system.
6. Create secure applications that are accessible and usable.

Syllabus

- Client-Server architecture and technology
- Client-side scripting
- Server-side programming and scripting
- Web databases
- Security issues

Learning, Teaching and Assessment Strategy

The lecture of one hour covers the theory of Web application development while weekly directed study periods of two hours build practical skills and experience which occur both in labs and private study. Online materials such as tutorials, resources and test materials guide the study periods. This material will both reinforce lecture materials and extend the student's knowledge in stages. Students will complete weekly online journal entries based on their guided study activities and assignments. It is these journals that form the basis of the assessment.

The work produced needs to set out and be aware of, related ethical/legal and professional issues which are raised by the usage of web technologies and the context in which they are set. This awareness should be shown in both written work, how the student conducts themselves and as software developers the functionality of applications produced.

Module learning outcomes are assessed with individual assessment based on the journal (blog) (learning outcomes 2, 4, 5 and 6) and practical exercises (learning outcomes 1, 2 and 3).

Assessment Weighting

Coursework (no examination): 100%

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD2221**
Module Title	**Software Development and Projects**
Credit	**60**

Aims
This module provides the core notions required to develop, with confidence, software for real applications that students are likely to meet in their placement year. The module covers the whole software development process from requirements elicitation, to analysis and design, development as testing, documentation, maintenance, and quality assurance. In addition, through group work, this module offers students the opportunity to develop a range of professional skills needed to work successfully as a member of a project team. Students will learn to appreciate the importance of ethical, legal, organisational and environmental issues and business principles. Upon completion of this module the students will have practical knowledge of a number of tools to support all the steps involved in software development, project planning and project management. The students will be able to assess the suitability of a range of solutions to contribute effectively to the planning, development, and evaluation of software systems.

Learning Outcomes

Knowledge

- Understand and apply appropriate strategies, tools and technologies to analyse, plan, implement and evaluate a software development project.
- Critically evaluate the relevance of a project to current and future needs, taking into account economic, ethical, social, environmental, and business issues.
- Exercise judgement to ensure successful project outcomes within the constraints of professional practice.

Skills

- Make effective use of development tools (modelling tools, IDEs, versioning systems, testing platforms) to contribute to the development of software projects.
- Work effectively as an individual and as a member of a group

Successfully demonstrate a comprehensive application of information searching, critical evaluation, writing, and communication skills to enable effective documentation and communication

Syllabus

- Software development methods
- Requirements engineering
- The Unified Modelling Language
- Software development and management tools, including versioning systems
- Testing (techniques and tools)
- Software architectures
- Software documentation
- Research methods
- Software project management and evaluation techniques and principles
- Ethical and moral principles and issues
- Professionalism, including the British Computer Society's Codes of Conduct
- Legislation, including Intellectual Property Rights and legislation Disability Discrimination Act, Human Rights Act (Article 10), Data Protection Act, Human Rights Act (Art. 8) and Freedom of Information Act; computer and network abuse.
- Business models and management methods

Learning, Teaching and Assessment Strategy

The module will be structured into a series of teaching blocks that reflect distinct projects. Weekly classes will comprise a main lecture (a double lecture), a ‘projects’ laboratory (2 hours) and ‘support’ workshop (2 hours). The module will be assessed via a series of coursework components throughout the duration of the course (two of which will be group-based assessments). These will comprise a range of assessment techniques, including in-lab tests, group planning and design tasks, software implementation, and software system documentation. An overall assessment grade will be calculated on a combined total percentage for the assessments.

Assessment Weighting
Coursework (no examination): 100%

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3340**
Module Title	**Computer Graphics**
Credit	**30**

Aims

The aim of this module is to examine in depth the concepts and techniques needed in the construction of interactive graphics and visualisation systems covering advanced graphics programming techniques. It will cover theory and mathematics as required and it aims to provide students with practical experience via significant individual project work developing 2D and 3D programs using an industry standard environment.

Learning Outcomes

Knowledge:

1. Demonstrate understanding of a wide range of techniques and algorithms used in implementation of 3D graphics and visualisation systems.
2. Demonstrate in depth knowledge of an industry standard development environment including the appropriate use of libraries.
3. Demonstrate understanding of mathematical methods used in the domain of graphics and visualisation including the use of vectors, matrices and transformations.

Skills:

4. Use an industry standard graphics and visualisation development library and tools to develop an individually researched and designed graphics and visualisation application.
5. Manage resources and time in the design and development of a substantial individual coding project.
6. Document the design and development of a significant software development project.

