BSc (Hons) in Applied Software Engineering

Swansea University
Prifysgol Abertawe

Course Handbook

2023/ 2024

technocamps

institute of CODING in wales

difficilia quae pulchra

things that are excellent are difficult

This Course Handbook is intended for students of the BSc in Applied Software Engineering at Swansea University. The information contained in this Handbook is, to the best of our knowledge, correct at the time of producing. The Department retains the right to change details it contains, but every effort will be made to notify students of such changes. Where discrepancies arise, University Regulations will apply. – September 6, 2023.

Contents

The BSc in Applied Software Engineering

This Course Handbook seeks to draw together all of the information that you will need to understand as you embark on the BSc in Applied Software Engineering at Swansea University.

Swansea University is a research-led university that has been making a difference since 1920.

The Department of Computer Science sits within the College of Science and is home to world-class researchers, fine laboratories and excellent teaching programs. Friendly staff are committed to providing an excellent student experience and education of a very high standard. It is also a top research environment, containing the highest percentage of world-leading researchers in any Computer Science Department in Wales (and 11th highest in the UK) according to the most recent research assessment exercises REF 2014. This research informs our teaching and the development of new degree programmes.

The Department of Computer Science also has excellent collaborative links with local industry, through its IT Wales programme. Our knowledgeable academic staff and students are valued by companies for bringing a new perspective and insight into the latest developments in computer science, particularly within the software, IT services and telecoms sectors.

It is in this setting of world-leading expertise that the BSc in Applied Software Engineering was created. It seeks to widen higher education access to students who are in work, both in an apprenticeship style as well as to provide those in possession of substantial industry experience the opportunity to frame that experience against appropriate academic perspectives. By integrating academic and work-based learning, this degree programme serves a demand from industry, addressing head-on the recognised growing skills gap in this high-value industry.

The BSc in Applied Software Engineering has been accredited by the UK's Sector Skills Council for IT, Software, Web and Telecoms Professionals for their Degree Apprenticeship Framework Level 6 Qualification, meaning that the BSc degree you are awarded will be widely recognised and respected.

The BSc in Applied Software Engineering is not a programme of study to be taken lightly. Whilst providing the flexibility of allowing its students to "earn-and-learn", there will inevitably be times of great challenge when trying to balance work and study, with each demanding your fullest attention to meet targets and deadlines. The onus is on you to face and conquer such challenges as and when they come along.

That being said, I very much hope that your studies here will be enjoyable and transforming.

Ms Casey Hopkins
Programme Director

THE DEPARTMENT OF COMPUTER SCIENCE

Computer Science Education at Swansea University is characterised by nine Educational Aims.

EDUCATIONAL AIMS

The aims of our education in Computer Science are to provide our students with

1. practical experience and theoretical understanding of design methods for the specification, programming and analysis of a wide range of computing systems;
2. a fundamental understanding of the scope and limits of Computer Science and Artificial Intelligence, and of their applications;
3. knowledge of the history and present state of Computer Science, and an insight into future technologies and their role in applications and society;
4. the ability to plan and accomplish a substantial project;
5. relevant mathematical knowledge and experience in its applications;
6. experience in co-operative working through team projects, with their demands on the management of partners and time;
7. skills in written and oral communication;
8. skills in locating information, and the ability to read critically, to précis and to judge information; and
9. the ability and confidence to learn, unaided, complex new subjects.

The first three Aims are focused on knowledge of the subject of Computer Science, the next three are concerned with related knowledge and experience, and the last three with personal competence.

The work of the Department is not just its teaching. We are dedicated to studying and advancing Computer Science and its applications. We play our part in the international community of computer scientists. We run various programmes for industry and the community which link the skills, experience and needs of industry to the knowledge, expertise and resources of members of the Computer Science Department. To fully appreciate our educational work, it helps to understand the Department's wider mission.

MISSION

1. To accomplish outstanding research in Computer Science and its applications that is fundamental and useful.
2. To provide an excellent education in Computer Science for undergraduate and postgraduate students that is of the highest international standards.
3. To recognise and respond to the changing needs of society through specific high-quality research, teaching programmes and initiatives.

Dr Liam O'Reilly
Departmental Programme Director

The Structure of the Programme

The BSc in Applied Software Engineering is a 3-year programme where each year consists of three trimesters. Lectures and labs are delivered on campus every Wednesday from 1pm-8pm.

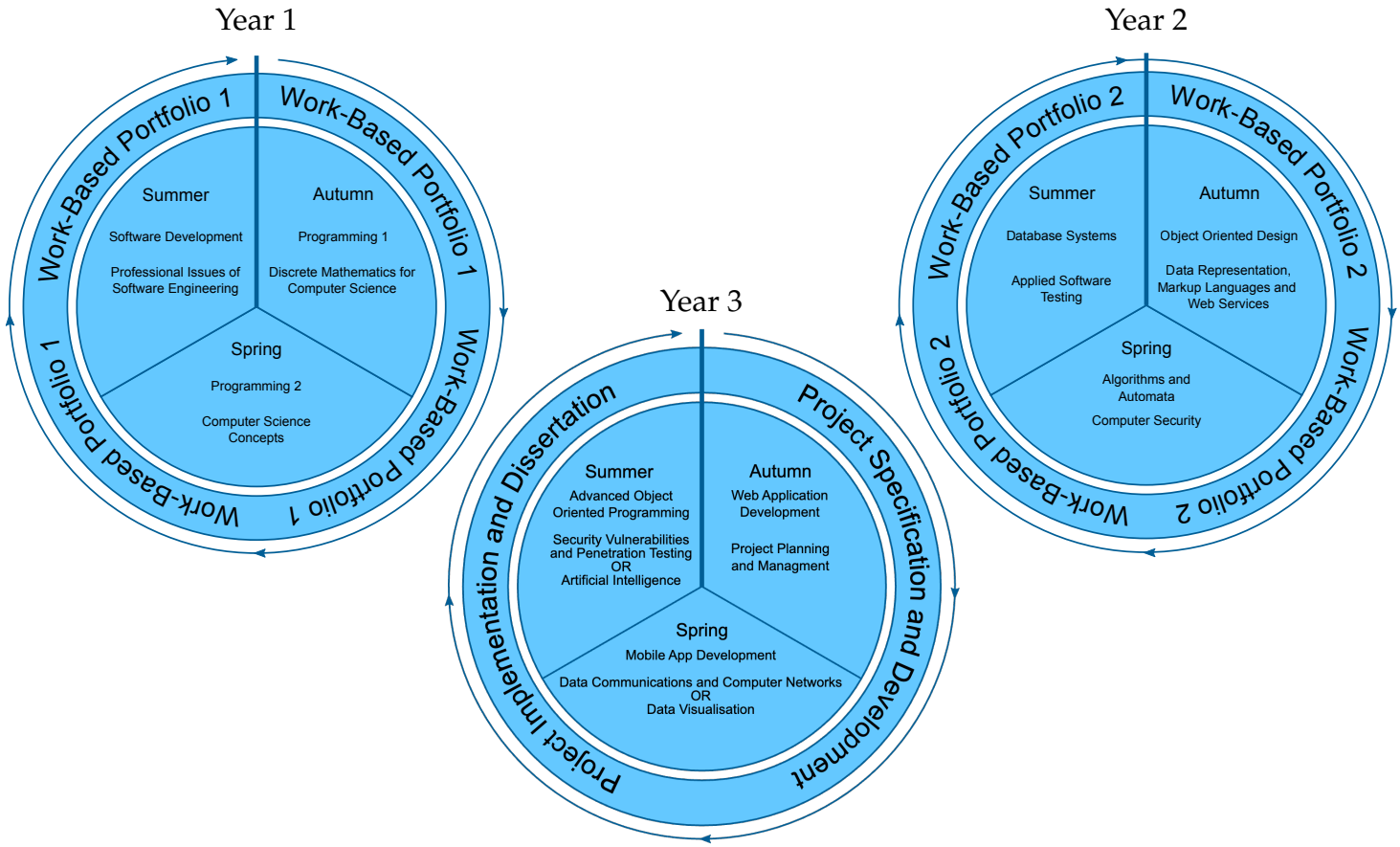
Taught Modules Each trimester consists of two taught modules each worth 15 credits. Each taught module is typically assessed with a combination of lab work, coursework and two class tests (a Mid-Term Test and End-Term Test). The pass mark for every module is 40%. At the end of each trimester, students will be offered the opportunity to redeem any failed modules (i.e., a module with a mark of less than 40%) from the current trimester; modules with a redeemed failure will have the mark capped at 40%.

