CE/CZ 2001 – Algorithms

Course Code	CE/CZ 2001							
Course Title	Algorithms							
Pre-requisites	CE/CZ 1007: Data Structures							
No of AUs	3							
Contact Hours	Lectures	25	TEL	2	Tutorials	11	Example classes	8

Course Aims

This core engineering course aims to develop your knowledge, understanding and skills about algorithms, including: (1) methods and techniques to design and implement algorithms; (2) methods and techniques to analyse the correctness and resource requirements (mainly the time and space complexities) of algorithms. Because algorithms are essential to both Computer Engineering (CE) and Computer Science (CS), this course has vital importance for learning other courses in CE or CS, and shall prepare you for future careers in the science and technology of computing.

Intended Learning Outcomes (ILO)

By the end of this course, you should be able to:

1. Conduct complexity analysis of basic algorithms.
2. Design and analyse algorithms using the suitable strategies (e.g. incremental, divide and conquer, data-structure, and greedy approaches) to solve a problem.
3. Compare the efficiencies of different algorithms for the same problem (e.g. searching, sorting or graph traversal).
4. Describe various heuristic problem-solving methods.
5. Implement algorithms from pseudo code into real code.

Course Content

	Topics	Lectures (Hours)	Tutorials (Hours)	Example Classes (Hours)
1	**Introduction to algorithms** What is an algorithm? Algorithm design and analysis. Examples of different design: Power function, Sine Function, Maclaurin Series vs CODIC. Mathematical Foundation: Set, Functions, Floor and Ceiling, Power and Logarithm, Summations and Series, Limits, Proof Method.	2	1	0
2	**Analysis of Algorithms** Time and space complexities of algorithms. Analyzing basic program constructs. Best case, worst case and average case time complexity analysis. Deducing recurrence relations for time complexity of recursive algorithms. Solving elementary recurrence relations. Asymptotic time complexity analysis. Big-Oh, big-Omega, and big-Theta notations. Common Complexity Classes. Basic techniques for proving asymptotic bounds. Space Complexity. Faster computer or faster algorithms.	5	2	2
3	**Searching** Iterative and recursive sequential search algorithms. Binary search, its invariance, and complexity. Open and closed address hashing using linear probing and double hashing collision resolution techniques. All algorithms covered are accompanied by asymptotic complexity analysis.	2	1	2
4	**Sorting** Insertion-sort, Heap-sort, Quick-sort, Merge-sort, which contrast three approaches: Incremental, Data Structures and Divide-and-Conquer. All algorithms covered are accompanied by asymptotic complexity analysis.	6	3	2

5	**Graphs** Basic graph representation methods, adjacency lists, and adjacency matrix, Systematic graph traversals with breadth-first search (BFS) and depth-first search (DFS) algorithms. A generic backtracking algorithm and its complexity. Maze-Search. Introduction to Greedy algorithms; Dijkstra's Single-source Shortest Paths algorithm, Correctness proof and complexity of Dijkstra's algorithm. Minimum Spanning trees and their properties. Greedy algorithm for finding Minimum Spanning Trees (MSTs) using Prim's algorithm.	6	3	2
6	**Basic Computability and Complexity Theory** Tractable and intractable problems. Un-computable functions. The Halting Problem. Non-determinism as a computational model. The Classes NP and P. Brute-force and heuristic algorithms Nearest-link-first and shortest-link-first for Travelling Sales Person (TSP) problem.	2	1	0
7	Revision	2	0	0
	Check for Hours	=25	=11	=8

Assessment (includes both continuous and summative assessment)

a) Final Examination: 60%
b) Mid-Term Quiz: 20%
c) Example Classes: 20%

CE/CZ 2002 – Object Oriented Design & Programming

Course Code	CE/CZ2002							
Course Title	Object Oriented Design & Programming							
Pre-requisites	CE/CZ1007: Data Structure							
No of AUs	3							
Contact Hours	Lectures	24	TEL	0	Tutorials	10	Laboratories	10

Course Aims

The object-oriented paradigm to the design of software is one major successful approach to address complexity and maintainability issues in software systems. We want students to establish an object-oriented mindset and to gain valuable insights into how software can be developed using the object-oriented approach. This course should not be interpreted to be a pure programming language course. Rather, the programming language serves to illustrate, via practical examples, the concepts learnt in the course.