Syllabus

- 2D transformations, vectors and matrices
- Introduction to an industry standard graphics environment
- 2D graphics systems development
- 3D transformations
- 3D graphics systems development

- Advanced 3D topics e.g. visualisation, virtual and augmented reality, video games, serious games, etc.

Learning, Teaching and Assessment Strategy

A weekly lecture will be used to deliver theoretical material, cover the necessary mathematical background, to introduce the programming environment and to discuss example programs. The laboratory time will be used for the development of course works and as an opportunity for students to get continual formative feedback on their work. Due to the development focus of the course it is assessed by 100% coursework, over three coursework assessments.

- A low level graphics programming assignment (25% of the total mark) which will cover learning outcomes 1, 2, and 3.
- A topical programming assignment which will be a preparation for the third assignment (worth 25% of the total mark) which will cover learning outcomes 1, 2, and 3.
- A major design and programming assignment, the design and development of a 3D graphics and visualisation system, worth 50% of the total course mark which covers the assessment of all learning outcomes from 1 through 6.

Assessment Weighting

Coursework (no examination): 100%

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3810**
Module Title	**Novel Interaction Technologies**
Credit	**30**

Aims

Interactive technologies are developing continually, and new devices that offer novel ways of interacting with computer-based systems are constantly finding their way into our homes, workplaces and lives. Students on this module will encounter and study a range of innovative and emerging interaction technologies. The module affords an opportunity to become familiar with the technologies and devices themselves as well as ways of analysing their applicability for particular uses and situations, and approaches evaluating their use.

By understanding how computing devices and products are used and studying the ways that usage changes over time, students will gain a critical awareness of the processes by which interactive products gain in popularity and become successful. After completing the module, students will therefore be better equipped to anticipate and select the successful interaction technologies of the future, analyse situations of use and potential users, design using the latest interaction technology, and evaluate novel and innovative designs.

Learning Outcomes

Knowledge:

1. Describe the underpinning concepts and key features of a range of novel and emerging interaction technologies
2. Critically assess and compare such technologies
3. Provide arguments for how such technologies challenge concepts of HCI

Skills:

4. Analyse problem situations with regard to task, context and user needs
5. Design and prototype solutions that effectively harness the potential of novel interaction devices and technologies to address specific problems
6. Plan and conduct appropriate evaluations of designed artefacts, and use such evaluations in the iterative improvement of designs

Syllabus

- Fundamentals of HCI: Usability, User Experience, and the Evaluation of Novel Interaction Technologies
- Tangible User Interfaces
- Augmented Reality & Virtual Reality
- Theory: Reality-based interaction
- Haptic Perception and Interaction
- Haptic Assistive interactive technologies
- Novel Assistive Technologies
- Pervasive and Ubiquitous Computing

Learning, Teaching and Assessment Strategy

Student’s learning will be assessed through a combination of presentations of practical work, reports based on literature or practical work, and examinations to assess conceptual knowledge.

The practical nature of the topic, and the way that theory and conceptual machinery can appropriately be articulated in the context of practice, is reflected exam/coursework weighting.

Assessment Weighting

Unseen examination: 40% Coursework (no examination): 60 %

Exam Duration

Examination, 2 hours

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3401**
Module Title	**Quantum Information Processing, Verification, and Security**
Credit	**30**

Aims

The aim of this module is to develop an appreciation and understanding of system modelling and analysis techniques in three different, yet related areas in computer science.

Quantum Information Processing is the study of the relation and applications of quantum mechanics to information processing. It is an exciting field at the intersection of Quantum Physics and Computer Science. It provides a novel computational paradigm enabling the implementation of algorithms that outperform their classical counterparts. It also allows the development of cryptographic protocols, which can be unconditionally secure. This topic provides students with an opportunity to study some of these techniques, evaluate their effectiveness and make the comparison between classical and quantum information processing.

Model checking is a prominent formal verification technique for assessing functional properties of information and communication systems. It is a general approach and is widely applied in areas such as hardware verification and software engineering. This topic provides students with an opportunity to apply model checking techniques in a practical setting, evaluate their effectiveness and prepare them for industry or further study.

Security is a broad subject that covers many disciplines across the board from social sciences, and formal methods through to networking. In the last part of this module, we introduce basic notions of security and security engineering as well as cryptographic protocols. We study the application of quantum information processing to security by introducing quantum key distribution as well as the application of model checking to quantum security protocols. We thus unify the subjects of this module.

Learning Outcomes

On completion of this module, the successful student will be able to:

Knowledge:

1. Understand the basic concepts of quantum information processing.
2. Understand the differences between classical and quantum computation.
3. Be familiar with some key quantum algorithms and protocols.
4. Appreciate the capabilities and limitations of quantum computation.