At the end of each trimester, any student who has a non-tolerated failure in any module (i.e., less than 30% in any 1st- or 2nd-year module), or has accumulated more than 30 credits of failed modules during the academic year, will be required to withdraw from the programme.

Work-Based Modules In addition to the six taught modules, a 30-credit Work-Based Portfolio module runs throughout each of the first two academic years. These are core modules and must be passed with a minimum of 40%. Each Work-Based Portfolio module consists of a significant task from each of the taught modules where the knowledge acquired from that taught module is put into practice in the workplace. The work-based tasks are completed in the students own time.

In the final year, two 15-credit Work-Based Project modules run through the whole academic year. These are also core modules, and represent the specification, development and implementation of a substantial software project.

Summary A visual overview of the structure of the course is given below. Fine details of each of the modules can be found at the end of this Course Handbook.



CONTACT POINTS

Name	Telephone	Email
Programme Director		
Mrs Casey Hopkins	(29) 5152	C.L.Hopkins@swan.ac.uk
Head of Department		
Professor X Xie	(60)2370	X.Xie@swan.ac.uk
Departmental UG Programme Director		
Dr L O'Reilly	(60)4033	L.P.OReilly@swan.ac.uk
Business Liaison		
Dr M Moller	(60)6998	M.Moller@swan.ac.uk
Useful Numbers		
Operator	0	
Emergencies	333	
Estates	5240	
Health Centre	5321	
Nursery	3151	

Useful Websites

Swansea University student portal: http://myuni.swan.ac.uk
Here you can find a wealth of information and links to all university websites and services, including student email, Blackboard, etc.

Canvas: http://canvas.swan.ac.uk
Swansea University's virtual learning environment where you can find all teaching resources and submit coursework and assignments.

Intranet: http://intranet.swan.ac.uk
Swansea University's Intranet. Here you can manage your student record.

General Information

UNIVERSITY INFORMATION

Health and Safety Owing to Fire and Health & Safety Regulations (and from the general considerations of security), the hours of entry to certain buildings on the campus are restricted. Outside these hours, entry can only be gained on the written permission of the Head of School or other authority concerned.

Emergency Procedures In the event of a medical (or other) emergency call 333 immediately on a University landline or ask a member of staff to do so. If you are unsure if 333 has been called, call 333 again. For all emergency calls on University property dial 333 on a University phone. This helps the University to speed the arrival of the Emergency Services.

In the event of a fire:

1. Raise the alarm at once by breaking the glass of the nearest fire alarm call point.
2. Send the first available person to telephone 333 and give the location of the fire.
3. If appropriate, call for assistance and attack the fire with the correct extinguisher.
4. If the fire should get out of control, or your escape is threatened, leave the building at once, closing doors and windows as you go.

If you hear the fire alarm:

1. Leave the building immediately, closing all doors behind you: use the nearest available exit; do not stop to collect personal belongings; do not use the lifts; and do not re-enter the building.
2. When clear of the building proceed at once to the assembly area for that building (as indicated on the blue Fire Action signs around the building).

Data Protection The University's procedures comply with the principles of the Data Protection Act 2018. The responsibilities of students in relation to the provision of personal data can be found under the Publications tab on the Academic Registry Website. Students as data subjects have a right to request from the University a copy of their own personal data. A standard form must be completed. Forms and further details can be found at `http://www.swansea.ac.uk/the-university/world-class/vicechancellorsoffice/compliance/dataprotection/dataprotectionatswanseauniversity`.

The University's registration number with the Information Commissioner is Z6102454.

Assessment and Feedback

The university operates a strict policy with regards to submission deadlines. Late submissions will not be accept and students will receive 0% for any assignments which are not submitted before the deadline.

The university adheres to a policy of providing feedback on all submissions within three weeks of the submission deadline.

Transcripts and Diploma Supplements You can expect to receive an academic transcript at the end of your studies at Swansea which details the modules you pursued and the marks obtained.

LECTURING STAFF OF THE DEPARTMENT

Degree Apprenticeship Staff

C L Hopkins	Software Testing, Mobile/Web Application Development, Software Engineering
R Gaya	Machine Learning, Augmented/Virtual Reality, Software Testing
L P O'Reilly	Formal Methods, Algebraic Specification, Modelling and Verification, Software Engineering
F Pantekis	Modelling and Verification, Programming
O Petrovska	Logic, Functional Programming
S Shanmugasundaram	Security, Databases
J Roberts	History of Computing
L Clift	Robotics, AI
B Lloyd-Roberts	Security
C Clarkson	Machine Learning, AI

For further staff details of those in the Department visit `https://www.swansea.ac.uk/staff/science/compsci/`

General Reading List

Computer Science is at the heart of revolutionary changes in science and engineering, medicine, design and manufacture, commerce and financial services, government and public services, defence, transportation, media and communication, and at home. Students immersed in the technical world of computer science do not usually find it easy to develop an overview of the subject, its applications and influence on society.

The following is an annotated list of references which are accessible to all students of Computer Science with as yet little or no training, and which contain material which is helpful to the understanding of the subject and its role in the world. The exploration of the subject is a lifetime's work.

General

1. J Bentley. Programming Pearls. 2nd Ed, Addison-Wesley, 2000.

   A compendium of 13 essays, written for programmers, which develop insightful solutions to various programming tasks.

2. J Daintith and E Wright (eds). A Dictionary of Computing. 6th Ed, Oxford University Press, 2008.

   An established reference on computing, containing entries explaining terms from across the subject.

3. N Dale and J Lewis. Computer Science Illuminated. 6th Ed, Jones and Bartlett, 2015.

   This text provides a very broad overview of the subject, touching on the subject matter of much of the computer science curriculum.

4. A K Dewdney. The New Turing Omnibus: 66 Excursions in Computer Science. Palgrave MacMillan, 2003.

   66 easy-to-read "excursions" into 66 of the main points of interest in the subject.

Software Engineering

1. S Baase. A Gift of Fire. 4nd Ed, Prentice-Hall, 2012.

   Social, legal and ethical issues for computers and the Internet. It covers the challenges and implications of computer technology-and the responsibilities of professionals who design and use computer systems.

2. F Brookes. The Mythical Man Month. Anniversary Edition, Addison Wesley, 1995.

   Every software developer should read this, if only to understand something of just how hard it is to plan stuff. Of course modern technologies (e.g., Agile and Minimum Viable Product) have changed the game somewhat, but the core lessons remain.

3. N Leveson. Safeware: System Safety and Computers. Addison-Wesley, 1995.

   A study of the causes, and methods for prevention, of system failures, through the analysis of various famous major accidents caused by system failure.

