COMMISSION FOR HIGHER EDUCATION

Credit Accumulation and Transfer System

Computer Science

Undergraduate

Assuring Quality Higher Education

September 2010

Commission for Higher Education

Credit Accumulation and Transfer System
Computer Science
Undergraduate

September 2010

CREDIT ACCUMULATION AND TRANSFER SYSTEM
COMPUTER SCIENCE
UNDERGRADUATE

Commission for Higher Education
Nairobi, Kenya, September 2010

ISBN: 978-9966-009-06-7

Commission for Higher Education (CHE)
P O Box 54999 – 00200
NAIROBI, KENYA.

Tel. No. +254 - 20 - 7205000, +254 - 20 - 2021151, Fax No. + 254 - 20 - 2021172
Web site : www.che.or.ke

TABLE OF CONTENTS

S/No. **Page No.**

FOREWORD

In the last twenty years, Kenya has witnessed rapid expansion of university education as a result of establishment of new public and private universities. Consequently, universities developed many academic programmes with some having similar contents, but taught under different names. Overtime this scenario brought challenges. Employers started to question the names and similarities associated with some programmes and secondly students could not transfer credits in the related programmes from one institution to another. This problem is common to all member states of the East African Common market.

To address this problem, the Higher education regulatory bodies of East African community member states, namely Commission for Higher Education (CHE) Kenya, National Council for Higher Education (NCHE) Uganda and Tanzania Commission for Universities (TCU) started the Credit Accumulation and Transfer System (CATS) project .The project was funded by Rockefeller Foundation. During this phase of the project, minimum core requirements in Human Medicine, Engineering, Basic Sciences and Agriculture were developed. After this phase it was decided that each country proceeds with the project by developing minimum core requirements in other fields of study in higher education.

Phase two of the project comprised the development of core requirements in the field of Business Studies and Information Technology/ Computer Science. Experts from the universities that offer these fields were invited by the Commission to develop the minimum core requirements. Universities offering these fields are expected to implement them and to provide feedback to the Commission for further refinements.

The Commission is committed to the CATS project in the development of minimum core requirements in the various fields of study offered by higher education institutions in the country. The objective of the CATS project is to regulate the mobility of students within institutions and ensure programme integration and harmonization of the higher education landscape in Kenya and to serve as input into the National Qualifications Framework. The success of the CATS project in Kenya is dependent on the cooperation between the Universities offering the fields of study and the Commission for Higher Education for coordination.

Prof. Everett M Standa, MBS
Commission Secretary/CEO
Commission for Higher Education
Nairobi, Kenya

2.0 INTRODUCTION

Computer Science spans a wide range, from its theoretical and algorithmic foundations to cutting-edge developments in robotics, computer vision, intelligent systems, bioinformatics, and other exciting areas. We can think of the work of computer scientists as falling into three categories. They design and implement software.

Computer scientists take on challenging programming jobs. They also supervise other programmers, keeping them aware of new approaches. They devise new ways to use computers. Progress in the Computer Science areas of networking, database, and human-computer-interface enabled the development of the World Wide Web.

Currently, Computer Science researchers are working with scientists from other fields to make robots become practical and intelligent aides, to use databases to create new knowledge, and to use computers to help decipher the secrets of our DNA. They develop effective ways to solve computing problems. Computer scientists develop the best possible ways to store information in databases, send data over networks, and display complex images. Their theoretical background allows them to determine the best performance possible, and the study of algorithms helps them to develop new approaches that provide better performance.

Computer Science spans the range from theory through programming. Computer Science offers a comprehensive foundation that permits graduates to adapt to new technologies and new ideas.

3.0 GROUP MEMBERS

Prof. Daniel Rotich,	-	Moi University
Dr. Joseph Sevilla,	-	Strathmore University
Dr Waweru Mwangi,	-	Jomo Kenyatta University of Agriculture and Technology
Mr. Emmanuel Kweyu,	-	Strathmore University
Mr. John Konyino,	-	Maseno University
Mr. Humphrey Kilwake,	-	Masinde Muliro University of Science and Technology
Mr. Kennedy Waweru	-	St. Pauls'University

4.0 ADMISSION REQUIREMENTS

A candidate must satisfy any of the following minimum requirements:

a) Be a holder of KSCE Mean Grade of C+ and above, with at least a C+ in Mathematics and Physics/Physical Science;

b) Be a holder of KACE with at least 2 Principles passes in Mathematics and Physics and 1 subsidiary pass; or

c) Be a holder of Validated Diploma in Computer Science, Information Technology or a related field with at least a credit pass or related work experience.

5.0 MINIMUM CREDIT TRANSFER

Credits are transferable within 5 years of obtaining the results. A student registered in one university must have taken at least 51% of the courses at this university to obtain a degree from the same. The remaining 49% can be courses transferred from other universities. Students must complete all the pre-requite courses prior to credit transfer. To qualify for credit transfer a student must have obtained at least a grade of C in the given course.

TABLE 1: GRADING SYSTEM

Marks Range	Cluster/grade	GPA
69.5% and above	A	4.00
59.5%-69.4%	B	3.00
49.5%-59.4%	C	2.00
40%-49.4%	D	1.00
Below 40 %	Fail	0.00

6.0 OVERALL GOAL OF PROGRAMME

The overall goal is to produce a computer science graduate capable of using computing principles, concepts and techniques to design, implement, manage and maintain computing systems that address and provide solutions to various economic activities for benefit of humankind. The programme also seeks to prepare students for lifelong learning that will enable them to move beyond today's technology to meet the challenges of the future.

7.0 PROGRAMME DURATION AND STRUCTURE

The minimum number of credit units required to complete the programme is 168 credit units, which is equivalent to 2352 lecture hours. However, for one to obtain a Bachelor of Science degree in Computer Science, he/she must have taken 120 compulsory credit units, which is equivalent to 1680 lecture hours, as provided in this curriculum. The remaining units are left to the discretion of the Institution offering the programme.

1 credit unit (CU) is equivalent to:

a) 14 lecture hours;

b) 28 hours of tutorials; or

c) 42 hours of Practicals.

8.0 PROGRAMME LEARNING OUTCOMES

The graduate of this programme should be able to:

1. Demonstrate knowledge and understanding of essential facts, concepts, principles, and theories relating to computer science and software applications;

2. Model, specify, design and implement computer-based systems;

3. Make sound decision in the process of understanding product development guided by the social, professional, legal, ethical and cultural issues;

4. Effectively apply the principles of information management, organization, and retrieval;

5. Apply the process of critical reasoning in managing computer systems and information; and

6. Be a team player and communicate clearly to a range of audiences about technical problems and their solutions.

9.0 PROGRAMME LEARNING OUTCOMES AGAINST COURSES PER YEAR

Various courses, which are distributed in a progressive manner in the various years of study, contribute to the realization of the programme learning outcomes as summarized in Table 2.

TABLE 2: PROGRAMME LEARNING OUTCOMES AGAINST COURSES PER YEAR

	COURSES PER YEAR			
PROGRAM LEARNING OUTCOMES	**YEAR 1**	**YEAR 2**	**YEAR 3**	**YEAR 4**
	Courses	Courses	Courses	Courses
PLO 1	• Electronics • Introduction to programming • Discrete Structures1 • Linear algebra • Fundamentals of computing • Object Oriented programming 1 • Discrete Structures2 • Basic Calculus • Digital Electronics 1 • Introduction to Internet Technologies	• Object Oriented programming2 • Data Structures • Probability and statistics • Digital Electronics 2 • Principles of Operating systems • Systems Analysis and design • Computer Organisation and architecture 1 • Assembly language programming • Database systems I • Application Development for the Internet • Data Communications • Automata Theory	• Principles of programming languages • Design and Analysis of algorithms • Compiler Construction and design • Computer networks • Human Computer Interaction • Computer Systems Security • Mobile Computing • Software Engineering • Computer Graphics • Research Methods and Technical Writing	• Software Project management • Distributed systems • Management information systems • IT and Society • Entrepreneurship • Legal and ethical issues in Computing • C S project

	COURSES PER YEAR			
PROGRAM LEARNING OUTCOMES	YEAR 1	YEAR 2	YEAR 3	YEAR 4
	Courses	Courses	Courses	Courses
PLO 2	• Introduction to programming • Discrete Structures1 • Linear algebra • Object Oriented programming 1 • Discrete Structures2 • Digital Electronics 1	• Object Oriented programming2 • Data Structures • Digital Electronics 2 • Systems Analysis and design • Assembly language programming • Database systems I • Application Development for the Internet	• Principles of programming languages • Design and Analysis of algorithms • Compiler Construction and design • Computer Networks • Human Computer Interaction • Computer Systems Security • Mobile Computing • Software Engineering • Computer Graphics	• Software Project management • Distributed systems • Management information systems • C S project
PLO 3			• Mobile Computing • Software Engineering • Research Methods and Technical Writing	• Software Project management • Management information systems • IT and Society • Entrepreneurship • Legal and ethical issues in Computing • C S project