5. Understand the basic concepts of verification, specifically model checking.
6. Be familiar with basic modelling and specification methods.
7. Be familiar with basic model checking algorithms and data structures.
8. Be able to apply model checking techniques to simple examples.
9. Understand basic notions of security (CIA) and cryptography.
10. Be familiar with standard authentication protocol descriptions.
11. Be familiar with basic security modelling techniques.
12. Understand the use of verification for security and its limitations.

Skills:

13. Apply operations to quantum states and calculate the effect of measurement.
14. Design and analyse simple quantum circuits.
15. Use tools to simulate quantum algorithms and protocols.
16. Know how to model computer systems and specify properties
17. Know basic definitions of transitions systems and temporal logics
18. Use tools to perform model checking of simple examples
19. Know how to understand a basic security scenario
20. Know how to read a protocol description and understand implicit security assumptions
21. Use mechanized verification to model a security protocol and check for attacks.

Syllabus

- Introduction to quantum bits
- Quantum gates and circuits
- Quantum Parallelism and Deutsch’s algorithm
- Quantum Cryptography and Communication (Quantum Bit Commitment, Quantum Key Distribution)
- Tools for Simulation of Quantum Algorithms and Protocols
- Introduction to Formal Verification
- Kripke Structures
- The Temporal Logic CTL
- Model Checking algorithm for CTL and associated tools
- Security models
- Authentication protocols
- Verification of Security protocols

Learning, Teaching and Assessment Strategy

The majority of the factual material is covered in a weekly one hour lecture, which will be in a traditional lecture format. Skills and experience are built up through weekly one and a half hour seminar and laboratory sessions, which will vary between interactive seminar sessions involving class discussion and individual or group exercises, and experimenting with demos. The student will be expected to pursue current issues more deeply through self-study and group discussions.

The formative assessment of this module involves a combination of group work and individual work. The laboratory based practical ability is assessed through group work.

Students will be required to submit coursework, which should critically evaluate the techniques used as well as the results obtained (Outcomes 2, 4 and 7).

Typically, the outcomes of, and feedback on, assessment activities on this module will enable students to record in the PDP portfolio their progress in developing effective learning and communications skills.

Assessment Weighting

Coursework (no examination): 100 %

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3335**
Module Title	**Social Network Analysis and Visual Analytics**
Credit	**30**

Aims

To learn the underlying theories for Social Network Analysis (SNA) and Visual Analytics (VA) and apply this theory to produce analysis on Social Media, such as Consumer Generated Websites, Information in Organisations, e.g. email, and information related to criminal and digital investigations.

Learning Outcomes

Knowledge:

1. Demonstrate an understanding of fundamental mathematics and related theory of SNA and VA via reports.
2. Differentiate between Clique Relaxation techniques and how the results vary.
3. Differentiate between different Centrality metrics and how the results vary.
4. Knowing when to use certain measure to present information clearly.
5. Provide analysis of detailed and complex data.
6. Knowledge of the theory and principle of combining computation and visualisation for analysis of large data set.
7. Knowledge of common information visualisation techniques for different data types.
8. Knowledge of popular visual analytics tools and their limitations.

Skills:

9. Identifying appropriate software to apply the theory above.
10. Displaying and reporting data in clear, succinct and scientific way.
11. Using software to generate data from Consumer Generated Websites and other media.
12. Ability to apply theory and deploy appropriate software to represent and visualise data.

Syllabus

- Displaying and visualising data from Social Networks
- Analysis of Social Networks

- Analysis of Social Media
- Fundamentals of Graph Structure
- Various representations of Graphs
- Graph Theory and discrete mathematics
- Clique Relaxation Techniques
- Centrality measures.
- Information visualisation theory and model
- Common visualisation techniques
- Popular visual analytics software

Learning, Teaching and Assessment Strategy

An unseen written exam (40%) testing learning outcomes [1-3].

Two in-course assignments (2*30%) testing learning outcomes [4-9].

Coursework 1 will analyse social media by applied theory using tools and techniques to visualise data appropriately.

Coursework 2 will analyse a given problem by applied theory using tools and techniques to visualise data appropriately.

All assessment to be attempted.

Assessment Weighting
Unseen examination: 40% Coursework (no examination): 60%

Exam Duration
Examination, 3 hours

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**BIS3400**
Module Title	**Social, Professional and Ethical Issues in Information Systems**
Credit	**30**

Aims
With the increasing use of information systems in all areas of life it is increasingly important for today's graduates – who are tomorrow's professionals - to understand and appreciate the ethical implications and social impact of current technologies, to have a working knowledge of the legislation that applies in this area, and to apply their expertise in a professional way.