4. P G Neumann. Computer Related Risks. Addison-Wesley, 1995.

   A compendium of computer mishaps, a study of their causes, and discussion of possible technologies for preventing similar mishaps.

Algorithms for Problem Solving

1. D Harel and Y Feldman. ALGORITHMICS: THE SPIRIT OF COMPUTING. 3rd Ed, Addison-Wesley, 2004.

   A readable presentation of the design of algorithms for problem solving, including their efficiency and inherent limitations.

2. D Harel. COMPUTERS LTD: WHAT THEY REALLY CAN'T DO. Oxford University Press, 2003.

   A small book describing as non-technically as possible the limitations of computers, the sorts of problems that you might think they can solve but cannot, and why they can't.

3. D Knuth. THE ART OF COMPUTER PROGRAMMING. Addison Wesley.

   Not a single book, buth rather a series of volumes by a pioneer in algorithms and programming which is commonly considered to be the bible of fundamental algorithms.

Logic and Discrete Mathematics

1. D Barker-Plummer, J Barwise and J Etchemendy. LANGUAGE, PROOF AND LOGIC. 2nd Edition, University of Chicago Press, 2011.

   An elementary introduction to syntax (formal language) and semantics (meaning) in the framework of mathematical logic; comes with software that allows the reader to explore the material and test their intuition and understanding in an interactive mode for self-study.

2. R Fagin, J Y Halpern, Y Moses and M Y Vardi. REASONING ABOUT KNOWLEDGE. MIT Press, 2004.

   A highly readable introduction to one of the most exciting themes in (applied) logic and informatics: how formal logic can be applied to express and reason about statements not only about "facts" but also about one's knowledge of facts and others' knowledge, ...

3. D R Hofstadter. GÖDEL, ESCHER, BACH: AN ETERNAL GOLDEN BRAID. 20th-Anniversary Edition, Penguin, 2000.

   An entertaining presentation of the logical underpinnings of computers and intelligent behaviour.

4. F Moller and G Struth. MODELLING COMPUTING SYSTEMS. Springer, 2013.

   An introduction to the basic mathematics and logic needed in Computer Science as presented in this programme of study.

5. R Smullyan. FOREVER UNDECIDED: A PUZZLE GUIDE TO GÖDEL. Oxford University Press, 1987.

   A playful presentation of some of the most important, and complex, 20th-century discoveries in logic which have profound impact on Computer Science and Artificial Intelligence.

6. D J Velleman. HOW TO PROVE IT: A STRUCTURAL APPROACH. 2nd Edition, Cambridge University Press, 2006.

   An introduction to the basic mathematics and logic needed in Computer Science, presented in the style and rigour of "structured" programming.

7. COMAP (The Consortium for Mathematics and Its Applications). FOR ALL PRACTICAL PURPOSES: MATHEMATICAL LITERACY IN TODAY'S WORLD. 10th Ed, W H Freeman and Company, 2015.

   A hefty text covering a large number of basic mathematical topics, most of direct relevance to computer science, presented at an accessible level, and backed up by numerous exercises and simple programming projects, as well as an extensive professionally-designed web site.

Computer Industry and Applications

1. R X Cringel. ACCIDENTAL EMPIRES. Penguin, 1996.

   Informative and amusing account of the American PC industry.

2. G W Flake. THE COMPUTATIONAL BEAUTY OF NATURE: COMPUTER EXPLORATIONS OF FRACTALS, CHAOS, COMPLEX SYSTEMS, AND ADAPTATION. MIT Press, 2000.

   An eye-opening account of instructive and exciting applications of computer science.

3. K Hafner and M Lyon. WHERE WIZARDS STAY UP LATE: THE ORIGINS OF THE INTERNET. Simon and Schuster, 1998.

   The remarkable 30-year story of the Internet.

4. W J Kaufmann III and L L Smarr. SUPERCOMPUTING AND THE TRANSFORMATION OF SCIENCE. Scientific American, 1993.

   A presentation in Scientific American's illustrative style of the capabilities of the world's most powerful computers.

History

1. M Davis. THE UNIVERSAL COMPUTER: THE ROAD FROM LEIBNIZ TO TURING. Norton, 2000.

   A history of the concepts underlying computers, and the scientists who developed them, written in an engaging style.

2. C and R Eames. A COMPUTER PERSPECTIVE. Harvard University Press, 1990.

   Superb picture-book on the origins and development of information processing, compilers and computing.

3. A Hodges. ALAN TURING: THE ENIGMA. Vintage, 1992.

   Fascinating biography of an original thinker on computing and computers.

4. S Singh. THE CODE BOOK: THE SECRET HISTORY OF CODES AND CODE-BREAKING. Fourth Estate, 1999.

   Entertaining history of the important problem of the encryption of information.

5. M Campbell-Kelly. FROM AIRLINE RESERVATIONS TO SONIC THE HEDGEHOG: A HISTORY OF THE SOFTWARE INDUSTRY. MIT Press, 2003.

   Excellent account of the development of the software industry.

First-Year Modules

CSF100
WORK-BASED PORTFOLIO 1

30 Credits YR	Coursework 1: 1666% Coursework 2: 1667% Coursework 3: 1667% Coursework 4: 1667% Coursework 5: 1666% Coursework 6: 1667%

Description

In this module, students will develop a portfolio of documents and projects which will demonstrate the application of knowledge gained from the co-requisite modules to their work experience.

Module Aims

This module will enable students to apply the knowledge gained in their academic studies to their work environment.

Syllabus

In each of the co-requisite academic modules, relevant tasks will be set which students will be able to apply to their workplace experience. In conjunction with their academic tutor and their mentor, they will prepare a portfolio of documents and projects to demonstrate their understanding of the relationship between their academic study and their work experience.

Learning Outcomes

Students will be able to demonstrate the relevance of their academic study to their working environment.

More specifically, they will develop the relevant skills presented in the learning outcomes of the co-requisite modules within the context of their industrial environment:
- they will understand and appreciate the legal, social, ethical, professional and security concerns surrounding modern computing systems.
- they will be able to choose appropriate technical solutions in specific problem instances.
- they will be able to choose appropriate software engineering project management methodologies based on specific project circumstances.
- they will understand the impact of theoretical concepts, particularly where they impose limitations, as they apply to particular software projects.
- they will be able to program in a range of languages including those in widespread industrial use.
- they will be able to plan projects with reference to risks, deadlines, dependencies and critical paths.
- they will be able to systematically test software during detailed development (unit and regression testing), project development (integration testing) and for quality assurance (acceptance testing).
- they will be able to use development tools and environments to create, maintain and manage software projects.
- they will be able to apply specialised computing concepts, for example, databases, markup languages, and network technologies, to the development of software systems.

Transferable Skills

Students will be able to prepare concise and clear documentation to demonstrate their understanding of how academic knowledge can be applied in the workplace. In particular, they will develop skills, e.g., in project design and time management, reflection, report writing, presentation and team-working skills.

CSF101
PROGRAMMING 1

15 Credits	Laboratory work:	10%
TB1	Coursework 1:	15%
	Coursework 2:	15%
	In class test (Invigilated on campus):	30%
	Class Test 2:	30%

Description

This module will teach students to solve computational problems by writing simple programs in a high-level language, specifically Java. Students will understand the fundamental principles underlying imperative programming languages, and have the ability and confidence to write programs in Java to solve a variety of simple problems.

Module Aims

The aim of this module is to introduce students to programming and allow them to solve computational problems by writing simple programs.