	COURSES PER YEAR			
PROGRAM LEARNING OUTCOMES	**YEAR 1**	**YEAR 2**	**YEAR 3**	**YEAR 4**
	Courses	Courses	Courses	Courses
PLO 4	• Object Oriented programming 1	• Object Oriented programming2 • Data Structures • Principles of Operating systems • Systems Analysis and design • Database systems I • Application Development for the Internet	• Design and Analysis of algorithms • Human Computer Interaction • Computer Systems Security • Mobile Computing • Software Engineering	• Software Project management • Management information systems • C S project
PLO 5	• Introduction to programming • Object Oriented programming 1 • Basic Calculus	• Object Oriented programming2 • Data Structures • Probability and statistics • Systems Analysis and design • Database systems I • Application Development for the Internet	• Design and Analysis of algorithms • Human Computer Interaction • Computer Systems Security • Software Engineering	• Software Project management • Management information systems • C S project

	COURSES PER YEAR			
PROGRAM LEARNING OUTCOMES	**YEAR 1**	**YEAR 2**	**YEAR 3**	**YEAR 4**
	Courses	Courses	Courses	Courses
PLO 6	• Introduction to programming • Object Oriented programming 1 • Digital Electronics 1	• Object Oriented programming2 • Data Structures	• Software Engineering • Research Methods and Technical Writing	• Software Project management • IT and Society • Entrepreneurship • Legal and ethical issues in Computing • C S project

10.0 CORE PROGRAMME COURSES

The core courses for the Bachelor of Science in Computer Science degree are summarized per year in Table 3.

TABLE 3: CORE COURSESFOR THE PROGRAMME

COURSES	Credit Units
YEAR ONE	
Electronics	3
Introduction to programming	3
Discrete Structures I	3
Linear algebra	3
Fundamentals of computing	3
Object Oriented programming I	3
Discrete Structures II	3
Basic Calculus	3
Digital Electronics 1	3
Introduction to Internet Technologies	3
YEAR TWO	
Object Oriented programming II	3
Data Structures	3
Probability and statistics	3
Digital Electronics II	3
Principles of Operating systems	3
Systems Analysis and design	3
Computer Organisation and architecture 1	3
Assembly language programming	3
Database systems I	3
Application Development for the Internet	3
Data Communications	3
Automata Theory	3
YEAR THREE	
Principles of programming languages	3

Design and Analysis of algorithms	3
Compiler Construction and design	3
Computer networks	3
Human Computer Interaction	3
Computer Systems Security	3
Mobile Computing	3
Software Engineering	3
Computer Graphics	3
Research Methods and Technical Writing	3
YEAR FOUR	
Software Project management	3
Distributed systems	3
Management information systems	3
Computer Science project	6
IT and Society	3
Entrepreneurship	3
Legal and ethical issues in Computing	3

11.0 COURSE DESCRIPTIONS

	FUNDAMENTALS OF COMPUTING
Credit Units	3
Pre-requisite	None
Purpose of the course	To introduce different parts of the computer and its applications to computing and demystify the way a computer system works and understanding its underlying theories and concepts, history and basic data communication of a computer
Expected Learning Outcomes	At the end of the course, the students should be able to: 1. Explain the basic principles of computer communication through number systems, data representations and computer networks; 2. Discuss the evolution of computers, its different parts and integration; 3. Relay the important role of the computer in today's business and society; 4. Develop an early interest in lifelong learning; 5. Write and simulate algorithms; 6. Draw and analyze flowcharts; 7. Troubleshoot basic computer problems and install operating system; 8. Assemble and disassemble computers parts; 9. Actual practical in the labs.

Content	History and classification of computers; different number systems; memory and storage; input/output peripherals discussion and familiarizations; algorithms; flow charting, introduce different programming languages; operating system installation; basic computer troubleshooting; assembly and disassembly; basic networking theories and concepts; internet.	
Learning and Teaching Methodologies	PC assembly and troubleshooting demonstrations, lectures using powerpoint, assignments, exams and recitations	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100

Recommended Reading	TITLE	AUTHOR	PUBLISHER	ISBN
	1. Data Processing and information technology 10th Ed	C. S. French	DP Publications	185805-171-1
	2. Computers and information systems 2nd ed,	T. J. O'Leary,	Benjamin/ Cummings,	0-8053-6942-2
	3. A Balanced Introduction to Computer Science	David Reed	Prentice Hall. 2004	013046709X
	4. Fundamentals of computing	Rowntree, G.	Manchester NCC publications	0-85012-661-4

Support Materials and Resources	Texts, audio and video cassettes, computer software Other resources: 1. http://www.daileyint.com/hmdpc/manual.htm 2. http://www.internet4classrooms.com/support.htm 3. http://www.pcguide.com/ts/index.htm

COURSE NAME	**INTRODUCTION TO PROGRAMMING**
Credit Units	3
Pre-requisite	Fundamentals of Computing
Purpose of the Course	To impart an understanding of fundamental concepts underlying programming and problem solving.

<table>
<tr><td><b>Expected Learning Outcomes</b></td><td colspan="2">At the end of the course the students will be able to:
1. Write, compile, and run C/Pascal programs on a computer;
2. Write programs involving sequence, selection, and iteration operators;
3. Read an existing program and determine its output;
4. Discuss the fundamental concepts of object–oriented programming;
5. Discuss the theory of data structures and algorithms;
6. Identify the necessary properties of good algorithms;
7. Create algorithms for solving simple problems;
8. Use a procedural programming language to implement, test, and debug algorithms for solving simple problems.</td></tr>
<tr><td><b>Course Content</b></td><td colspan="2">History and overview of programming languages; overview of programming paradigm; structured programming: problem solving techniques, algorithms, pseudo code; basic syntax and semantics of a higher-level language, data types, expressions, statements, input/output, control structures, data structures; basic sequencing, alternation, and looping control constructs; subprograms: functional and procedural abstractions and data abstraction, files</td></tr>
<tr><td><b>Mode of delivery</b></td><td colspan="2">Lectures, directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.</td></tr>
<tr><td><b>Instructional Material and/or Equipment</b></td><td colspan="2">Audi visual equipment, chalkboard, computer simulation software, computer programming tools</td></tr>
<tr><td rowspan="4"><b>Course Assessment</b></td><td>Type</td><td>Weighting (%)</td></tr>
<tr><td>Examination</td><td>60</td></tr>
<tr><td>Continuous Assessment</td><td>40</td></tr>
<tr><td>Total</td><td>100</td></tr>
<tr><td><b>Recommended reference material</b></td><td colspan="2">1. You Can Do It: A Beginner's Introduction to Computer Programming, Francis Glassborow, John Wiley &amp; Sons (2003), ISBN: 0470863986, 978-0470863985
2. Beginning Programming for Dummies, 4th Edition, Wallace Wang, John Wiley &amp; Sons (2006), 0470088702, 978-0470088708</td></tr>
</table>

COURSE NAME	**OBJECT ORIENTED PROGRAMMING I**
Credit Units	3
Pre-requisite	Introduction to Computer Programming
Purpose of the Course	To impart an understanding of fundamental concepts underlying Object oriented programming paradigm

Expected Learning Outcomes

At the end of the course the students will be able to:

1. Demonstrate understanding of object technology and its applications;
2. Explain the main principles of good design;
3. Design and develop programs in an object-oriented language using abstract data types;
4. Design well-structured algorithms and code them in an object oriented programming language;
5. Describe the application of a variety of data structures;
6. Discuss the advantages and disadvantages of various data structures;
7. Demonstrate mastery of object oriented programming concepts such as inheritance, polymorphism and operators.

Course Content

Introduction to general principles underlying the practice of object-oriented programming; primitive data types; variables, constants, operands and operators; Input/output; floating point input; control structures; logical; file I/O; arrays: one-dimensional, two-dimensional; separation of behavior and implementation; objects, classes and subclasses; compound types; control loops; pointers; function; abstraction; Encapsulation; inheritance and polymorphism; virtual function

Mode of delivery

Lectures, directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.