This module encourages students to develop an awareness of their role in the implementation of new technologies, and the knowledge and skills necessary for a professional approach. The module will take an ethical perspective to computer technology, focussing on UK legislation and standards as they relate to IS practice (for example Intellectual Property Rights in web design, database systems etc.), and will include considerations for design and the responsibilities and requirements of the IT profession – for example, as identified in The British Computer Society Code of Conduct:

- Promote equal access to the benefits of IS by all groups in society
- Have regard for the legitimate rights of third parties
- Promote public understanding of IS – its benefits and pitfalls
- Have knowledge and understanding of relevant legislation, regulations and standards.

Learning Outcomes

Knowledge & Understanding

1. Identify ethical, social and legal issues relevant to Information Systems within a context of application (e.g. information systems management, project management, security management).
2. Assess and evaluate key aspects of legal, ethical and cultural issues relevant to an IT professional.
3. Distinguish opposing perspectives and authority of reference material.

Skills

1. Apply professional principles to an application domain of current and emerging technologies.
2. Recognise the limits and vulnerabilities of information systems.

3. Evaluate and select the appropriate conceptual tools available to an information systems practitioner.

Syllabus

• Professional issues: Professional ethics and standards, Codes of Conduct, legislation, professional obligations.

• Ethical, Social and cultural issues relevant to the design, development, implementation and management of information systems.

• Ethical, social and legal issues relevant to the prevention of information system misuse and abuse.

Learning, Teaching and Assessment Strategy

Learning and Teaching
Home students

Present through lectures concepts and theories prevalent in the field, enhanced with reference to application domains and contemporary examples. (1 hour per week)

Seminar sessions will be used to apply the concepts and theories in context specific cases, based on discussion and group work. (2 hours per week)

Directed reading and research tasks will underpin the material given in Lectures and seminars.

Assessment Strategy

A formative assessment strategy will be used, based on individual reflection of the material delivered and researched, and incorporating the outcomes of group discussions in seminars. Direction will be given to students and milestones set where work will be assessed. (LO’s 1,2, 3.) Milestone 1: 1500 word essay, submitted initially to ‘turnitin’ available on Unihub (to aid students’ understanding regarding plagiarism), and reviewed by tutors. Milestone 2: student presentation of the development of work since the essay milestone, and envisaged work to be undertaken for the final piece of work. Thus the work presented at milestones will be developed in a cumulative way leading to a final piece of work that brings together the principles learnt, and their application in a particular context. (LO’s 4,5,6.)

Learning and Teaching
Distance learning students

Students overseas are supported by their Local Study Centre for lecture and seminar work. They will have access to online Module material in the same way as home students.

Assessment as home students.

Assessment Weighting
Coursework (no examination): 100%

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3600**
Module Title	**Open Source Software**
Credit	**30**

Aims
The module aims to give students an introduction into the Open Source software ecosystem. A range of issues involving Open Source will be discussed, both technical (the Open Source development model) and non-technical (legal, ethical and political issues). In order to gain hands-on experience, students will also participate in an existing Open Source project.

Learning Outcomes

Students who have successfully completed this module will be able to:

Knowledge

1. Demonstrate an understanding of the Open Source paradigm, its difference from traditional software development practices and its strengths and weaknesses.
2. Demonstrate an understanding of concepts important in Open Source concepts such as licensing, community management and collaborative work.
3. Understand and discuss the ethical, political and economical issues involving Open Source software.

Skills

4. Be able to participate in or set up an Open Source project, and use the Open Source development model.

Syllabus

- History of Open Source
- Definitions of Open Source
- Collaborative development model (source code management, bug tracking, communication methods)
- Licensing and legal issues
- Open Source business models

- Ethical considerations
- Open Source and politics

Learning, Teaching and Assessment Strategy

The topics enumerated above will be discussed in a weekly lecture.

Students will be required to participate in an existing project. Since the focus is not specifically on the quality of the end product, but on the interaction with the project community, this does not necessarily involve programming, but can also take the form of advocacy, adding to documentation, translation, etc. Creativity is encouraged.

Each student will be required to give a presentation on a current Open Source-related topic.

Assessment Weighting
Coursework (no examination): 100 %

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3820**
Module Title	**Human Factors in Design**
Credit	**30**

Aims
Students of this module will gain understanding of underpinning concepts and practical techniques relevant when considering humans, both in the organisation of design and design processes, and as a way of incorporating a user perspective in the design of products and services.