Syllabus

Introduction to programming in Java.
Program design techniques and best practice.
Simple data types, variables and assignments, control structures, functions, lists.
Compile-time and run-time errors.
Applications.

Learning Outcomes

Students will be able to develop straightforward programs to solve basic problems.
Students will be able to read and debug straightforward programs written by others.

Transferable Skills

Programming.
Computational thinking and problem solving.

CSF102
PROGRAMMING 2

15 Credits	Laboratory work:	10%
TB2	Coursework 1:	15%
	Coursework 2:	15%
	In class test (Invigilated on campus):	30%
	Class Test 2:	30%

Description

This module is a continuation of CSF101: Programming 1. In it, students will enhance their programming skills and will be introduced to object oriented programming.

Module Aims

The aim of this module is to enhance students knowledge of programming and to give them experience of programming with objects, algorithms and data structures.

Syllabus

Objects and classes.
Instance methods and variables.
Static methods and variables.
Object oriented programming and design techniques.
Encapsulation: public and private methods and fields.
Inheritance: sub-classes and overriding.
Polymorphism.
Graphical user interfaces.

Learning Outcomes

Students will be able to develop substantial programs to solve specific problems based on algorithms using standard data structures.
Students will have an awareness of efficiency considerations for different algorithms.
Students will be able to read and debug substantial programs written by others.

Transferable Skills

Programming.
Computational thinking and problem solving.

CSF104
SOFTWARE DEVELOPMENT

15 Credits	Laboratory work:	10%
TB3	Coursework 1:	15%
	Coursework 2:	15%
	In class test (Invigilated on campus):	30%
	Class Test 2:	30%

Description

This course will give students an understanding of the fundamental software tools, testing and design methods that are used to create reliable software. A state-of-the-art development environment will be shown, with hands-on experimentation and use of test systems. Students will also be given a sound grasp of the use of these systems in the different professional software development processes used in the software industry. Innovative software development methods such as Extreme Programming will be introduced and learnt in hands-on laboratory work.

Module Aims

This aim of this module is to expose students to development tools and techniques for developing reliable software.

Syllabus

Introduction to Integrated Development Environments (IDEs).
The Software Development Process.
Software Development Strategies.
Coding conventions.
Agile Programming/Extreme Programming.
Boundary value analysis, robustness testing and equivalence class testing.
Program debugging tools and debugging strategies.
Unit testing and tools for unit testing.
Version control systems.
Continuous integration tools.

Learning Outcomes

An understanding of the methods for developing reliable software.
A sound knowledge of current tools and methods for developing and testing software to ensure its reliability and to pinpoint known errors.
Students will be able to explain the operation and testing of computer programs.

Transferable Skills

Problem solving and logical thinking.
Demonstration skills.
Competency with software development tools.

CSF105
COMPUTER SCIENCE CONCEPTS

15 Credits	Coursework 1:	15%
TB2	Coursework 2:	15%
	In class test (Invigilated on campus):	30%
	Class Test 2:	30%
	Laboratory work:	10%

Description

This module gives an overview of some of the main principles underlying computers and computing from both a theoretical and an applied point of view.

Module Aims

This module aims to introduce students to the theoretical foundation of computing and how these foundations are applied in practice.

Syllabus

Binary values and number systems.
Data representation.
Abstract data types.
Gates and circuits.
Computing components.
Data compression techniques.
Software categorisation (application software vs system software).
Low-level programming languages and pseudocode.
Problem solving and algorithms (e.g., sorting, searching, and recursion).

Learning Outcomes

Students will gain an appreciation of the scope and limitations of computer science and its applications. They will be familiar with the principles involved in a number of areas of modern computing.

Transferable Skills

Logical thinking.
Computational problem solving.

CSF106

DISCRETE MATHEMATICS FOR COMPUTER SCIENCE

15 Credits	Coursework 1:	15%
TB1	Coursework 2:	15%
	In class test (Invigilated on campus):	30%
	Class Test 2:	30%
	Laboratory work:	10%

Description

This module introduces students to mathematical tools and techniques for modelling computing systems.

Module Aims

This module aims to provide the mathematical foundations required for modelling computing systems.

Syllabus

Motivating Case Studies.
Propositional logic.
Logical reasoning.
Requirements capture and analysis.
Sets.
Predicate logic.
Functions.
Relations.

Learning Outcomes

Students will become familiar with the fundamental mathematical techniques for modelling hardware and software systems and will develop skills in scientific modelling such as abstraction, the precise formulation of informal notions, rigorous reasoning and analysis.

Transferable Skills

General mathematical discipline.
Abstract modelling.
Formal reasoning.
Computational thinking.

CSF107

PROFESSIONAL ISSUES OF SOFTWARE ENGINEERING

15 Credits	In class test (Invigilated on campus):	30%
TB3	Class Test 2:	30%
	Coursework 1:	15%
	Coursework 2:	15%
	Laboratory work:	10%

Description

This module explores the main professional, societal and ethical issues associated with software development and computer science.

Module Aims

Students will be introduced to the main professional issues associated with software development and computer science. Students will also examine the main impacts of computing on society and social issues on the practice of computing, including legal and ethical concerns such as copyright and the data protection act.

Syllabus

Impact and reach of Computer Science and Software Engineering in Society, covering topics such as: domains of use and influence, ethical frameworks, codes of conduct, legal constraints, freedom of speech and censorship, privacy and surveillance and sustainability.
The impact of these issues on the development and testing of software will be examined.

Learning Outcomes

Students will be aware of major societal, ethical and profession-level issues associated with Computer Science.
They will also have experience of writing a report, producing a video, presenting material and working as a team.

Transferable Skills

Independent discovery of literature, reflecting on and critiquing different perspectives via report writing.
Development of video production skills as well as presentation skills.
Experience of team working.

Second-Year Modules

CSF200
WORK-BASED PORTFOLIO 2

30 Credits YR	Coursework 1: 1666% Coursework 2: 1667% Coursework 3: 1667% Coursework 4: 1666% Coursework 5: 1667% Coursework 6: 1667%

Description

In this module, students will develop a portfolio of documents and projects which will demonstrate the application of knowledge gained from the co-requisite modules to their work experience.

Module Aims

This module will enable students to apply the knowledge gained in their academic studies to their work environment.

Syllabus

In each of the co-requisite academic modules, relevant tasks will be set which students will be able to apply to their workplace experience. In conjunction with their academic tutor and their mentor, they will prepare a portfolio of documents and projects to demonstrate their understanding of the relationship between their academic study and their work experience.

Learning Outcomes

Students will be able to demonstrate the relevance of their academic study to their working environment.

More specifically, they will develop the relevant skills presented in the learning outcomes of the co-requisite modules within the context of their industrial environment:
- they will understand and appreciate the legal, social, ethical, professional and security concerns surrounding modern computing systems.
- they will be able to choose appropriate technical solutions in specific problem instances.
- they will be able to choose appropriate software engineering project management methodologies based on specific project circumstances.
- they will understand the impact of theoretical concepts, particularly where they impose limitations, as they apply to particular software projects.
- they will be able to program in a range of languages including those in widespread industrial use.
- they will be able to plan projects with reference to risks, deadlines, dependencies and critical paths.
- they will be able to systematically test software during detailed development (unit and regression testing), project development (integration testing) and for quality assurance (acceptance testing).
- they will be able to use development tools and environments to create, maintain and manage software projects.
- they will be able to apply specialised computing concepts, for example, databases, markup languages, and network technologies, to the development of software systems.