Instructional Material and/or Equipment

Audi visual equipment, chalkboard, computer simulation software, computer programming tools

Course Assessment

Type	Weighting (%)
Examination	60
Continuous Assessment	40
Total	100

COURSE NAME **OBJECT ORIENTED PROGRAMMING II**

Credit Units 3

Pre-requisite

1. Object Oriented Programming I
2. Data structures and algorithms

Purpose of the Course The purpose of this course is to ground students to advanced concepts in object-oriented programming and in writting more difficult/complex programs

Expected Learning Outcomes At the end of the course the students will be able to:

1. Use important features of an object-oriented language, which include exception handling, polymorphism, operator overloading and inheritances;
2. Develop multi-thread applications.

Course Content Hiding the Implementation; initialization and cleanup; passing and returning objects, dynamic object creation; reusing classes: inheritance and composition; advanced Object Oriented Programming: interfaces inner classes, polymorphism and inheritance, memory management internals, handling exception and safety; generic programming: run-time type identification, templates and partial specialization, design patterns, proxy classes, multiple dispatch, operator overloading; collections of objects: arrays, container, iterators; I/O system with streams; concurrency; coding guidelines, programming guidelines.

Mode of delivery Lectures , directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.

Instructional Material and/or Equipment Audi visual equipment, chalkboard, computer simulation software, computer programming tools

Course Assessment

Type	Weighting (%)
Examination	60
Continuous Assessment	40
Total	100

COURSE NAME	**DISCRETE STRUCTURES I**
Credit Units	3
Pre-requisite	Mathematical preparation sufficient to take calculus at the college/university level
Purpose of the Course	Introduces the foundations of discrete mathematics as they apply to computer science, focusing on providing a solid theoretical foundation for further work. Although the principal focus is discrete mathematics, the course is likely to be more successful if it highlights applications whose solutions require proof, logic, and counting. For example, the number theory section could be developed in the context of public-key cryptography, so that students who tend to focus on the applications side of computer science will have an incentive to learn the underlying theoretical material.

Expected Learning Outcomes	At the end of the course students will be able to: 1. Design simple logic circuits; 2. Describe digital logic elements; 3. Discuss fundamental structures; 4. Describe basic Boolean algebra; 5. Discuss propositional logic, basic counting, and elementary number theory.	
Course Content	Introduction to logic and proofs: Direct proofs; proof by contradiction; mathematical induction. Fundamental structures: Functions (surjections, injections, inverses, composition); relations (reflexivity, symmetry, transitivity, equivalence relations); sets (Venn diagrams, complements, Cartesian products, power sets); pigeonhole principle; cardinality and countability. Boolean algebra: Boolean values; standard operations on Boolean values; de Morgan's laws. Propositional logic: Logical connectives; truth tables; normal forms (conjunctive and disjunctive); validity. Digital logic: Logic gates, flip-flops, counters; circuit minimization. Elementary number theory: Factorability; properties of primes; greatest common divisors and least common multiples; Euclid's algorithm; modular arithmetic; the Chinese Remainder Theorem. Basics of counting: Counting arguments; pigeonhole principle; permutations and combinations; binomial coefficients	
Mode of delivery	Lectures, tutorials, and seminars.	
Instructional Material and/or Equipment	Computers, Learning Management System, Writing boards, writing materials, projectors etc	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100
Core Reading Material for the course	1. Molluzo and Buckley - A First Course in Discrete Mathematics, Any publisher or Coursepak; 2. L. Lovasz, J. Pelikan, K. Vestergombi - "Discrete Mathematics", Springer 2003	
Recommended reference material	1. Stanat and McAllister - Discrete Mathematics in Computer Science. 2. Schaum's Discrete Mathematics 3. Cormen, Leiserson, and Rivest - Algorithms	

COURSE NAME	**DISCRETE STRUCTURES II**	
Credit Units	3	
Pre-requisite	Mathematical preparation sufficient to take calculus at the college/ university level	
Purpose of the Course	This course advances understanding of mathematical topics in the context of applications.	
Expected Learning Outcomes	At the end of the course students will be able to: 1. Predicate logic; 2. Apply the concepts of matrices in solving computational problems; 3. Apply computational complexity, elementary computability and discrete probability to mathematical problems; 4. Interpret and apply graphs and trees.	
Course Content	Review of previous course. Predicate logic: Universal and existential quantification; modus ponens and modus tollens; limitations of predicate logic. Recurrence relations: Basic formulae; elementary solution techniques. Graphs and trees: Fundamental definitions; simple algorithms; traversal strategies; proof techniques; spanning trees; applications. Matrices: Basic properties; applications. Computational complexity: Order analysis; standard complexity classes. Elementary computability: Countability and uncountability; diagonalization proof to show uncountability of the reals; definition of the P and NP classes; simple demonstration of the halting problem. Discrete probability: Finite probability spaces; conditional probability, independence, Bayes' rule; random events; random integer variables; mathematical expectation.	
Mode of delivery	Lectures, tutorials, and seminars.	
Instructional Material and/or Equipment	Computers, Learning Management System, Writing boards, writing materials, projectors etc	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100
Core Reading Material for the course	1. Molluzo and Buckley - A First Course in Discrete Mathematics, Any publisher or Coursepak. 2. L. Lovasz, J. Pelikan, K. Vestergombi - "Discrete Mathematics", Springer 2003	
Recommended reference material	1. Schaum's Discrete Mathematics 2. Randall E. Bryant and David O'Hallaron - Computer Systems: A Programmers Perspective,, Prentice-Hall 3. Morris de Groot - Probability and Statistics	

COURSE NAME	**PROGRAMMING LANGUAGES**
Credit Units	3
Pre-requisite	1. Compiler Design 2. Data Structures 3. Object Oriented Programming
Purpose of the Course	To enhance understanding on the different styles of programming promoted by different languages, the variety of programming languages and the design tradeoffs between the different programming paradigms.
Expected Learning Outcomes	At the end of the course the students will be able to: 1. Summarize the evolution of programming languages illustrating how this history has led to the paradigms available today; 2. Identify distinguishing characteristics of programming paradigms ; 3. Evaluate the tradeoffs between the different paradigms; 4. Distinguish between programming-in-the-small and programming-in-the-large.
Course Content	Overview of programming languages: History of programming languages. Programming language semantics: Informal semantics; Overview of formal semantics; Denotational semantics; Axiomatic semantics; Operational semantics. Programming language design: General principles of language design; Design goals; Typing regimes; Data structure models; Control structure models; Abstraction mechanisms. Programming paradigms: Procedural languages; Object-oriented languages; Functional languages; Declarative non-algorithmic languages, Scripting languages. Virtual machines: The concept of a virtual machine; Hierarchy of virtual machines; Intermediate languages; Security issues arising from running code on an alien machine. Introduction to language translation: Comparison of interpreters and compilers; Language translation phases; lexical analysis, parsing, code generation, optimization; Machine-dependent and machine-independent aspects of translation.
Mode of delivery	Lectures, directed reading, Group/class discussions and practical exercises
Instructional Material and/or Equipment	Audi visual equipment, Computers, writing boards, writing materials, projectors etc
Course Assessment	Assignments, tutorials, tests, practical exercises and written examinations.
Core Reading Material for the course	1. Sethi - Programming Languages: Concepts and Constructs. 2. Tennent - Principles of Programming Languages.
Recommended reference material	Hopcroft and Ullman - Introduction to Automata Theory, Languages, and Computation.

COURSE NAME	**DESIGN AND ANALYIS OF ALGORITHMS**
Credit Units	3
Pre-requisite	1. Discrete Structures 2. Data Structures and Algorithms 3. Introduction to Programming
Purpose of the Course	The study of algorithms provides insight into the intrinsic nature of the problem as well as possible solution techniques independent of programming language, computer hardware, or any other implementation aspect. This course is designed to provide students with an understanding of the principles and techniques used in the design and analysis of computer algorithms. They learn to apply these to efficiently solve real problems, and to determine which problems are intractable.
Expected Learning Outcomes	At the end of the course the students will be able to: 1. Analyze algorithm performance using complexity measurement; 2. Implement an algorithm using major algorithms design techniques such as divide and conquer, greedy and dynamic programming;. 3. Solve a variety of practical problems including sorting and selection, graph problems, and other optimization problems; 4. Evaluate algorithms; 5. Implement algorithms in a programming context; 6. Design and implement an appropriate hashing function for an application and a collision-resolution algorithm for a hash table; 7. Elaborate the theory of NP-completeness.
Course Content	Review of proof techniques. Basic algorithmic analysis: Asymptotic analysis of upper and average complexity bounds; best, average, and worst case behaviours; big-O, little-o, Ω, and Ø notation; standard complexity classes; empirical measurements of performance; time and space tradeoffs in algorithms; using recurrence relations to analyze recursive algorithms. Fundamental algorithmic strategies: Brute-force; greedy; divide-and-conquer; backtracking; branch-and-bound; heuristics; pattern matching and string/text algorithms; numerical approximation algorithms. Fundamental Algorithms: Simple numerical algorithms; Sequential and binary search algorithms; Quadratic sorting algorithms (selection, insertion); O(N log N) sorting algorithms; Quicksort, heapsort, mergesort; Hash tables, including collision-avoidance strategies; Binary search trees; Representations of graphs (adjacency list, adjacency matrix); Depth-and breadth-first traversals; Shortest-path algorithms (Dijkstra's and Floyd's algorithms); Transitive closure (Floyd's algorithm); Minimum spanning tree (Prim's and Kruskal's algorithms); Topological sort; Distributed algorithms[3]: Concurrency; Scheduling; Fault tolerance; Basic Computability: Finite-state machines; Context-free grammars; Tractable and intractable problems; Definition of the classes P and NP; NP-completeness (Cook's theorem); Standard NP-complete problems; Uncomputable functions; The halting problem; Implications of uncomputability.