Learning Outcomes

Knowledge

On completion of this module, the successful learner will have a

Knowledge and understanding of:-

1. Explain key theories and concepts that underpin our understanding of human roles in the design process, both as design participants, and as other stakeholders (e.g. users of designed artefacts)

Skills

On completion of this module, the successful learner will be able to:

2. Select and apply appropriate techniques, methods, and tools to research and understand human stakeholders in design, and to deploy the knowledge so gained to achieve effective and efficient design processes and positive user experiences

Syllabus

- Design thinking
- Individual behaviour: Understanding individual cognition
- Team interaction and collaborative working
- Individuals and teams in the design process
- Understanding the user in design
- Methods: User research

- Methods: Design and prototyping
- Methods: Evaluation

Learning, Teaching and Assessment Strategy

A combination of weekly lectures and practical sessions will be used to deliver conceptual material and to allow students to develop practical skills.

Student’s learning will be assessed through a combination of presentations of practical work, reports based on literature or practical work, and examinations to assess conceptual knowledge.

The strong practical elements of the topic, and the way that theory and conceptual machinery can appropriately be articulated in the context of practice, is reflected exam/coursework weighting.

Assessment Weighting
Unseen examination - 40% Coursework - 60%

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3939**
Module Title	**Artificial Intelligence**
Credit	**30**

Aims

The aim of the module is to introduce students to a range of AI theories and techniques, including the most commonly used. This will extend to the ability to implement these techniques, and the students will extend their own development skills.

Learning Outcomes

Knowledge:

On completion of this module, the successful learner will have a knowledge and understanding of:

1. Demonstrate understanding of common knowledge representation mechanisms.
2. Demonstrate understanding of common machine learning mechanisms.

Skills:

3. Ability to implement knowledge bases in common knowledge representation formats.
4. Ability to implement machine learning algorithms for particular applications.
5. Ability to use common AI development techniques and languages.

Syllabus

- First Order Predicate Logic
- Semantic Nets
- XML
- Statistical techniques including linear approximation.
- Multi-layer perceptrons
- Self-organising maps
- Genetic algorithms
- Rule based systems
- Case base reasoning
- Search mechanisms
- Algorithms for large data sets
- AI areas including language, vision and robotics.

Learning, Teaching and Assessment Strategy

A weekly hour long lecture will deliver theoretical material, and allow students to engage with each other and the faculty. A weekly two hour laboratory session will be used for the practical implementation of knowledge bases and AI systems.

Assessment Strategy

The course works will enable the student to develop their skills with in depth experience of particular algorithms and representation techniques. A typical module run will have the student developing two course works. For example:

1. The student will develop an XML tag set, documents tagged with that set, and a system to retrieve documents with particular tag values. This coursework is due in December and worth 20% of the total mark. This will cover learning outcomes 1, 3 and 5.
2. The student will implement a MLP system, and use that system to learn to categorise yeast (a standard benchmark). This will be due in late March and be worth 30% of the total mark. This will cover learning outcomes 2, 4 and 5.

The exam will cover all five learning outcome. While the course works allow in depth coverage of particular topics, the exam will test the student's acquisition of the breadth of topics.

Assessment Weighting
Unseen examination - 50% Coursework - 50% Length of exam: 3hours

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3999**
Module Title	**Computer Science Project**
Credit	**30**

Aims

This module provides students with the opportunity to demonstrate the theoretical knowledge and practical skills they have developed whilst studying the computer science degree by undertaking a substantial piece of individual project work. The project will involve the production of a system that is expected to be of considerably greater scope than any of the coursework encountered in the taught part of the programme and demonstrates a significant level of scholarship.

Learning Outcomes

On completion of this module, the successful student will be able to:

Knowledge:

1. Apply a range of computer science technologies, theories, research, design, and evaluation techniques to the solution of a specific and substantial problem and recognise the professional, legal and ethical issues involved.

Skills:

2. Address a complex problem with a spirit of critical enquiry, effectively locating, planning, using and managing resources and time and to effectively communicate the outcomes.

3. Successfully demonstrate a sophisticated application of information searching, critical evaluation, writing and communication skills to enable effective documentation and communication for the final year project, as well as life-long personal and career development.

Syllabus

There is no taught syllabus.

Learning, Teaching and Assessment Strategy

Students are allocated a supervisory team for the duration of their project. The team is responsible for advising the student on all aspects of the project.

A number of support workshops on the various stages of the project will be provided as necessary throughout the year, on topics such as plagiarism, evaluation and testing, literature review, referencing and citations, dissertation structure and research methodologies.

Students are expected to operate as autonomous reflective learners, using the knowledge and skills gained from their studies over the previous two years.

Students will liaise regularly with their supervisory team throughout the year for the continuous provision of summative and formative feedback. Students will submit a report and agreed deliverables. Each of these stages will be assessed as below and feedback given as necessary.