Transferable Skills

Students will be able to prepare concise and clear documentation to demonstrate their understanding of how academic knowledge can be applied in the workplace. In particular, they will develop skills, e.g., in project design and time management, reflection, report writing, presentation and team-working skills.

CSF202
OBJECT ORIENTED DESIGN

15 Credits	Coursework 1:	15%
TB1	Coursework 2:	15%
	Laboratory work:	10%
	In class test (Invigilated on campus):	30%
	Class Test 2:	30%

Description
This module exposes the students to the major principles of object-oriented software development.

Module Aims
This module aims to extend students knowledge of programming by considering aspects and principles of object-oriented software development.

Syllabus
UML (Universal Modelling Language).
Requirement analysis and specification in an object-oriented framework.
Object oriented software architecture design.
Object orientation: inheritance, abstraction, encapsulation and polymorphism.
Using design patterns.
Object-oriented software development and reusability in software engineering.

Learning Outcomes
Students will gain an understanding of the principles of object-oriented programming concepts, and knowledge of their applications in software design and engineering processes.

Transferable Skills
Problem solving through analysis and abstract reasoning.

CSF203
DATABASE SYSTEMS

15 Credits	Coursework 1:	15%
TB3	Coursework 2:	15%
	Laboratory work:	10%
	In class test (Invigilated on campus):	30%
	Class Test 2:	30%

Description

This module will discuss the theory, design and implementation of databases.

Module Aims

This module aims to provide students with the skills needed to design and implement databases.

Syllabus

Introduction to databases and data.
Database software and benefits.
ANSI/SPARC model, database structure.
Relational databases: properties, designing, problems.
Normalisation: normal forms, functional dependence, primary keys, integrity constraints and rules, validation.
Real world examples: SQL and practical sessions using relational databases.
Client/server technology and web servers.
ER Models: entities, relationships, modelling, attributes, converting to relational model.
Relational calculus and its application to databases, relational algebra: select, project, join, union, intersection, difference, cartesian product, query optimisation.
Recovery: transaction processing, locking, detecting deadlocks.
Concurrency: Multi-user databases - client/server, distributed, commit protocols.
Security: managing users and passwords, encryption, securing MySQL.
Introduction to NoSQL databases.

Learning Outcomes

Students will be aware of relational databases and the need for the normalisation of data. Students will have been exposed to transaction processing and how to detect and avoid problems that can arise in a multi-user and/or distributed environment. Students will have designed a database using the ER model, and have practical experience of a relational database.

Transferable Skills

Problem identification, problem analysis and abstract modelling. Abilities to learn and use computer systems and software packages effectively, and to evaluate and deploy new technologies.

CSF205

DATA REPRESENTATION, MARKUP LANGUAGES AND WEB SERVICES

15 Credits	Coursework 1:	15%
TB1	Coursework 2:	15%
	Laboratory work:	10%
	In class test (Invigilated on campus):	30%
	Class Test 2:	30%

Description

Web services is a software system designed to support interoperable interaction between systems over the network. The two key distributed web service architectures that are popular and have been in use for the past two decades are the SOAP and the REST. This module will cover the background and architecture of a web service, the design and implementation of web services using SOAP and REST along with the various data formats (XML, JSON) they support.

Module Aims

The aim of this module is to enhance students' knowledge of the background and architecture of a web service, the design and implementation of web services using SOAP and REST along with the various data formats (XML, JSON) they support. It will allow the students to gain hands-on experience in creating their own web services and also learn how to write testable code to test their web services as used in the industry. They will also learn the evolution of the web services to the new cloud-based service architecture.

Syllabus

Web services and Networks (TCP/IP, Sockets)
Web Services using SOAP:
- SOAP Concepts, Structure and WSDL (Web Service Description Language)
- Service Endpoints
- Messages
- Types and their representation
- RPC vs Document
- Literal vs Wrapped (inc. Document Literal Wrapped)

Web Services using REST:
- REST Concepts and HTML/CSS
- REST and CRUD Operations
- HOTEAS and REST

Data Formats for Web Services:
- XML, XML Namespaces, XSD (Schema Definition)
- JSON

SOAP and REST Implementations in practice:
- Using annotations to generate proxy code in Java
- Loosely coupled web applications
- Build Tools and Libraries

Cloud Service Models (IaaS, PaaS, SaaS) concepts

Learning Outcomes

Students will be able to:
- Understand the underlying data technology (in particular sockets and HTTP) of web services;
- Describe and explain the format and operation of SOAP Web Services;
- Describe and explain the format and operation of REST Web Services;
- Be able to build Web Services in Java using both SOAP and REST;

- Apply markup/data representation languages to model and represent data for the web services;
- Critically evaluate and select a web service architecture for user requirements;
- Analyse the cloud based service architecture;

Transferable Skills

Programming skills.
Problem definition.
Ability to evaluate and deploy new technologies.

CSF206
ALGORITHMS AND AUTOMATA

15 Credits	Coursework 1:	15%
TB2	Coursework 2:	15%
	In class test (Invigilated on campus):	30%
	Class Test 2:	30%
	Laboratory work:	10%

Description

This module introduces students to theoretical tools and techniques important in software development. The first half of the module is concerned with algorithms and data structures and will enable the student to understand how the selection of different algorithms and data structures affects the performance and efficiency of a program. The second half of the module will concerned with automata in the form of labelled transition systems and will enable the student to understand their role in modelling computing systems.

Module Aims

This module introduces students to theoretical tools and techniques that are important for software development.

Syllabus

The formal concept of algorithm, data structure, and program efficiency.
The definition and analysis of basic sorting and searching algorithms.
Algorithmic design concepts including divide and conquer, greedy algorithms and dynamic programming.
The study of labelled transition systems and their role in modelling computing systems.
Games and strategies and their use in verifying the correctness of computing systems.

Learning Outcomes

Students will appreciate the idea of analysing an algorithm to determine its efficiency.
Students will be familiar with, and be able to use, basic data structures.
Students will know and understand standard sorting and searching algorithms and be able to comment on their relative performance.
Students will be familiar with the use of automata for modelling computing systems.

Transferable Skills

Abstract modelling.
Formal reasoning.
Computational thinking.
Problem solving.

CSF207
COMPUTER SECURITY

15 Credits	In class test (Invigilated on campus):	30%
TB2	Class Test 2:	30%
	Coursework 1:	15%
	Coursework 2:	15%
	Laboratory work:	10%

Description

The course examines the practical and theoretical aspects of computer and network security.

Module Aims

The aim of this course is to examine the theoretical and practical aspects of computer and network security. The module also aims to enhance students' knowledge and skills in Cryptography.

Syllabus

Security threats and their causes.
Security criteria and models.
Cryptography: including basic encryption, DES, AES, hash functions.
Access Control.
Security tools and frameworks: including IPSec, TLS, SSL, SSH and related tools.
Vulnerabilities and attacks: including port scanning, packet sniffing, and SQL injection.
Security issues in wireless networks.
Security on the cloud.
Blockchain Technology and Bitcoin.
Penetration Testing.
Tor Network.

Learning Outcomes

Students will be able to:
- Identify security threats and their causes in today's computing infrastructures.
- Apply techniques from Cryptography and Cryptanalysis.
- Build secure systems.
- Apply techniques to enhance the security of existing systems, and critically evaluate the limits of these techniques.

Transferable Skills

Problem solving.
Analysis.
Ability to evaluate and deploy new technologies.