As a student of this course, you will learn essential object-oriented concepts such as encapsulation, the separation of design from implementation; the use of inheritance and polymorphism. You will discover how to describe these concepts using appropriate UML diagrams. Finally, you will also learn good design principles for reuse, and to realise these principles using object-oriented programming languages such as Java and/or C++.

Intended Learning Outcomes (ILO)

Upon completion of the course, you should be able to:

1. Explain the concepts of object-oriented methodology and demonstrate it in developing software programs.
2. Design simple software programmes using good design principles with consideration for reuse and maintainability, to solve problems.
3. Implement a given design in Java and/or C++.

Course Content

	Topics	Lectures (Hours)	Tutorials (Hours)	Presentations (Hours)
1	**Introduction to Object Orientated Programming** Procedural vs OO programming; Object and Class; Four basic elements and four features of OO programming.	2	0	
2	**Class and Object** Attributes; Class Definition; Message Sending; Copying Objects; The Keyword 'this'; Accessors and Mutators; The Keyword 'static'; Static vs. Instance methods; Object Composition.	4	2	
3	**Inheritance in Java** Generalization and Specialization; Method Overloading & Overriding; Liskov Substitution Principle; Visibility Modifiers; Final Classes and Methods; Abstract Classes and Methods; Multiple Inheritance and Interfaces.	3	2	
4	**Exception Handling & Persistent Objects (E-Learning)** Error Handling; Java's Exception Handling; Java's Exception Hierarchy; Exception Classes; Object serialization; Saving objects, e.g. to file, to RDBMS.	2	1	
5	**Polymorphism in Java** Polymorphism; Binding; Object Typecasting; Benefits of Polymorphism; Three ways of Method Overriding.	3	2	
6	**Object Relationships** Aggregation. Composition. Associations. Delegation. Cardinality.	2	1	
7	**Object Collaboration** Messages. Object Interactions. Sequence Flow. UML Sequence Diagram.	2	1	
8	**Modelling OO Application** Identifying Class and Objects, Defining Classing, Use of Class and Sequence	2	1	
9	**Designing for Reuse** Good design principles e.g. Single Responsibility Principle (SRP). Don't Repeat Yourself (DRY) Principle, Open-Closed Principle (OCP). Interfaces and abstract classes. Design by contract. Inheritance	2	1	

	versus Delegation. Loose coupling. Design pattern e.g. Singleton, Façade.			
10	**C++ Programming Language** Types and declarations. Pointers and arrays. Expressions and statements. Functions/Methods. Standard libraries. Transforming class specification into code.	4	2	
	Check for Hours	=26	=13	

Assessment (includes both continuous and summative assessment)

a) Final Examination: 60%
b) Clicker/Java Quiz (Participation): 5%
c) Laboratory Assessment: 5%
d) Assignment: 30%

CZ2003 – Computer Graphics & Visualization

Course Code	CZ2003
Course Title	Computer Graphics & Visualization
Pre-requisites	CZ1011: Engineering Mathematics I
No of AUs	3
Contact Hours	26 hours of lectures, 12 hours of tutorials, 10 hours of labs

Course Aims

This course aims to introduce methods for visualizing data, especially in cases where simple plots and diagrams are insufficient. Information visualization and graphics rendering of abstract data involves various mathematical models, and has become important in modern science and technology. It has applications across many disciplines including interior and architectural designs, as well as engineering software systems that employ graphical presentations as part of their user interfaces. You will be immersed into "visual mathematics", i.e. you will learn how to see geometry and colours beyond mathematical formulas and how to represent geometric shapes and motions by analytic functions and algorithmic procedures. You will develop spatial visualization skills which will permit you to perform 3D rotations and other transformations of geometric objects. You will also learn how a common personal computer, freeware and de-facto standard software can be used for solving complex computer graphics problems. The course requires knowledge of engineering math and analytic geometry.

Intended Learning Outcomes (ILO)

By the end of this course, you should be able to:

1. Use geometry and colors as ways to visualize data beyond mathematical formulas.
2. Represent geometric shapes and motions by analytical functions and algorithmic procedures.
3. Use spatial visualization skills to perform 3D rotations and other transformations and motions of geometric objects.
4. Use a common personal computer and open source software to solve complex computer graphics problems.