Assessment Weighting
Coursework 100%, with the following deliverable components: • Project report (75%) • Project presentation (25%)

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3201**
Module Title	**Computation, Algorithms, and Complexity**
Credit	**30**

Aims

Computer Science has developed to be an academic discipline with an essential impact industrially, commercially and in everyday life. Students on this module will learn of the fundamentals and origination of computation combined with analytical (logical) skills applied to conceptual issues originating in theoretical and applied computer science. The second part of the module introduces students to algorithms through the lens of the resources they take up, such as time and space. Students will be able to compare two algorithms for the same problem and understand probable barriers to efficient algorithms through a gentle introduction to Turing Machines and completeness for non-deterministic polynomial time (NP).

Learning Outcomes

Knowledge:

- Foundations of logic and computability
- Demonstrate an appreciation for the rationale of a Turing Machine
- Articulate the use of divide-and-conquer algorithms and local search/ greedy algorithms
- Understand and critically compare different graph colouring algorithms
- Demonstrate an appreciation for different complexities, such as polynomial time (P) and NP-complete

Skills:

- Articulate the workings of a Turing Machine
- Articulate how an algorithm works
- Assess the time taken and space used by fundamental algorithms in computer science
- Explain the use of different algorithmic techniques and critically compare algorithms
- Articulate an understanding of reduction between problems
- Research results outside of the classroom and present them

Syllabus

- Computation: Enumerability, Diagonalization and Turing-Computability
- Universal Turing Machines and Uncomputability

- First-order logic
- Introduction to Algorithms and Complexity
- Time and Space measured by asymptotic notation
- Sorting algorithms: INSERTION-SORT and MERGE-SORT
- Graph (map) colouring: 2-, 3- and 4-COLOURING
- Deterministic and non-deterministic Turing Machines
- P and two definitions of NP
- Satisfiability and NP-completeness

Learning, Teaching and Assessment Strategy

The course will support a collaborative mode of learning combined with assessment of individual critical capacities. To this aim, students will be required to show both individual work and group activities. The learning process will be based on a combination of theoretical knowledge and practical skills.

The student will be required to show knowledge of basic principles of Computation, Algorithms and Complexity

Assessment Weighting

Coursework 100%

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3202**
Module Title	**Correctness in Computer Systems**
Credit	**30**

Aims

Computer systems are almost never perfect, exhibiting errors, crashes and hangs whose implications range from benign to fatal. This module will examine the different notions of correctness relevant to computer systems, and how these are applied to the different parts of a computer system. Automatic and user-guided methods that attempt to find possible problems within systems will be covered and demonstrated on practical examples. Also, methods for ensuring that no problems can possibly exist within a system design will be examined and applied.

Learning Outcomes

Knowledge:

- Gain understanding of different correctness conceptualisations including safety/liveness, functional correctness and temporal properties.
- Become familiar with the different ways of specifying correctness including pre- and post-conditions, temporal logic properties, code contracts and unit tests.
- Gain awareness of the role of correctness in industry: safety-critical industries and certification/validation processes.

Skills:

- Apply and master open source industry-standard tools for finding bugs in software.
- Abstract and model complex systems so as to make them amenable to analysis with tools such as model checkers.
- Design unit test suites manually and automatically, using test generation tools.
- Develop advanced debugging techniques and apply them to software systems.
- Apply advanced code writing techniques aiming to reduce the incidence of bugs including defensive coding and coding-by-contract.

Syllabus

1. Logic as specification language: propositional, first-order, Hoare, temporal logics.

2. States and traces as models of computations.
3. Safety vs liveness, reachability vs termination and other correctness concepts.
4. Absence of bugs vs presence, and techniques for proving those.
5. Tests, code coverage and other code metrics.
6. International and national standards for system correctness, and related processes.

Learning, Teaching and Assessment Strategy

The course will support a collaborative mode of learning combined with assessment of individual critical capacities. To this aim, students will be required to show both individual work and group activities. The learning process will be based on a combination of theoretical knowledge and practical skills.

The student will be required to show knowledge of basic principles of Computation, Algorithms and Complexity

Assessment Weighting
Coursework 100%

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**PDE3413**
Module Title	**Systems Engineering for Robotics**
Credit	**30**

Aims

- To develop understanding of the integration of modular robotic and sensor systems.
- To acquire practical skills of robotic software/hardware integration and validation.
- To develop understanding of human-robot interaction.

Learning Outcomes

Knowledge:

1. To demonstrate knowledge of multi-sensor signal processing and fusion techniques.
2. To demonstrate knowledge of human-robot interaction in a social context.
3. To demonstrate knowledge of robot architectures

Skills:

4. Be able to select and apply appropriate signal processing and fusion techniques to solve specific challenges in robotics.
5. Be able to develop suitable robot control techniques in response to human input.
6. Be able to devise a systems architecture for complex robot operations.