CSF209
APPLIED SOFTWARE TESTING

15 Credits	In class test (Invigilated on campus):	30%
TB3	Class Test 2:	30%
	Coursework 1:	15%
	Coursework 2:	15%
	Laboratory work:	10%

Description

Testing is the process of systematically experimenting with an object, i.e., the System Under Test, in order to establish its quality, where quality means the degree of accordance with the intention or specification. This module will cover various test scenarios; practical exercises will allow the students to gain hands-on experience. As well as studying the theory behind testing, students will also learn how to write testable code as well as work with a variety of testing tools used in the industry.

Module Aims

The aim of this module is to introduce students to various practical applicable testing techniques and to make them aware of the different levels of testing.

Syllabus

The module provides a profound overview of industrially relevant methods in software testing and points out current research directions.
- Functional Testing: Boundary Value Testing, Equivalence Class Testing, Decision Table- Based Testing.
- Structural Testing: Path Testing, Data Flow Testing.
- Integration and System Testing: Levels of Testing, Approaches to Integration Testing.
- Object-Oriented Testing: Issues, Class Testing, Object-Oriented Integration Testing.
- Practical application within the industry.

Learning Outcomes

Students will be able to:
- Explain testing as a method to validate software systems
- Analyse software testing problems
- Critically evaluate and select software test scenarios
- Develop code suitable for testing
- Understand and use basic tools for testing

Transferable Skills

Analysis of systems; systematic design of experiments.

Third-Year Modules

CSF300

PROJECT IMPLEMENTATION AND DISSERTATION

15 Credits YR	Presentation: 20% Report: 80%

Description

This module forms the second part of the capstone project for the BSc in Applied Software Engineering project (together with CSF301). It consists of the implementation of a software system; a substantial written dissertation; and a video demonstration of the system.

Module Aims

This module, in conjunction with CSF301, aims to enable students to demonstrate their ability in delivering a major software application using established industry-standard software engineering principles.

Syllabus

Students must document and present a substantial software (or software/hardware) system. The deliverables consist of: the software system itself; the dissertation completely documenting the system; and a video demonstration of the system.

Learning Outcomes

By the end of this module, students will
(1) have defined the requirements for, specify, designed and implemented a complete software system,
(2) have applied the major phases of the life-cycle of a software engineering project,
(3) have managed a substantial project and have carried out a risk analysis,
(4) have evaluated and reflected on the management of their project,
(5) have researched and presented the background material, and documented their project.

Transferable Skills

Written communication and documentation, verbal presentation and interactive discussion. Information retrieval, ability to read critically, and judge information.

CSF301
PROJECT SPECIFICATION AND DEVELOPMENT

15 Credits	Report: 40%
YR	Report: 30%
	Presentation: 30%

Description

This module forms the first part of the capstone project for the BSc in Applied Software Engineering (together with CSF300). It consists of the project proposal, presentation, progress report and development of a software system.

Module Aims

The aims of this module are:

• to provide students the opportunity of specifying, designing and implementing a complete system and experiencing the major phases of the life-cycle of a computing project;

• to enhance students' competence in system design, risk analysis and management, and their fluency in using programming languages and tools;

• to give students an intellectual challenge to their abilities to learn new subjects without instruction, and to further develop their abilities in literature searching, report writing, verbal presentation, project planning and time management.

Syllabus

Students must develop a substantial software (or software/hardware) system using a well-defined software life-cycle model. The deliverables consist of: a project proposal document describing the system to be built; a presentation of the proposed project; and a progress report.

Learning Outcomes

By the end of this module, students will

(1) have defined the requirements for, specify, designed and implemented a complete software system,

(2) have applied the major phases of the life-cycle of a software engineering project,

(3) have managed a substantial project and have carried out a risk analysis,

(4) have evaluated and reflected on the management of their project,

(5) have researched and presented the background material, and documented their project.

Transferable Skills

Written communication and documentation, oral presentation, and interactive discussions. Time management, risk analysis and management, and project management. Problem solving. Information retrieval, ability to read critically, to précis and judge information, and ability to manage learning processes. Software development.

CSF302
PROJECT PLANNING AND MANAGEMENT

15 Credits	In class test (Invigilated on campus):	45%
TB1	Laboratory work:	10%
	Coursework 1:	15%
	Coursework 2:	15%
	Coursework 3:	15%

Description

Software projects have long had a reputation for cost and time overruns - but they need not, and there are well-established, and emerging, techniques and processes to manage them well and effectively: for example, agile methodologies like Scrum which are becoming a de-facto standard in the industry. Also, many projects have significant legal, social, ethical and professional consequences that a practitioner needs to be aware of and sensitive to.

This module develops the fundamental skills of successfully building complex software systems, and the implications, including on wider society, of doing so. It will also prepare students for work on any project by equipping them with the skills to successfully plan them, and to commence that planning process.

Module Aims

This module aims to introduce students to the process of developing software using modern life-cycle methodologies, and in understanding the legal, social, professional and ethical ramifications of software.

Syllabus

Project planning and management principles:
- Timescales and dependencies
- Scoping and resources
- Risks: identifying, quantifying, managing, monitoring, and mitigating
- Team management

Requirements and Specification
- Heavyweight and lightweight models

Methodologies for developing software:
- Traditional - waterfall, prototyping, spiral
- Rapid Application Development (RAD)
- Iterative and incremental
- Agile development
- Hybrid models - Scrum (agile/incremental)

Legal, Social, Ethical, and Professional Issues

Learning Outcomes

Students will be able to:
- Explain the legal, social, ethical and professional framework, particularly with reference to software engineering.
- Apply a range of software development models, including agile as well as traditional, based on an understanding of their specific properties, advantages, and disadvantages.
- Make choices between a range of software development models.
- Plan and manage basic projects, including risk analysis and controls, time scale and resource planning, exception monitoring, and progress monitoring and control.
- Work in teams on software projects.

Transferable Skills

Project planning; time management; risk analysis and management; legal, social, ethical and professional issues.

CSF304
WEB APPLICATION DEVELOPMENT

15 Credits	Coursework 1:	15%
TB1	Coursework 2:	15%
	Laboratory work:	10%
	In class test (Invigilated on campus):	30%
	Class Test 2:	30%

Description

The module will develop the principles and technologies used for building web-based systems. Practical experience of building web systems will be gained via laboratories and coursework.

Module Aims

The module will develop the principles and technologies for building modern web-based systems. Students will develop an appreciation for the security risks which web applications face. Students will gain experience developing a database driven web application. At the end of the course students will be equipped to apply this knowledge to the ever evolving range of web technologies.

Syllabus

The history of web application development.
HTML and CSS: Introduction and Good Practices.
Web Application Design.
Introduction to PHP.
MVC driven web applications.
Security and identity in web applications.
Web development using Javascript and AJAX.

Learning Outcomes

Students will be able to explain the key aspects of current web programming principles and technologies.
Students will be able to critically assess the advantages and disadvantages of competing web technologies.
Students will be able to create web applications following methodological good practice.
Students will be able to design secure web applications and evaluate their effectiveness.

Transferable Skills

Computational thinking and problem solving.
Using advanced software packages for programming.
Self-study.

CSF306
MOBILE APP DEVELOPMENT

15 Credits	Coursework 1:	10%
TB2	Coursework 2:	30%
	Laboratory work:	10%
	In class test (Invigilated on campus):	50%

Description

This module will introduce students to developing well-designed and functional apps for mobile devices. Special emphasis is placed on general design paradigms for mobile devices, taking into account limitations such as battery life, limited memory and low user attention compared with desktop computers.

Module Aims

The aim of this module is to give the student an understanding of the history, development and future of apps for mobile phones, tablets and other related devices. Students will gain experience in app development for Android/iOS devices. They will learn how to write apps and what kinds of apps work best with mobile devices.