Mode of delivery	Lectures, directed reading, practical demonstrations of simulation applications, and hands-on laboratory sessions and projects.	
Instructional Material and/or Equipment	Computers, Learning Management System, writing boards, writing materials, projectors.	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100
Core Reading Material for the course	1. Anany Levitin - Introduction to The Design and Analysis of Algorithms 2. Cormen, Leiserson, Rivest, and Stein - Introduction to Algorithms, Third Edition, 2009	
Recommended reference material	1. Cormen, Leiserson, and Rivest - Algorithms. 2. Jon Kleiberg – Algorithm Design	

COURSE NAME	**DATA STRUCTURES AND ALGORITHMS**
Credit Units	3
Pre-requisite	1. Procedural Programming 2. Object-Oriented Programming
Purpose of the Course	This course unit is designed to focuses on implementation and mathematical analysis of fundamental data structures and algorithms.
Expected Learning Outcomes	At the end of the course the students will be able to: 1. Describe the representation of numeric and character data; 2. Explain the effects of precision and round-off on numeric calculations; 3. Discuss the use of primitive data types and built-in data structures; 4. Describe common applications for each data structure in the topic list; 5. Implement the user-defined data structures in a high-level language; 6. Compare alternative implementations of data structures with respect to performance; 7. Write programs that use each of the following data structures: arrays, strings, linked lists, stacks, queues, and hash tables; 8. Compare and contrast the costs and benefits of dynamic and static data structure implementations.

Course Content	Representation of numeric data; Range, precision, and rounding errors; Arrays; Representation of character data; Strings and string processing; Runtime storage management; Pointers and references; Linked structures; Implementation strategies for stacks, queues, and hash tables; Implementation strategies for graphs and trees; Strategies for choosing the right data structure. Recursion: Recursive mathematical functions, Simple recursive functions. Introduction to algorithm analysis. Non Linear Structures: Trees; Binary Trees, Binary Search Trees, Binary Heaps. Abstract Data Types: stacks, queues, lists, Priority Queues. Introduction to Graphs. Applications of: Huffman codes, Heapsort, TreeSort, Linear Search, Binary Search etc	
Mode of delivery	Lectures , directed reading, Group/class discussions and practical exercises	
Instructional Material and/or Equipment	Audi visual equipment, Computers, writing boards, writing materials, projectors etc	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100
Core Reading Material for the course	Cormen, Leiserson, and Rivest - Algorithms	
Recommended reference material	Anany Levitin - Introduction to The Design and Analysis of Algorithms	

COURSE NAME	**ADVANCED ALGORITHMS**
Credit Units	3
Pre-requisite	1. Design and Analysis of Algorithms 2. Introduction to Programming
Purpose of the Course	This course advances students understanding of techniques for the design and analysis of algorithms, and explores a variety of applications.
Expected Learning Outcomes	At the end of the course the students will be able to: 1. Use the potential method to provide an amortized analysis of previously unseen data structure, given the potential function; 2. Explain why competitive analysis is an appropriate measure for online algorithms; 3. Explain the use of randomization in the design of an algorithm for a problem where a deterministic algorithm is unknown or much more difficult;. 4. Create simple extensions of cryptographic protocols, using known protocols and cryptographic primitives; 5. Describe implementation of linked lists on a PRAM; 6. Explain Brent's theorem and its relevance.

<table>
<tr><td><b>Course Content</b></td><td colspan="2">Advanced Analysis: Amortized analysis; Online and offline algorithms; Randomized algorithms; Dynamic programming; Combinatorial optimization. Cryptographic Algorithms: Historical overview of cryptography; Private-key cryptography and the key-exchange problem; Public-key cryptography; Digital signatures; Security protocols; Applications (zero-knowledge proofs, authentication, and so on). Geometric Algorithms: Line segments; properties, intersections; Convex hull finding algorithms. Parallel Algorithms: PRAM model; Exclusive versus concurrent reads and writes; Pointer jumping; Brent's theorem and work efficiency.</td></tr>
<tr><td><b>Mode of delivery</b></td><td colspan="2">Lectures, directed reading, Group/class discussions and practical exercises</td></tr>
<tr><td><b>Instructional Material and/or Equipment</b></td><td colspan="2">Audi visual equipment, Computers, writing boards, writing materials, projectors etc</td></tr>
<tr><td rowspan="4"><b>Course Assessment</b></td><td>Type</td><td>Weighting (%)</td></tr>
<tr><td>Examination</td><td>60</td></tr>
<tr><td>Continuous Assessment</td><td>40</td></tr>
<tr><td>Total</td><td>100</td></tr>
<tr><td><b>Recommended reference material</b></td><td colspan="2">1. Vijay Vazirani, <i>Approximation Algorithms</i>, Springer, 2001.
2. Dorit S. Hochbaum (ed.), <i>Approximation Algorithms for NP-hard Problems</i>, PWS Publishing, 1997.
3. Rajeev Motwani and Prabhakar Raghavan, <i>Randomized Algorithms</i>, Cambridge University Press, 2000.
4. Michael Kearns and Umesh Vazirani, <i>An Introduction to Computational Learning Theory</i>, The MIT Press, 1994.
5. Allan Borodin and Ran El-Yaniv, <i>Online Computation and Competitive Analysis</i>, Cambridge University Press, 2005.
6. Jon Kleinber and Eva Tardos, <i>Algorithm Design</i>, Addison-Wesley, 2006.
7. T. Cormen, C. Leiserson, R. Rivest, and C. Stein, <i>Introduction to Algorithms</i>, 2nd edition, 2001.</td></tr>
</table>

COURSE NAME	**HUMAN COMPUTER INTERACTION**
Credit Units	3
Pre-requisite	1. Programming Fundamentals; or 2. Fundamentals of Web Systems
Purpose of the Course	This course emphasizes on the understanding of human reactions to displays of various kinds and on human behavior in the context of interactive objects. Based on this, students' understanding of the principles associated with the evaluation of interfaces including those that embody interaction will be enhanced. Students will be exposed to the principles and guidelines that reflect best practice in the design, development, and maintenance of interfaces for multiple types of systems.