Course Content

	Topics	Lectures (Hours)	Tutorials (Hours)
1	**Introduction to computer graphics and foundation mathematics** Principles of visualisation. Coordinate systems. Time as another dimension. Vectors and matrices. Vector and matrix algebra with application to geometric coordinate space. Geometric meaning of dot and cross products. Surface normal calculation.	2	2
2	**Programming Computer Graphics and Visualization** Introduction to common computer graphics software used for solving engineering and information visualisation problems. Introduction to the software used in the coursework.	2	0
3	**Geometric shapes** Points, polygons voxels and procedural models. Analytical definitions of curves, surfaces, and solid objects. CSG. Blobby shapes (e-learning week). Rendering curves, surfaces and solids (e-learning week). Revision	10	5 (including one e-learning tutorial)

4	**2D Transformations** 2D Affine transformations. Translation, Scaling, Rotation. Composition of transformations.	2	1
5	**3D Transformations** 3D Affine transformations. Translation, Scaling, Rotation. Composition of transformations.	2	1
6	**Motions and Morphing** Motions by parametric functions and time-dependent affine transformations.	2	1
7	**Visual Appearance: Illumination** RGB colour model and light sources. Illumination calculation.	2	1
8	**Visual Appearance: Surface Mapping** Image texture mapping. 3D geometric textures. Appearance assignment to geometry.	2	1
9	**Efficient rendering** Hierarchical representation. Spatial partitioning. Bounding volumes. Level of detail. Revision.	2	1
	Check for Hours	=26	=13

Assessment (includes both continuous and summative assessment)

a) Final Examination: 60%
b) Continuous Assessment 1 (CA1) Tutorials: 10%
c) Continuous Assessment 2 (CA2) Software Laboratory: 30%

Note: The final examination is closed book.

CZ2004 – Human-Computer Interaction

Course Code	CZ2004							
Course Title	Human-Computer Interaction							
No of AUs	3							
Contact Hours	Lectures	26	TEL	0	Tutorials	12	Example Classes	10

Course Aims

This course aims to provide an introduction to human-computer interaction, with an overarching goal of inculcating into you the habit of adopting a user-centric perspective on usability when designing, evaluating and innovating new user interfaces. More specifically, the objectives are to get you to: (a) appreciate and understand the significance of considering usability issues in interface development, including user requirements, measurements and various usability tests; (b) acquire vocabulary to frame and articulate HCI issues and considerations for different computing applications; (c) learn first principles in user interface design and develop basic ability to apply design considerations to both current and future interface modalities; (d) obtain a perspective of how HCI needs to be aligned with human thought processes and physical abilities, and (e) be aware of the large range of user interfaces in society today, and appreciate how HCI design is applied in various sectors of the computing industry.

Intended Learning Outcomes (ILO)

Upon the successful completion of this course, you shall be able to:

1. Articulate the rationale for user-centric interface design;
2. Describe and apply methods and principles for creating and evaluating good user interfaces;
3. Implement important steps in a UI design process, with focus on creating lo-fidelity and high-fidelity design prototypes;
4. Elaborate on the different sensing and actuation modalities of both humans and existing user interfaces;
5. Use and explain various concepts and terminology that is widely used in the UI/UX community.

Course Content

	Topics	Lectures (Hours)	Tutorials (Hours)	Example Classes (Hours)
1	**Introduction** The origin, development, and significance of human-computer interaction. Case studies: successful and failure examples of software and hardware products involving HCI, novel devices etc.	2	0	0
2	**Usability and application spaces** Life critical, industrial, commercial, entertainment and sociotechnical application types. Influence of age, gender, intellect, language, personality, and disability (e.g., colour blindness). Case studies such as digital camera operations and satellite navigation.	4	2	0
3	**Guidelines and principles** Guidelines for navigation, organization, attention and data entry. Principles: Eight golden rules.	4	2	0
4	**Prototyping and evaluating interface designs** Expert review, Usability labs, acceptance testing, surveys, and on-going feedback. Case studies: various prototype techniques.	4	2	10
5	**Understanding humans, modelling users** Human senses, higher-level perception, attention; human memory, reasoning and problem solving, mental models, affect, personality traits; motor coordination, verbal and non-verbal communication	4	2	0
6	**Human-computer interfaces** Evolution of computing interfaces; keysets, handwriting, speech, pointing interfaces, sketching, natural motion, affective computing, brain-computer interface; 2D and 3D displays, audio, haptics, motion simulators, scent synthesis	4	2	0
7	**Interaction and design analysis** Major interaction metaphors (instruction, conversation, object and ego manipulation), peripheral feedback; anthropomorphism, software posture; affordances, metaphors, idioms; widgets, interface builders, design and pattern languages	4	2	0
	Check for Hours	=6	=12	10