Syllabus

Brief history of mobile apps
Apple & Android Design Principles
Mobile HCI & Usability
Building a Basic Application
User Interfaces
- Layouts
- Widgets/Components
- Event handling
- Forms & Inputs

Application Life Cycle
Data Persistence
- Using databases to store data

Native Device Features
- Using the Camera
- Maps and working with mapping services
- Accessing and using Location information

Notifications
The Web
- Sending HTTP requests

State Management

Learning Outcomes

Students will be able to:
- Apply the methods and techniques to design and implement Android and iOS apps.
- Explain the interface and communications paradigms for mobile applications on small-screen devices with non-traditional IO.
- Develop applications targeted on mobile systems by means of device simulators and deploy them to the actual hardware.
- Describe the history of mobile apps.

Transferable Skills

Computational thinking and problem solving.

CSF307

ADVANCED OBJECT-ORIENTED PROGRAMMING

15 Credits	Laboratory work:	10%
TB3	Coursework 1:	15%
	Coursework 2:	15%
	In class test (Invigilated on campus):	30%
	Class Test 2:	30%

Description

This module will give an overview of advanced techniques in object oriented programming. It will investigate some standard libraries in-depth and a range of good practices when using higher-level and lower-level languages. It will examine many elements of these languages and students will understand how to write more efficient code, and how to guard against potential security flaws.

The module introduces the programming language C and, including low-level aspects of programming that are usually abstracted in languages like Java and further expands the knowledge of students in Java by looking into advanced features of the language, and their respective impact on performance.

Module Aims

This module aims to develop skilled programmers that understand C and are proficient in Java. It also aims at increasing industrially relevant skills in some specific areas of Computer Science, such as system programming and High-Performance Computing.

This module aims to investigate standard libraries and techniques in-depth, giving the students experience in developing highly robust and efficient software, and understanding and applying their knowledge on good memory management.

Syllabus

Introduction to C:
- Basic principles
- Memory access, responsible management of memory and resources
- Pointers and pointer arithmetic, hazards in the use of pointers
- Pre-processing, compiling, linking
- Structures, type definitions

Advanced features of Java:
- Generics
- The functional API and Multithreading
- Efficient hashed structures
- Reflective access and modification
- Serialization
- The memory model and Garbage Collection
- Sensitive data handling and manipulation

Learning Outcomes

Students will be able to:
- Develop industry-standard solutions to a specific problem.
- Critically apply object orientated design to large/complex programming problems.
- Explain the ideas and techniques of generic programming, including how to use the ideas and techniques to write efficient code.
- Understand low-level principles and take them into account when using higher-level languages, to write robust and secure software.

Transferable Skills

Abstract modelling and implementation of software via genetic techniques. Problem solving. Ability to to evaluate and deploy new technologies.

CSF308
DATA COMMUNICATIONS AND COMPUTER NETWORKS

15 Credits	In class test (Invigilated on campus):	30%
TB2	Class Test 2:	30%
	Coursework 1:	15%
	Coursework 2:	15%
	Laboratory work:	10%

Description

This module will examine the use of computer networks and to identify the forces behind their development. The design and implementation of various network topologies, architectures, protocols and algorithms will be considered. The module will place network concepts in the context of the widely-used TCP/IP model.

Module Aims

This module aims to give students a broad understanding of computer networks and the concepts involved within the various levels of networking.

Syllabus

Introduction: Objectives and basic concepts of networking; topologies, switching, and circuits; multiplexing; connected and connectionless networks.
Protocols: Protocol concepts and examples (POP3); protocol hierarchies; OSI 7 layer model; TCP/IP model.
Application layer: HTTP; DNS; SMTP; IMAP vs POP; RFC 822; MIME.
Data representation; data compression; data security and cryptography.
Transport layer: Ports and sockets; UDP; concepts of reliable data transfer; TCP.
Network layer: Routing and forwarding; routing algorithms; internetworking; the IP protocol; IP addresses, networks and subnetmasks; congestion control.
Data link layer: Framing; byte stuffing/bit stuffing; data encoding; error correcting codes; Hamming distance; CRC algorithm; Hamming single bit code.
Data and signal; bandwidth and data rate; PSTN; MODEMs; data encoding; ADSL and SDSL.
MAC sublayer and wireless: MAC protocols; CSMA/CD; MAC addressing and ARP; wireless LANs; DCF vs PCF; 802.11.
Physical layer: modulation, analogue-digital conversion, line codes.

Learning Outcomes

Students will acquire an understanding of the fundamental concepts of data communication and computer networks, and will be familiar with the techniques used in designing and implementing data communication systems. Students will also gain an appreciation of the rapid development of this technology and be aware of its impact upon industry and society.

Transferable Skills

Ability to evaluate and deploy new technologies.

CSF325
ARTIFICIAL INTELLIGENCE

15 Credits TB3	Coursework 1: 15% Coursework 2: 15% In class test (Invigilated on campus): 30% Class Test 2: 30% Laboratory work: 10%

Description

An introduction to Artificial Intelligence, focusing primarily on reasoning and problem solving as a search for a solution rather than on statistical techniques for classification. The course may cover topics from amongst: search techniques; knowledge representation and expert systems; planning; scheduling; qualitative reasoning; language processing with grammar rules; and meta-programming, as well as agents, multi-agent systems, and agent collaboration.

Module Aims

The aims of the course are for students to:
- gain a firm foundation in Artificial-Intelligence techniques;
- understand how to analyse problems and data so as to design an AI implementation to solve these problems with the data.

Syllabus

- Fundamental Issues in AI
- Basic Search Strategies
- Advanced Search
- Reasoning Under Uncertainty
- Programming for AI
- Basic Knowledge Representation and Reasoning
- Advanced Representation and Reasoning
- Natural Language Processing
- Advanced: Application of NLP and Explainable AI
- Concept of rational agent
- Multi-Agent Systems
- Agent communication and collaboration

Learning Outcomes

On completion of this module, students will be able to:
1. Demonstrate a systematic knowledge of the fundamental concepts in AI.
2. Apply a wider range of AI techniques and to evaluate their advantages and disadvantages.
3. Identify problems that are amenable to solution by AI methods and methods which may be suited to solve a given problem.
4. Demonstrate competency in developing programs to address problems in AI automatically.

Transferable Skills

Ability to learn and use computer systems and software packages. Ability to evaluate new technologies. General mathematical analysis. Problem solving and especially abstract modelling and reasoning.

CSF328

SECURITY VULNERABILITIES AND PENETRATION TESTING

15 Credits	Coursework 1:	10%
TB3	Coursework 2:	10%
	In class test (Invigilated on campus):	25%
	Class Test 2:	25%
	Laboratory work:	30%

Description

The aim of this course is to examine methodological and practical aspects of cyber security threats and analysis techniques.

Module Aims

The aim of this course is to explore both classical and current threats in cyber security and in particular to explore how these threats are typically exploited. The module aims to provide a hands-on experience of industry standard security analysis techniques. The module also aims to enhance students' knowledge and skills in programming and system analysis with respect to security.

Syllabus

Security threats and their causes.
Vulnerabilities and attacks: including port scanning, packet sniffing, SQL injection.
Countermeasures for attacks.
Security analysis tools and frameworks: including Kali Linux and Metasploit.
Shell code.
Legal and ethical issues of ethical hacking.
Social Engineering.
Methodologies for penetration testing.

Learning Outcomes

Students will have the ability to identify security threats and their causes in today's computing infrastructures.
Students will be able to explain in detail a number of methodologies for security analysis of a system.
Students will have practical experience in recognising vulnerabilities and will be able to defend against them.
Students will be able to apply techniques of penetration testing to existing systems, and gain a critical awareness of the limits of these techniques.