Syllabus

Indicative topics that may be covered:

- Robotic systems architectures and programming
- Appropriate signal processing techniques e.g. Kalman filtering
- Multi-sensor data fusion techniques, e.g. occupancy grids
- Mobile robot locomotion and navigation systems. Mapping and localisation techniques e.g. SLAM
- Interactive robotics in social contexts (interfacing, interaction and ethics)
- Physical human robot interaction (haptics and appropriate control techniques e.g. force, admittance, hybrid, impedance

- Teleoperation/telepresence, master/slave manipulators
- Distributed and swarm robotics
- Artificial Intelligence reasoning methods for robotics

Learning, Teaching and Assessment Strategy

The module will be delivered through a series of lectures and seminars, which are supported by lab demonstration and guided exercises. In addition students will perform a number of challenging mini projects in which they will work both in teams and individually.

Assessment would be entirely by coursework, which will comprise of a selection of the following types: problem-solving exercises, lab exercise involving programming, practical work, project reports. The makeup of the coursework submission and any individual weightings will be communicated to students through the documentation at the beginning of the module. The assessment will typically involve 2 pieces of assessed work which will depend on the type of technologies and hardware used on the module at the time.

Assessment Weighting
Coursework (No examination) 100%

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3203**

Module Title	**History and Philosophy of Computing**
Credit	**30**

Aims Computer Science is developing as an academic discipline with essential industrial, commercial and everyday life impact. Practitioners to be aware of the delicate and extensive role they are playing in society, including a critical and historical understanding of their subject matter. Students on this module will learn fundamentals of computing combined with analytical (logical) skills applied to conceptual issues originating in theoretical and applied computer science. The philosophy of computer science represents a new essential learning and thinking method for students of computer science courses, be they oriented towards an academic or professional career, to reflect and judge their own subject matter, assess its working methodology and foresee critical future developments. By locating computing devices in their structural and historical evolution and by learning the epistemic and ontological principles that define Computer Science, students will gain critical awareness of the processes by which computing has become an essential aspect of our lives and will understand how this subject is located with respect to other sciences.

Learning Outcomes

Knowledge:

- Logic and computability
- Philosophical Method applied to CS
- Knowledge of the literature in Philosophy of CS
- Knowledge of the historical development of CS and some original writings

Skills:

- Analyse socio-technical problems in the light of a philosophical approach
- Develop case studies by computational means: web-software, modelling tool, programming language, theorem-prover, robotic platforms
- Plan and conduct appropriate evaluations on the process: experiments and results

Syllabus

- Logic and Computation: Enumerability, Diagonalization, Turing-Computability and Uncomputability, Recursion, FOL
- Epistemology of CS: what is CS, What does CS study? Principles of scientific method
- Ontology of Computing: what is an algorithm, what is a program, what are machines, what are implementations?
- History of Computing: machines, programming languages, WEB
- Applications: socio-technical problems and implementations

Learning, Teaching and Assessment Strategy

The course will support a collaborative mode of learning combined with assessment of individual critical capacities. To this aim, students will be required to show both individual work and group activities. The learning process will be based on a combination of theoretical knowledge and practical skills.

The student will be required to show knowledge of basic principles of History and Logic of Computation, to show understanding of some basic issues in the Philosophy of Computing and develop such understanding through a practical development project.

Assessment Weighting
Coursework (No examination) 100%

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3302**
Module Title	**Introduction to Ubiquitous Computing**

Credit	**30**

Aims Ubiquitous Computing is an area of Computer Science which uses sensing technology to develop systems to support human activity. Sensors are small devices which translate physical measurements in the real world into digital information. This allows us to make decisions in real-time based on a specific current context. There is also the added value of being able to see the historical evolution of this data to detect interesting patterns. There are nowadays several examples of these systems in the market; for example, mobile phones provide sensors which allow developers to write apps to provide services to the mobile phone owner. Ubiquitous Computing is being used in large scale projects like smart homes, cars, offices, shopping malls as well as on a myriad of gadgets which being sold ranging from devices which can tell us whether our pot plant needs nutrients or a weighting scale which track our weight and gives us wellbeing tips.

These technologies have stimulated research and innovation which developed under several different but essentially closely related labels, for example, Internet of Things, Pervasive Computing, Ambient Intelligence and Intelligent Environments. The principles and technologies explored in this module is common to all those areas.

The module aims to provide an overview of the fundamental principles used in the emerging area of Ubiquitous Computing. These fundamental concepts will be illustrated with examples and small projects. Students will be requested to write essays and develop small projects proposals to show their understanding of the module.