Assessment (includes both continuous and summative assessment) CZ2004

a) Final Examination: 60%
b) Lo-fi prototype: 18%
c) Usability Evaluation report: 6%
d) Hi-fi prototype: 16%

CE/CZ 2005 – Operating Systems

Course Code	CE/CZ 2005							
Course Title	Operating Systems							
Pre-requisites	CE/CZ 1006: Computer Organisation and Structure and CE/CZ 1007: Data Structures							
No of AUs	3							
Contact Hours	Lectures	26	TEL	0	Tutorials	12	Laboratories	8

Course Aims

This course aims to develop your understanding of fundamental concepts and principles of modern operating systems, and to build your knowledge on the design and implementation of main operating system components.

Intended Learning Outcomes (ILO)

The course covers the following main components of an operating system: process management, memory management, file systems, and I/O. Upon the successful completion of this course, you shall be able to:

1. Explain fundamental concepts and principles of modern operating systems;
2. Identify the functions and services provided by each component of an operating system;
3. Describe how processes, memory, files, and I/O devices are managed in an operating system;
4. Analyze and evaluate algorithms that implement core functions of major operating system components; and
5. Implement functions for operating system components in accordance to design requirements using the concepts and principles learnt in this course.

Course Content

	Topics	Lectures (Hours)	Tutorials (Hours)
1	**Overview of Operating Systems (OS)** Functions, Types, Services, Evolution, Concepts, Structures. OS vs kernel vs scheduler. Batch vs real-time. From 'large' to 'small'. Influential legacy OS. UNIX, Linux, Windows and others.	2	1
2	**Processes and Threads** Process concept, operations on processes, inter-process communication. Concept of Thread, multi-threading models, relationship between processes and threads.	2	1
3	**Process Scheduling** Central Processing Unit (CPU) scheduling concepts, criteria and algorithms.	2	1
4	**Process Synchronization** The critical-section problem, mutual exclusion and condition synchronization, software/hardware solutions. Semaphores, Monitors, and classical synchronization problems.	4	2
5	**Deadlock and Starvation** Deadlock vs starvation, Characterization of deadlock prevention, avoidance, detection and recovery	2	1
6	**Memory Organization** Local vs physical address space, fixed-partitioning vs variable partitioning, Paging vs segmentation vs swapping.	2	1
7	**Virtual Memory Management** Virtual memory concepts, Page replacement algorithms, Frame allocation, and Issues related to demand paging.	4	2
8	**File System Organization and Implementation** File concept, access methods, structure and implementation. File allocation methods and disk space management. Directory structure and implementation.	4	1
9	**Input/Output (I/O) Management and Disk Scheduling** I/O software structure. Kernel I/O sub-system; Device drivers, Disk performance parameters. Disk scheduling policies.	2	1

10	**Issues in Real-time Operating Systems** Features of real-time kernels. Real-time CPU scheduling and real-time performance issues.	1	1
11	**Protection and Security** Goals of protection; Domain of protection; Protection models; Security problems and threats; Authentication; and Encryption	1	1
	Check for Hours	=26	=13

Assessment

a) Final Written Examination: 60%
b) Continuous Assessments: 40%

CE/CZ 2006 – Software Engineering

Course Code	CE/CZ 2006							
Course Title	Software Engineering							
Pre-requisites	CE/CZ 2002 (can be taken concurrently): Object Oriented Design and Programming							
No of AUs	3							
Contact Hours	Lectures	26	TEL	0	Tutorials	10	Student presentations	2

Course Aims

This course aims to equip SCSE students with foundation knowledge on issues and techniques required for the design and implementation of quality software.
You will have the necessary knowledge and skills to delve deeper into systems analysis (CZ3003) and advanced topics in Advanced Software Engineering (CZ3002).