Transferable Skills

Problem analysis and solving. Ability to manage learning process. General programming skills. Ability to learn and use computer systems effectively. Ability to evaluate and deploy new technologies.

CSF337
DATA VISUALISATION

0 Credits TB2	Coursework 1: 15% Coursework 2: 15% In class test (Invigilated on campus): 30% Class Test 2: 30% Laboratory work: 10%

Description

Data Visualisation is concerned with the automatic or semi-automatic generation of digital images that depict data in a meaningful way(s). It is a relatively new field of computer science that is rapidly evolving and expanding. It is also very application oriented, i.e., real tools are built in order to help scientists from other disciplines.

Module Aims

Data Visualization is concerned with the automatic or semi-automatic generation of digital images that depict data in a meaningful way(s). It is a relatively new field of computer science that is rapidly evolving and expanding. It is also very application oriented, i.e., real tools are built in order to help scientists from other disciplines.

Syllabus

We will start off by introducing the fundamentals of visualisation. Introductory topics include purposes and goals of visualisation, applications, challenges, the visualisation pipeline, sources of data, data dimensionality, data types, and grid types.
The next sub-topic examines information visualisation, that is, visual representations of abstract data. Information visualisation topics include hierarchical data, tree maps, cone trees, focus and context techniques, graphs and graph layouts, multi-dimensional data, scatter plots, scatter plot matrices, icons, parallel coordinates, interaction techniques, linking and brushing.
The second major sub-topic is the study of volumetric data. Volume visualisation topics include slicing, surface vs. volume rendering, transfer functions, interpolation schemes, direct volume visualisation, ray casting, image order vs. object order algorithms, gradients filtering, interpolation, and isosurfacing.
The third major sub-topic is vector field visualisation. Topics include simulation, measured, and analytical data, steady and time-dependent (unsteady) flow, direct and indirect flow visualisation, applications, hedge hog plots, vector glyphs, numerical integration schemes, streamlines, streamline placement, geometric flow visualisation techniques, texture-based techniques, and feature-based flow visualisation.

Learning Outcomes

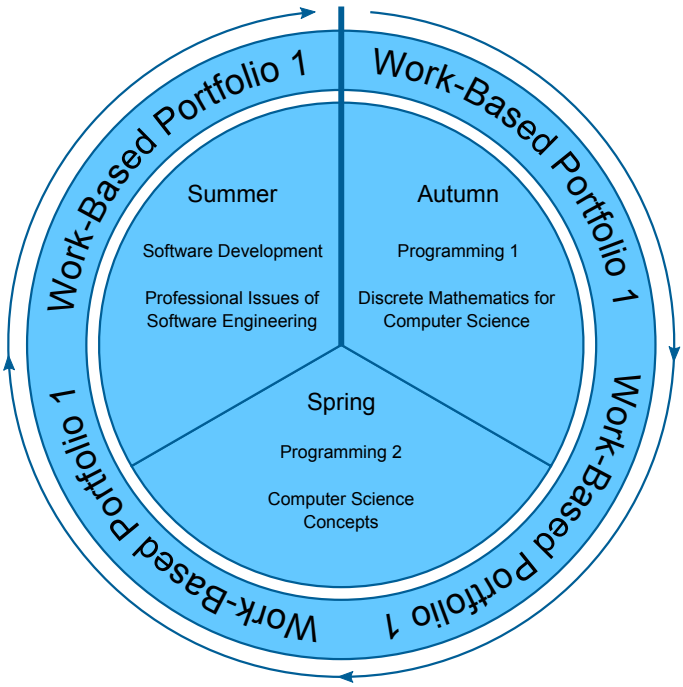
Students will gain competence in the field of data visualisation.
They will understand the basic methods available for the computer-aided depiction of data from several interdisciplinary and application oriented sources.
They will also gain and understanding of the visualisation problems that have been solved as well as the challenges that remain.

Transferable Skills

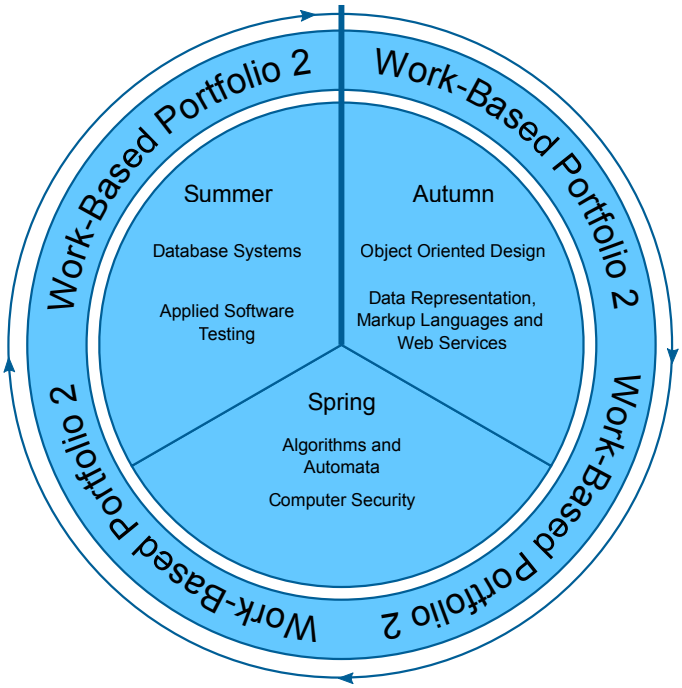
The ability to identify and generate advanced visualisations of data, comparative analysis, the ability to identify sources of data and the challenges when visualising data as well as the challenges that scientists and practitioners from other disciplines face.

Course Structure

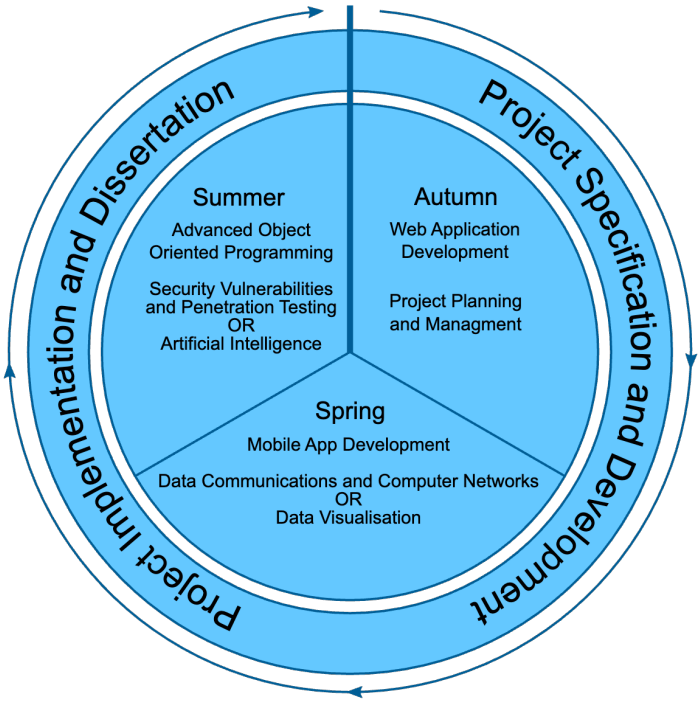
Year 1



Year 2



Year 3



Swansea University
Prifysgol Abertawe

www.swansea.ac.uk/undergraduate/courses/science/computer-science/bsc-applied-software-engineering/