Learning Outcomes

Knowledge

- Be familiar with the fundamental concepts underpinning Ubiquitous Computing systems.
- Be familiar with the technological infrastructure (hardware and software) used in building these systems.
- Be familiar with new trends emerging in Ubiquitous Computing.
- Demonstrate understanding of the development processes used in this area.
- Develop case studies by computational means: web-software, modelling tool, programming language, theorem-prover, robotic platforms
- Plan and conduct appropriate evaluations on the process: experiments and results

Skills

1. Being able to select solutions based on Ubiquitous systems.
2. Install and Use Ubiquitous systems

3. Being able to design new solutions based on Ubiquitous systems.
4. Present project designs to develop Ubiquitous systems to other peers.

Syllabus

- Introduction to Ubiquitous Computing
- Software Development Methodologies for Ubiquitous Systems
- Requirements for Ubiquitous Systems
- Ubiquitous Interfaces
- Ethics

Learning, Teaching and Assessment Strategy

The module will be divided into ‘blocks’ which typically will range from 4 to 6 weeks, depending on the availability of staff each year.

A summative assignment will be due at the end of each block. Each assignment will also be formative and feedback on each one will be given during the block following its submission. The number of blocks, their content, and their marks weighting will be given at the beginning of the module (during the ‘introduction to the module’ lecture given by the module leader during the first seminar of the module).

Coursework is based on the topics and skills needed for each block. Coursework specifications will be made available on UniHub (access to MOODLE) during the block to which they apply. All coursework includes a certain allocation of marks for preparation, which are achieved through participation in class. Hence missing class diminishes your chances of achieving the highest grades.

All classes will be in an interactive. Students will normally be expected to prepare for each seminar as they will be required to contribute during the seminars. Participation in seminars will be taken into account in grading coursework assignments. Because of the need for a high degree of skill in reading and writing academic material, some seminars will concentrate on these skills.

Students who do the homework well are well prepared for classes and can participate fully, which positively affects their grades given for the related coursework.

Seminars will be complemented with tutorials where students can discuss progress with their coursework and get support by the module team.

Assessment will be based in coursework. Students will have to submit one coursework for each of the blocks. It will be required that students score at least 40% in the majority of the assignments to be submitted in order to approve the module.

Assessment Weighting
Coursework (No examination) 100%

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.

Module Code	**CSD3301**
Module Title	**Image Processing with MATLAB**

Credit	**30**

Aims The advances of computer technology have led to the creation of large amount of images, in particular over the internet and in medical fields, benefiting human kind in an unprecedented way. The aim of this module is to introduce software of MATLAB to perform basic tasks of image processing, including enhancement, segmentation, and measurement. In addition, this module will have a focus on image search, classification and retrieval as well as on applications to a number of medical imaging modalities, including s as x-ray, computerised tomography (CT), magnetic resonance (MR), echocardiography (ultrasound video images) and retinal imaging.

Learning Outcomes

Knowledge:

1. Understand the importance of computer technology in contribution to the creation of images.
2. Acquire basic grasp of the software of MATLAB.
3. Get familiarised with basic image processing techniques.
4. Apprehend the complexity of the development of image search engines.

Skills:

5. Programming using MATLAB for image processing.
6. Mastering basic mathematical theories and formulas for designing image operators and filters.
7. Extracting salient features from medical image data.
8. Classifying images based on similar features, e.g. colour, shape, texture, etc.

Syllabus

- Introduction to software of MATLAB
- Introduction to medical imaging techniques (X-ray, CT, MR, PET, retinal)
- Image segmentation (edge detection using Sober, Canny operators)
- Image feature extraction (colour, shape, texture)
- Introduction to image classification (cars, flowers, sky)
- Introduction to image retrieval systems (e.g., finding white cats.)

Learning, Teaching and Assessment Strategy

Broad principles in the syllabus will be taught in the form of lectures and practiced in mini-project-based computer laboratories, where students will apply and evaluate the learned techniques in the assigned projects, in an effort to further the benefit of computer information technology to the application of medical fields. Each mini-project will be assessed, cumulating 50% coursework.

- The module is assessed by 100% coursework.
- 50% coursework will be based on a design of a classifier for classification of different images. A document of publishable quality judged by refereeing procedures used in international conferences will be requested to submit together with their MATLAB programme code detailing the reasons why this classifier is better.
- Another 50% is based on the assessment in the lab for the lab-based projects.

Assessment Weighting
Coursework (No examination) 100%

Learning Materials

Your online reading lists can be accessed from the My Study area of UniHub. They highlight essential and recommended reading for all modules you are registered on.