Intended Learning Outcomes (ILO)

Upon completion of the course, you will be able to understand the roles and purposes of various activities in software engineering process. Specifically, you will be able to

1. Participate in all stages of the Software Development Life Cycle for a medium-size software project to deliver the required work products.
2. Elicit and specify requirements clearly and correctly.
3. Use good software design concepts and considerations.
4. Design and carry out test activities to verify that requirements have been met.
5. Perform simple project management of a medium-size software project.

Course Content

	Topics	Lectures (Hours)	Tutorials (Hours)	Presentations (Hours)
1	**Introduction to Software Engineering** Overview. Software Engineering Development Activities: stages and work products. Life Cycle Models: Waterfall, V, Spiral, RAD, Iterative, Agile. Case Study. UML Activity Diagram.	3	0	0
2	**Requirement Specification** Concepts: functional and non-functional requirements, software requirements. Elicitation: techniques, use case model. Specification: software requirement specification document, traceability. Validation: reviews, prototyping.	4	1	0
3	**Requirement Analysis** Conceptual Modelling: control, entity, interface objects. From Use Cases to Objects.	2	1	0
4	**Project Management** Overview. Concepts: work breakdown structure, task scheduling, and resource estimation. Activities: planning, organising, controlling, terminating. Tools.	4	2	0
5	**Design** Design Process: architectural design, detailed design, and software design document. Design Fundamentals Recap: Abstraction, Coupling and Cohesion, Decomposition and Modularisation, Encapsulation, Separation of Interface and Implementation. Key Issues: Concurrency, Event Handling, Error and Exception Handling, Separation of Business Logic from Interface, Data Persistence, and related Design Patterns (note that a few simple ones have been introduced in CE2002/CZ2002).	6	2	0
6	**Implementation and Testing** Mapping design models to code. Test case design: selection criteria, effectiveness. Levels: unit, integration, system. Strategies: object- oriented testing, white-box, black box. Formal Method: Finite State Machine. UML State Machine Diagram.	5	3	2
7	**Maintenance** Maintainability: reuse, version control.	2	1	0
	Check for Hours	=26	=10	=2

Assessment

a) Final Examination: 60%
b) Case study: 40%

CZ 2007 – Introduction to Databases

Course Code	CZ2007							
Course Title	Introduction to Databases							
Pre-requisites	CE/CZ2001: Algorithms							
No of AUs	3							
Contact Hours	Lectures	26	TEL	0	Tutorials	10	Student presentations	0

Course Aims

Database management systems (DBMS) are software systems that control the creation, maintenance, and use of databases, i.e., organized collections of data. Relational DBMS (RDBMS) are incredibly ubiquitous today -- they underlie technology used by most people every day if not every hour. RDBMS reside behind a huge fraction of websites; they are a crucial component of telecommunications systems, banking systems, video games, and just about any other software system or electronic device that maintains some amount of persistent information. As a consequence, it is important that we equip you with knowledge of design of relational databases and the use of RDBMS for applications. This introductory course serves the purpose.

Intended Learning Outcomes (ILO)

This course introduces relational databases at an elementary level. Upon the successful completion of this course, you shall be able to:

1. Explain the importance of, and uses for, databases within organizations;
2. Design a basic relational database management system (DBMS) for storing and analyzing datasets of medium complexity;
3. Formulate basic relational database queries and execute these in order to search and analyse underlying data;
4. Ensure data integrity through enacting the process of database normalization;
5. Describe the usage of indexing to improve query efficiency;
6. Explain the significance of XML and JSON in today's world;