<table>
<tr><td rowspan="1"><b>Expected Learning Outcomes</b></td><td colspan="2">At the end of the course, students should be able to:
1. Design, implement and evaluate effective and usable graphical computer interfaces;
2. Describe and apply core theories, models and methodologies from the field of HCI;
3. Describe and discuss current research in the field of HCI;
4. Implement simple graphical user interfaces using the Java Swing toolkit;
5. Describe special considerations in designing user interfaces for users with special needs.</td></tr>
<tr><td><b>Course Content</b></td><td colspan="2">Foundations: Motivation; Contexts for HCI; Process for user-centered development; early focus on users, empirical testing, iterative design; Different measures for evaluation; utility, efficiency, learnability, user satisfaction; Models that inform human-computer interaction (HCI) design; attention, perception and recognition, movement, and cognition; Social issues influencing HCI design and use; culture, communication, and organizations; Accommodating human diversity, including universal design and accessibility and designing for multiple cultural and linguistic contexts;</td></tr>
<tr><td></td><td colspan="2">User interface standards. Building GUI Interfaces: Principles of graphical user interfaces (GUIs); Action-object versus object-action; User interface events; User-interface for a native system vs. the web. User Centered Software Evaluation: Evaluation without typical users; walkthroughs, KLM, expert-based analysis, heuristics, guidelines, and standards; Evaluation with typical users; observation, think-aloud, interview, survey, experiment; Challenges to effective evaluation; sampling, generalization; Reporting the results of evaluations. User Centered Software Development. GUI Design: Choosing interaction styles and interaction techniques; Choosing the right widget for users and tasks; HCI aspects of screen design; layout, color, fonts, labeling; Handling human/system failure; Beyond simple screen design; visualization, representation, metaphor; Multi-modal interaction; graphics, sound, and haptics; 3D interaction and virtual reality; Designing for small devices; Multi-cultural interaction and communication. GUI Programming: UIMS, dialogue independence and levels of analysis, Seeheim model; Widget classes and libraries; Event management and user interaction; Web design vs. native application design; Geometry management; GUI builders and UI programming environments; Cross-platform design; Design for small, mobile devices</td></tr>
<tr><td><b>Mode of delivery</b></td><td colspan="2">Lectures , directed reading, Group/class discussions and practical exercises</td></tr>
<tr><td><b>Instructional Material and/or Equipment</b></td><td colspan="2">Audi visual equipment, Computers, writing boards, writing materials, projectors etc</td></tr>
<tr><td rowspan="4"><b>Course Assessment</b></td><td>Type</td><td>Weighting (%)</td></tr>
<tr><td>Examination</td><td>60</td></tr>
<tr><td>Continuous Assessment</td><td>40</td></tr>
<tr><td>Total</td><td>100</td></tr>
</table>

Core Reading Material for the course	Alan Dix et al, *Human-Computer Interaction*, Third Edition by Prentice Hall (2004)
Recommended reference material	1. Rosson, M. and Carroll, J *Usability Engineering: Scenario-Based Development of Human-Computer Interaction.* 2. Nielsen, J. *Usability Engineering*

COURSE NAME	**FUNCTIONAL PROGRAMMING**	
Credit Units	3	
Pre-requisite	Object-Oriented Programming	
Purpose of the Course	The purpose of the course is to expose students to various aspects of functional programming and the use of specialized ideas developed within them in other programming paradigms	
Expected Learning Outcomes	At the end of course the students will be able to: 1. Outline the strengths and weaknesses of the functional programming paradigm; 2. Design, code, test, and debug programs using the functional paradigm; 3. Explain the use of functions as data, including the concept of closures.	
Course Content	Overview and motivation of functional languages; Recursion over lists, natural numbers, trees, and other recursively-defined data; Pragmatics (debugging by divide and conquer; persistency of data structures); Amortized efficiency for functional data structures; Closures and uses of functions as data: infinite sets, streams.	
Mode of delivery	Lectures, directed reading, Group/class discussions and practical exercises	
Instructional Material and/or Equipment	Audi visual equipment, Computers, writing boards, writing materials, projectors etc	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100
Recommended reference material	1. Simon Thompson, *Haskell: The Craft of Functional Programming*, Second Edition. Addison-Wesley, 1999. 2. J.-Y. Girard, Y. Lafont and P. Taylor, *Proofs and Types*, Cambridge University Press 3. Robert Harper, *Programming in Standard ML* 4. Simon Peyton Jones and David Lester, *Implementing Functional Languages: A Tutorial*.	

<table>
<tr><td><b>COURSE NAME</b></td><td colspan="2"><b>BASIC CALCULUS</b></td></tr>
<tr><td><b>Credit Units</b></td><td colspan="2">3</td></tr>
<tr><td><b>Pre-requisite</b></td><td colspan="2"></td></tr>
<tr><td><b>Purpose of the Course</b></td><td colspan="2">The course aims at introducing fundamental mathematical concepts and properties in calculus which are applicable in Information Technology</td></tr>
<tr><td><b>Expected Learning Outcomes</b></td><td colspan="2">At the end of the course the students will be able to:
1. Resolve problems involving trigonometric functions;
2. Apply properties of functions and equation;
3. Apply basic manipulations to functions such as differentiation and integration;
4. Apply properties of differentiation and integration in solving problems.</td></tr>
<tr><td><b>Course Content</b></td><td colspan="2">Trigonometric functions: addition, multiple angle and factor formulae; Limits, continuity and differentiability. Differentiation by first principles and by rule for x n(integral and fractional n), sums, products, quotients, chain rule, trigonometric, logarithmic and exponential functions of a single variable. Parametric differentiation. Applications: equations of tangent and normal, kinematics, rates of change and stationary points.Integration: anti-derivatives and their applications to areas and volumes</td></tr>
<tr><td><b>Mode of delivery</b></td><td colspan="2">Lectures , directed reading, practical demonstrations of communication services and projects.</td></tr>
<tr><td><b>Instructional Material and/or Equipment</b></td><td colspan="2">Audi visual equipment, chalkboard, computer simulation software, computer programming tools</td></tr>
<tr><td rowspan="4"><b>Course Assessment</b></td><td>Type</td><td>Weighting (%)</td></tr>
<tr><td>Examination</td><td>60</td></tr>
<tr><td>Continuous Assessment</td><td>40</td></tr>
<tr><td>Total</td><td>100</td></tr>
</table>

COURSE NAME	**LINEAR ALGEBRA**
Credit Units	3
Pre-requisite	
Purpose of the Course	To facilitate students develop abilities in abstract mathematical thinking, application problem solving and theorem proving, understanding mathematical systems, as well as mastering the specific content of linear algebra.
Expected Learning Outcomes	At the end of the course the students will be able to: 1. Resolve problems involving linear equations and equations of higher order; 2. Solve various algebraic equations; 3. Expand n by n matrix; 4. Use properties of determinants to solve equations written in determinant form; 5. Solve pairs of simultaneous equations; 6. Apply elementary algebraic techniques in problem solving.

Course Content	Basic concepts of linear algebra. Linear equations; Solutions of systems of linear equations; linear transformation; bases, projections; quadratic equations; vectors and subspaces; Matrices, matrix Multiplication, Matrix Inverse, Matrix Representation of Linear Maps, determinant, Laplace expansion, Cramer's rule subspaces, linear independence, and bases; determinants; Eigenvalues and Eigenvectors; other topics and applications.	
Mode of delivery	Lectures, directed reading, practical demonstrations of communication services and projects.	
Instructional Material and/or Equipment	Audi visual equipment, chalkboard, computer simulation software, computer programming tools	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100
Recommended reference material	1. Friedberg, Linear Algebra, 4th edition, Prentice Hall, 2003 2. Hoffman, D. and R. Kunze, Linear Algebra, Prenice-hall, Englewood Cliffs, NJ.	

COURSE NAME	**PROBABILITY AND STATISTICS**
Credit Units	3
Pre-requisite	Discrete Structures
Purpose of the Course	This course is an introduction to probability and statistics. Students will be introduced to organizing data, descriptive statistics, and inferential statistics while incorporating problem solving and critical thinking skills to real life situations
Expected Learning Outcomes	At the end of the course, students will be able to: 1. Solve problems using probability; 2. Interpret and construct statistical charts and tables; 3. Perform a wide variety of probability calculations and derivations; 4. Communicate results of statistical analyses graphically and verbally; 5. Accurately compute numerical summaries used to describe the central tendency, spread and shape in the distribution of numerical data sets; 6. Apply correlation, regression and hypothesis testing to statistical problems.
Course Content	Introduction to statistics and statistical thinking; fundamental elements of statistical analysis; introduction to the use of computers in statistical analysis; describing and exploring data: distributions of data, measures of location, measures of variation, basic elements of probability; random variables; moments and moment generating functions, linear combination of random variables; Bernouli trials and the binomial distribution, the geometric and negative binomial distributions; sampling without replacement; the Poison distribution; the normal distribution; the central limit theorem;

	confidence intervals and sample sizes. Basic concepts of inference, correlation, regression and hypothesis testing.	
Mode of delivery	Lectures, directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.	
Instructional Material and/or Equipment	Audi visual equipment, chalkboard, computer simulation software, computer programming tools	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100
Recommended reference material	1. Elementary Statistics, 7th Edition, Mario F Triola, Addison-Wesley, ISBN: 0201775700 2. Basic Business Statistics and Student CD-ROM, Berenson David M. Levine Timothy C. Krehbiel, Prentice Hall (2003), ISBN: 0131037919 3. Mathematical Statistics 5th Edition, John E Freund & Ronald E Walpole,, Prentice Hall, 0135638348. 4. Introduction to Mathematical Statistics, Hogg R. and Craig C., Prentice Hall, ISBN: 0023557222 5. Probability and Statistics Third Ed.,, De Groot,, Pearson Addison, Wesley, ISBN: 0-201524880	