Course Content

	Topics	Lectures (Hours)	Tutorials (Hours)	Presentations (Hours)
1	**Introduction to Databases:** Importance of data management, Overview of DBMS, DBMS Architecture, Relational DBMS, Physical and logical data independence, languages for databases, XML	1	0	0
2	**Entity-Relationship Data Model** Elements of E/R Model, Design Principles, Modelling of Constraints, Weak Entity Sets.	2	1	0
3	**Relational Data Model** Basics of Relational Model, E/R Diagram to Relational Design.	1	1	0
4	**Functional Dependencies (FD) and Normalization** Definition of Functional Dependency, Keys And Superkeys, Rules about FDs, Closure of FDs, Projecting FDs, Normal Forms.	5	3	0
5	**Relational Algebra** Algebra of Relational Operations, Relational Operations on Bags, Extended Operators.	3	1	0
6	**Querying Relational Databases** Introduction to SQL, Simple queries in SQL, 3-value logic, Multi-Relation Queries, Subqueries, Full-Relation Operations, Database Modifications, Database definition, Views, Constraints and Triggers, SQL in programming environment.	8	4	0
7	**Introduction to Database Security** Importance of secured data, levels of security, user authentication, access control mechanisms, SQL GRANT and REVOKE statements.	1	1	0
8	**Indexes** Motivation for indexes, Declaring indexes in SQL, Selection of Indexes	2	1	0

9	**Data Interchange Formats** **XML, JSON**	2	1	0
10	**Conclusion** **Summary of the course content, data management challenges ahead.**	1	0	0
	Check for Hours	=26	13	0

Assessment (includes both continuous and summative assessment) CZ2007

a) Final Examination: 50%
b) Term Project: 30%
c) Quiz: 20%

CE9010 – Introduction to Data Science

Course Code	CE9010					
Course Title	Introduction to Data Science					
Pre-requisites	CE/CZ1003, CE/CZ1011 (or equivalent - which is basics in statistics)					
No of AUs	3					
Contact Hours	Lecture	26	Tutorials	13	Laboratories	10

Course Aims

This course aims to develop skills in data science. In 2012, the Harvard Business School called data scientist the sexiest job of the 21st century. Why has data scientist become a most in-demand job? Today massive amounts of data are available in all areas of science, government and industry. Exploited sensibly, these raw data can significantly improve the efficiency of research, services and industries in as many fields as healthcare, engineering, finance, telecommunications or urban development just to name a few.

This course is designed for students who recognize the value of data science, but are not majoring from the School of Computer Science and Engineering (SCSE) and certain other programmes (see Appendix 3). The course provides you with an understanding of the fundamentals of data science and the practical skills set required to be a data scientist, including the design and implementation of basic techniques.

Intended Learning Outcomes (ILO)

By the end of this course, you (as a student) would be expected to be able to:

1. Identify and apply data analysis techniques to different data problems.
2. Implement data science techniques with Python.
3. Analyze and solve real-world data science projects.
4. Engage more professionally in written and oral scientific communication,
5. Work cohesively and effectively in teams.

Course Content

	Topics	Lectures (Hours)	Tutorials (Hours)	Laboratories (Hours)
1	**Introduction to Data Science** Why data science is essential? Big data era, powerful computational infrastructure, large IT companies invest in DS.	2	0	0
2	**Review of Linear Algebra and Coding Principles** Revise system of linear equations, standard solver, eigenvalue decomposition, Tikhonov regularization, and gradient descent technique. Review coding principles such as libraries, data structures, execution in Python.	2	2	0
3	**Linear Regression** Linear regression uses linear system of equations to solve real-valued prediction problem. Apply gradient descent technique to solve linear regression problems.	2	2	2
4	**Logistic Classification** Classification technique based on linear representation score and logistic loss function. Application to classification of images as cats and dogs.	2	1	0
5	**Support Vector Machine** Introduce SVM classification techniques from hard to soft class representations. Kernel trick for non-linear classification.	2	1	2
6	**Unsupervised Clustering** Present k-means technique for Gaussian data distribution. Discuss non-linear data grouping techniques such as Shi-Malik.	2	1	0
7	**Feature Extraction** Design Principal Component Analysis and Non-Negative Matrix Factorization. Application to face generative models.	2	1	2
8	**Recommender Systems** Introduce the two main classes of recommender systems; collaborative- and content-based filtering techniques.	2	1	0
9	**Neural Networks** Presentation of artificial neural networks, a.k.a. deep learning, with fully connected neural networks, learning process, convolutional neural networks and recurrent neural networks.	8	4	4
10	**Revision**	2		
	Check for Hours	= 26	= 13	= 10

Assessment (includes both continuous and summative assessment)
a) Final Examination: 40% b) Laboratory: 40% c) Project: 20% Note: Final examination is closed book.