COURSE NAME	**COMPUTER ARCHITECTURE AND ORGANIZATION**
Credit Units	3
Pre-requisite	Digital Electronics
Purpose of the Course	To elaborate the fundamental concepts underlying current implementations of Computer Architecture and Organization.
Expected Learning Outcomes	At the end of the course the students shall be able to: 1. Identify contributors to computer architecture and organization and relate their achievements to the knowledge area; 2. Explain the reasons and strategies for different architectures; 3. Identify some of the components of a computer; 4. Indicate some strengths and weaknesses inherent in different architectures; 5. Describe how computer engineering uses or benefits from computer architecture and organization.
Course Content	History and overview of computer architecture and organization; Fundamentals of computer architecture: Organization of the von Neumann machine ; Instruction formats ; The fetch/execute cycle; instruction decoding and execution ; Registers and register files; Instruction types and addressing modes ; Subroutine call and return mechanisms ; Programming in assembly language ; I/O techniques and interrupts; Computer arithmetic: Representation of integers: positive, negative numbers; Algorithms for common arithmetic operations:

	addition, subtraction, multiplication, division; significance of range, precision, and accuracy in computer arithmetic; Representation of real numbers: standards for floating-point arithmetic; Algorithms for carrying out common floating-point operations; Converting between integer and real numbers; Multi-precision arithmetic; Hardware and software implementation of arithmetic unit; The generation of higher order functions from square roots to transcendental functions; Memory system organization and architecture: Memory systems hierarchy; Coding, data compression, and data integrity ; Electronic, magnetic and optical technologies; Main memory organization and its characteristics and performance; Latency, cycle time, bandwidth, and interleaving; Cache memories (address mapping, line size, replacement and write-back policies); Virtual memory systems; Memory technologies such as DRAM, EPROM, and FLASH; Reliability of memory systems; error detecting and error correcting systems; Interfacing and communication: I/O fundamentals: handshaking, buffering; I/O techniques: programmed I/O, interrupt-driven I/O, DMA; Interrupt structures: vectored and prioritized, interrupt overhead, interrupts and reentrant code; Memory system design and interfacing; Buses: bus protocols, local and geographic arbitration; Device subsystems: External storage systems; organization and structure of disk drives and optical memory; Basic I/O controllers such as a keyboard and a mouse; RAID architectures; Video control; I/O Performance; SMART technology and fault detection; Processor to network interfaces.
Mode of delivery	Lectures , directed reading, practical demonstrations of typical computing systems.
Instructional Material and/or Equipment	Audi visual equipment, chalkboard, computer simulation software

Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100

Core Reading Material for the course	TITLE	AUTHOR	PUBLISHER	ISBN
	Computer Architecture: A quantitative approach	John L. Henessy, David A Patterson	Morgan Kaufmann Publishers	978-1558603295

COURSE NAME	**ASSEMBLY LANGUAGE PROGRAMMING**
Credit Units	3
Pre-requisite	
Purpose of the Course	To enhance students understanding in PC hardware, using assembly language and writing, linking and executing a program written in assembly language.

<table>
<tr><td><b>Expected Learning Outcomes</b></td><td colspan="2">At the end of the course the students will be able to:
1. Discuss concepts of computer organization and instruction and data representation;
2. Describe the basics of assembly language style programming;
3. Analyze a particular machine assembly language.</td></tr>
<tr><td><b>Course Content</b></td><td colspan="2">Basic Computer Organization, Number Systems, Data Codes and Internal Representation
Introduction to simple machine and Assembly Language instructions, DEBUG program, Interrupts, Program Logic and Control (selection and iteration), Arithmetic Operations, Logical and Bit Operations, Addressing Modes, Subroutines and the Hardware Stack, Macro Definitions; High-Level Language Interface, String Processing</td></tr>
<tr><td><b>Mode of delivery</b></td><td colspan="2">Lectures , directed reading, and hands-on laboratory sessions and projects.</td></tr>
<tr><td><b>Instructional Material and/or Equipment</b></td><td colspan="2">Audi visual equipment, chalkboard, cross-assemblers, computers</td></tr>
<tr><td rowspan="4"><b>Course Assessment</b></td><td>Type</td><td>Weighting (%)</td></tr>
<tr><td>Examination</td><td>60</td></tr>
<tr><td>Continuous Assessment</td><td>40</td></tr>
<tr><td>Total</td><td>100</td></tr>
</table>

COURSE NAME	**DIGITAL ELECTRONICS I**
Credit Units	3
Pre-requisite	Electronics
Purpose of the Course	To introduce fundamental concepts underlying current developments in mobile applications and wireless networks.
Expected Learning Outcomes	At the end of the course the students will be able to: 1. Evaluate communication standards used in today mobile computing; 2. Develop applications for a mobile device; 3. Discuss topics of caching, data replication, and synchronization in mobile computing.

Course Content	History and overview: digital logic ; logic circuits, switching, memory, registers, and digital systems; Boolean logic; sequential logic ; gates, circuits, combinational circuits, and modules ; special forms of memory module forms registers ; Switching theory: Number systems and codes ; Binary arithmetic ; Boolean and switching algebra; Representation and manipulation of switching functions ; Minimization of switching functions ; Incompletely specified switching functions; Combinational logic circuits: Basic logic gates (AND,OR,NOT,NAND,NOR,XOR) ;Realization of switching functions with networks of logic gates; 2-level networks: AND-OR,OR-AND,NAND-NAND,NOR-NOR ;Multi-level networks; Physical properties of logic gates (technology, fan-in, fan-out, propagation delay); Elimination of timing hazards/glitches; Memory elements: Unclocked and clocked memory devices (latches, flip flops) ;Level vs. edge-sensitive, and master-slave devices ;Basic flip flops (SR, D, JK, T); Asynchronous flip flop inputs (preset, clear) ;Timing constraints (setup time, hold time) and propagation delays ; Data registers (selection, clocking, timing) ;Random-access memory (RAM).	
Mode of delivery	Lectures , directed reading, laboratory practicals	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100

COURSE NAME	**DIGITAL ELECTRONICS II**
Credit Units	3
Pre-requisite	Electronics Digital Electronics I
Purpose of the Course	To advance students understanding of the concepts underlying developments in mobile applications and wireless networks.
Expected Learning Outcomes	At the end of the course the students will be able to: 1. Design combinational circuits; 2. Draw and interpret state diagrams and tables; 3. Design and analyse synchronous sequential circuits; 4. Create and interpret data and shift registers.
Course Content	Modular design of combinational circuits: Design of medium scale combinational logic modules ;Multiplexers, demultiplexers, decoders, encoders, comparators ;Arithmetic functions (adders, subtracters, carry lookahead) ;Multipliers, dividers ;Arithmetic and logic units (ALUs); Hierarchical design of combinational circuits using logic modules ; Sequential logic circuits: Finite state machines (FSMs), clocked and unclocked ;Mealy vs. Moore models of FSMs ; Modeling FSM behavior: State diagrams and state tables, timing diagrams, algorithmic state machine charts; Analysis of synchronous and asynchronous circuits; Design of synchronous sequential circuits: State minimization, state assignment,

	next state and output equation realization ; Sequential functional units: Data registers, shift registers, counters, sequence detectors, synchronizers, debouncers, controllers .	
Mode of delivery	Lectures , directed reading, and hands-on laboratory sessions and projects.	
Instructional Material and/or Equipment	Audio visual equipment, chalkboard	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100

COURSE NAME	**PRINCIPLES OF OPERATING SYSTEMS**
Credit Units	3
Pre-requisite	Fundamentals of computing
Purpose of the Course	To enhance students' understanding of important concepts and algorithms in operating systems
Expected Learning Outcomes	At the end of the course the students will be able to: 1. Construct computer softwares using different operating system environments and systems tools; 2. Discuss the concepts of modern operating systems; 3. Apply knowledge of operating systems to system performance tuning and system administration; 4. Enforce suitable synchronization in designing multithreaded applications.
Course Content	OS architecture: kernel, libraries, drivers, processes, daemons/services, GUI; Fundamental issues in design: processes, threads and memory management, process and thread co-ordination and synchronization (mutexes, barriers...), inter-process communication (signals, pipes, messages, events, etc.), real-time clock management. Multithread programming: Concurrent processing, mutual exclusion, deadlocks. I/O device drivers, file systems, and frame stage network communication. Features of UNIX or LINUX. Principles of distributed operating systems including networking protocols, distributed file systems, remote IPC mechanisms, graphical user interfaces, load balancing and process migration. Lab session will focus on Threads, Signals, Pipes, Inter-process communication; programming tools.
Mode of delivery	Lectures , directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.
Instructional Material and/or Equipment	Audio visual equipment, chalkboard, computer simulation software, computer programming tools

Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100
Core and reference material	1. Operating System Concepts, Fifth Edition (1997), A. Silberschatz and P. Galvin, (Addison-Wesley), ISBN: 0130313580 2. Modern Operating Systems, Andrew S Tannenbaum, Prentice Hall, ISBN: 0-13-031358-0 3. Operating Systems, William Stallings, Prentice-Hall 2001, 4. Operating Systems: A modern perspective, Gary Nutt, Addison Wesley, (2000)	

COURSE NAME	**INTRODUCTION TO DATABASE SYSTEMS**	
Credit Units	3	
Pre-requisite	Software Engineering	
Purpose of the Course	To provide students with an in-depth understanding of the design and implementation of database systems.	
Expected Learning Outcomes	At the end of the course the students will be able to: 1. Develop a database from logical to physical design; 2. Map a physical design to a database management system; 3. Apply SQL to databases.	
Course Content	Introduction to database systems. File systems and databases. Definitions and terminology. Database models + Modelling techniques. Entity Relational Modelling, Normalization techniques; Normalization anomalies. The relational system; Relational Calculus; Relational Algebra. Enterprise rules and constraints; Joins and relational algebra operations. Introduction to Transaction management and concurrency control.	
Mode of delivery	Lectures , directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.	
Instructional Material and/or Equipment	Audi visual equipment, chalkboard, computer simulation software, computer programming tools	
Course Assessment	Type	Weighting (%)
	Examination	50
	Practical lab-Continuous Assessment	50
	Total	100

Recommended reference material	1. Fundamentals of Database Systems. Second Edition, Ramez Elmasri, Shamkant B. Navathe, Addison-Wesley Pub Co, ISBN: 0805317481 2. Database Systems: A Practical Approach to Design, Implementation, and Management, Thomas M. Connolly, Carolyn E. Begg, Addison-Wesley Pub Co, ISBN: 0201342871 3. Oracle Database Environment and other resources: 4. http://www.wbluhm.com/MySQLTut.html 5. http://osiris.sund.ac.uk/~cs0dne/Teaching/com348.html 6. http://www.booksites.net/download/connbegg/download.htm

COURSE NAME	**INTERNET TECHNOLOGIES**
Credit Units	3
Pre requisite	None
Purpose of the Course	Introduce students to Internet technologies and have them be able to design and develop static web content and client-side programming using appropriate tools.
Expected Learning Outcomes	At the end of this course, the student will be able to:- 1. Analyze the different technologies that are used in the Internet and services it provides; 2. Criticize issues related to Internet governance and the roles of the different Internet institutions; 3. Create and publish static and multimedia content including graphics, sounds, media, and animations; 4. Create and publish a simple Web site including client-side behaviours.
Course Content	Introduction to the Internet: Services provided. Access Types, Intranets and Extranets. Brief History of the Internet, Social Impact, Internet Today in the World, in Africa and in Kenya; Internet Technologies: Packet Switching, Internet Protocols (TCP/IP, IPv4, IPv6); The World Wide Web: What is the World Wide Web? History of the World Wide Web, Hypertext Mark-up Language (HTML), Extensible Hypertext Mark-up Language (XHTML), Web Browsers, Hypertext Transfer Protocol (HTTP), Web Applications, Application Service Providers. Domain Name System (DNS): Introduction, Domain Name System (DNS). DNS Resolution, Recursion, Caching and Time to Live (TTL), Domain Names; Top-level Domains (TLD), Generic Top-level Domains (gTLD), Country Code TLD (ccTLD), ccTLD .ke; Second-Level Domain (SLD) and Country Code Second-level Domain (ccSLD); Other-level Domains and Sub-domains. Current Status of Domains, Uniform Resource Locator (URL) and Uniform Resource Identifier (URI), Domain Name History, Official Assignment of Domain Names, Root Nameservers, Internet Governance; Internet Related Organisations: Internet Corporation for Assigned Names and Numbers (ICANN), Internet Assigned Numbers Authority (IANA), Regional Internet Registry (RIR), Local Internet

	Registry (LRI), Number Resource Organization (NRO), Delegation and Management of ccTLD, International Telecommunications Union (ITU), Internet Engineering Task Force (IETF), World Wide Web Consortium (W3C). Domain Name Registries: Operation, Registrants - Legal Users of Domains, Domain Name Registrars, Designated Registrar, Domain Name Transfers, Introduction to the Hypertext Mark-up Language (HTML), Introduction, Basic Tags, Tables, Links and Anchors, Objects, Images and Applets, Forms, Style Sheets (CSS), Frames, Scripts and Events; Client-side programming: Scripting examples in languages (e.g. JavaScript, VBScript). The Web Browser Environment, Events and Event Handling, Forms and Form Elements, Windows and Frames Object Model, the Document Object Model, Functions, Objects, Arrays, Pattern Matching With Regular Expressions; Web Authoring tools (e.g. Dreamweaver), Multimedia: graphics, animations (Flash content);Introduction to java applets.			
Mode of delivery	Lectures, Tutorials, directed reading and hands-on laboratory sessions and projects.			
Instructional Material and/or Equipment	Audio-visual equipment, chalkboard, PCs and Web server access. Web authoring tools such as Dreamweaver, Flash, etc.			
Course Assessment	Type		Weighting (%)	
	Examination		60	
	Coursework		40	
	Total		100	
AUTHOR	TITLE	AUTHOR	PUBLISHER	ISBN
	1. The Internet Complete Reference, Hann, H. Osborne , McGraw-Hill, 2. ISBN: 0-07-882138-X 3. The World Wide Web Complete Reference, Hann, H. Osborne , McGraw-Hill, 4. ISBN: 0-07-882142-8 5. Web Design: The Complete Reference, Hann, H. Osborne , McGraw-Hill, 6. ISBN: 0-07-212297-8 7. 10 Minute Guide to HTML 3.2, Evans, T. Que,(Indianapolis, Indiana.), 8. ISBN: 789709651 9. JavaScript: the definitive guide, Flanagan, D. O'Reilly Media Inc 10. ISBN: 1-56592-392-8 11. JavaScript and DHTML Cookbook, Goodman, D., O'Reilly Media Inc, 12. ISBN: 0-596-00467-2 13. VBScript: Masters Handbook, Goddard C.J., and White Mark, Prima 14. ISBN: 0761507698 15. Learning VBScript, Lomax, P., O'Reilly and Associates			

	16. ISBN: 1-56592-247-6 17. ActiveX and VBScript, Lomax, P., Sams.net 1. ISBN: 1-57521-207-2
Recommended reference material	Dreamweaver 8 Bible, Lowery, J. Wiley, ISBN: 0471763128

COURSE NAME	**APPLICATION DEVELOPMENT FOR THE INTERNET**	
Credit Units	3	
Prė requisite	1. Data Communications 2. Introduction to Internet Technologies 3. Database Systems	
Purpose of the Course	To introduce students to programming dynamic web sites using PHP and MySQL (or an equivalent proprietary/open source environment).	
Expected Learning Outcomes	At the end of the course the students will be able to: 1. Demonstrate understanding in PHP, ASP and MySQL programming and integration or any equivalent; 2. Create dynamic web sites and on-line transactions using PHP, ASP and MySql techniques; 3. Integrate the system with different programming languages and scripting. Demonstrate understanding of PHP and MySql programming and integration or equivalent.	
Course Content	Using PHP and MySQL: storing and retrieving data, arrays, string manipulations and regular expressions, designing web database and advanced MySQL topics and queries. E-commerce and security; Advanced PHP Techniques; Building Practical Project using PHP and MySQL. Session management, cookies, server-side + Client-side scripting, Client-server architecture and content management systems. Client/server protocols (the HTTP protocol). Web site design (using HTML and CSS) Server-side scripting (using Perl/CGI and PHP). Client-side scripting (using JavaScript). Mixed-mode scripting (using AJAX).	
Mode of delivery	Lectures, Tutorials, directed reading and hands-on laboratory sessions and projects.	
Instructional Material and/ or Equipment	Audio-visual equipment, chalkboard, PCs and access to a Web server and a database server.	
Course Assessment	Type	Weighting (%)
	Final Project	60
	Coursework	40
	Total	100

Core Reading Material for the course	TITLE	AUTHOR	PUBLISHER	ISBN
	1. PHP and MySQL Web Development, Welling and Thompson, Sams, 2. ISBN: 0672317842, 978-0672317842 3. Beginning PHP and MySQL: From Novice to Professional, Jason Gilmore, Apress (2008), ISBN: 1590598628, 978-1590598627 4. PHP 6 and MySQL 5 for Dynamic Web Sites: Visual QuickPro Guide, 2007), 5. ISBN: 978-0321525994			
Recommended reference material	1. http://www.w3schools.com/PHP/DEfaULT.asP 2. http://www.php-mysql-tutorial.com/ 3. http://www.hudzilla.org/phpbook/			

COURSE NAME	**DATA COMMUNICATIONS**	
Credit Units	3	
Pre-requisite	Fundamentals of Computing	
Purpose of the Course	The purpose of this course is to explore theoretical and technical issues of communicating data between computers over networks and to examine the technology underlying the Internet.	
Expected Learning Outcomes	At the end of the course the student should be able to: 1. Explain the fundamentals of data communications and describe the different media available to support data communications; 2. Identify key limitations in networking technology implementations; 3. Describe the role of the various protocols in facilitating the transfer of data across a communication network; 4. Discuss the role of the OSI seven layer model which attempts to standardise communication protocols; 5. Describe the techniques by which interfaces to computers are designed and implemented.	
Course Content	Data signals (timing, codes); simple data communications; simplex, duplex, semi-duplex, telephone lines, modems, multiplexers and concentrators, circuit switching, message switching; packet switching; standards and protocols:-OSI model. Basic concepts: band width, protocols, architectures, types of network – LAN, MAN, WAN, Intranet, extranet, Internet, peripheral and data communication equipment; Data routing: Serial vs. Ethernet, manual routing vs. auto synchronization.	
Mode of delivery	Lectures , directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.	
Instructional Material and/or Equipment	Audi visual equipment, chalkboard, computer simulation software, computer programming tools.	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100

Recommended reference material	1. Forouzan B. A. (2005) Data Communications and Networking. 4th Ed. New Delhi: Tata McGraw-Hill Publishing Company Limited, ISBN 0-07-063414-9 2. Stallings W. (2004) Data and computer communications. 7th Ed. New Delhi: Pearson Education, Inc. ISBN 81-7758-581-9 3. Simmonds A.J. (1997) Data communications and transmission Principles - An Introduction. New York : Palgrave Macmillan, ISBN 0-333-64689-4 4. DeNoia L. A. (1987) Data communication: Fundamentals and applications. New York :Merrill Publishing Co, ISBN 0-675-20368-6.

COURSE NAME	**COMPUTER NETWORKS**
Credit Units	3
Pre-requisite	Data Communications
Purpose of the Course	The purpose of this course is to provide a theoretical background and practical understanding of the techniques and technologies that are used to build data communications networks and the applications which use them.
Expected Learning Outcomes	At the end of the course, students will be able to: 1. Define the common computer networking terminologies; 2. Design, install, and configure a home computer network; 3. Describe the differences between networks based on telephony and Internet standards including topology, services, philosophy, and the standards process; 4. Explain how one can integrate different types of networks; 5. Explain why packet loss occurs in the Internet and why packet loss reduces network cost and improves latency; 6. Demonstrate and explain the purpose for each of the following technologies: ISDN, DSL, ATM, SONET, WAP, VOIP and DNS.
Course Content	Types of computer networks, peer-to-peer and server-based networking, public and private networks, value added networks, error detection and correction: Topologies, OSI reference model, ISDN, Multimedia, emerging services, standard components. Telephone systems, key systems, switches, bridges, routers configurations, facilities in exchanges, connecting to LAN/PBX/ACD, computer telephone integration, cables – fiber, Copper, connections, Switches – configuration, ISDN, DSL technologies, ATM, SONET, WAP, VOIP, Mobile services. Internet: Introduction, web functions, privacy and security, Internet addressing and Domain Name Service(DNS); ICMP, IPV4 and IPV6;. Network Management: functions, operating system linkages, managed switches, remote configuration and management. Performance studies: performance indices, simulation
Mode of delivery	Lectures, directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.

Instructional Material and/or Equipment	Audi visual equipment, chalkboard, computer simulation software, computer programming tools	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100
Recommended reference material	1. Forouzan B. A. (2005) *Data Communications and Networking.* 4th Ed. New Delhi: Tata McGraw-Hill Publishing Company Limited, ISBN 0-07-063414-9 2. Comer D. E. & Stevens D. L. (2001) Internetworking with TCP/IP Client-Server Programming and Applications Linux/POSIX Sockets Version. New Jersey : Prentice Hall PTR, ISBN 0130320714 3. Tanenbaum A. S. (2003) *Computer Networks.* 4th Ed. New Jersey : Prentice Hall PTR, ISBN 0130661023 4. Microsoft Corporation (1997) *Networking Essentials.* 2nd Ed. Washington : Microsoft Press, ISBN 1-57231-527-X	

COURSE NAME **INFORMATION SYSTEMS SECURITY AND DESIGN**

Credit Units 3

Pre requisite

Purpose of the Course This course introduces students to the various potential threats and sources of attacks to network security and a solid understanding of the security mechanisms to maintain security on computer systems and networks.

Expected Learning Outcomes By the end of the course the student should be able to:

1. Discuss various issues related to computer security and network attacks.
2. Assess the risks on a computer system;
3. Develop suitable security policies for an organisation;
4. Choose appropriate security mechanisms to ensure data protection.
5. Assess the capabilities and limitations of current security techniques and protocols, identification and authentication, access controls, security models;
6. Elaborate on ethical and legal issues in information systems security.

Introduction to Computer Systems and Network Security. Security and the human aspect; Fundamentals of information security principles and concepts; Identification and authentication: Password Protection. Password Protection and Management; Malicious Software: Viruses and Other Malicious Content, Distributed Denial of Service Attacks (DDoS), Countermeasures; Access Control: access control matrix, access control lists and capabilities: Lattice Based Access Control: information flow policies, military lattice, Bell-La Padula model, Chinese Wall lattice, Role-Based Access Control, RBAC policies and management; Intrusion detection and prevention. Firewalls and Trusted Systems. Network attack and defences. Security support for popular web application frameworks and technologies.

Course Content

Overview of cyber-crime, an introduction to computer hacking. Ethical hacking. Attack techniques and how to defend against them, network attacks and defences, operating system holes; application security (web, email, databases), viruses, social engineering attacks; Risk Assessment, Security Policies. Internal Controls and IS Audit; Business continuity and disaster recovery planning; IT Governance: Corporate Governance. Board and Executive Management. IS Strategy. Policies and Procedures. Risk Management. IS Management Practices; Legal and Ethical issues.

Mode of delivery

Lectures, Tutorials, directed reading and hands-on laboratory sessions and projects.

Instructional Material and/or Equipment

Audio-visual equipment, chalkboard, PCs and networking equipment for lab simulations. Security software tools.

Course Assessment

Type	Weighting (%)
Examination	60
Coursework	40
Total	100

Core Reading Material for the course

TITLE AUTHOR PUBLISHER ISBN

1. Computer Security: Art & Science, Matt Bishop, Addison Wesley,
2. ISBN: 0201440997
3. Hacking Exposed: Network Security Secrets & Solutions, Joel Scambray, et al, McGraw-Hill Osborne Media 2004, ISBN: 0072227427

Recommended reference material

1. Wireless Security: Models, Threats, and Solutions, Randall K. Nichols, McGraw-Hill Professional, 2001, ISBN 0071380388
2. Know Your Enemy: Learning about Security Threats, the HoneyNet Project, Addison-Wesley Professional, 2004, ISBN: 0321166469

COURSE NAME	MOBILE COMPUTING
Credit Units	3
Pre-requisite	Data Communications
Purpose of the Course	To enhance an understanding among students of the fundamental concepts underlying current developments in mobile applications and wireless networks.
Expected Learning Outcomes	At the end of the course the students will be able to: 1. Evaluate communication standards used in today mobile computing; 2. Develop applications for a mobile device; 3. Discuss issues of caching, data replication, and synchronization in mobile computing.

<table>
<tr><td><b>Course Content</b></td><td colspan="4">Overview of current hardware, software, and communication standards for mobile computing, distinction between mobile and distributed computing; Development of mobile applications, software tools, languages, and APIs; Mobile networking and standards in depth, service location, data dissemination, routing; Mobile agents, caching, and data replication and synchronization; Introduction to wireless networking. Advantages and disadvantages of wireless networking, Characteristics of radio propagation. Fading, Multipath propagation; Introduction to digital transmission. Definition of bit-rate and signalling rate. Introduction to synchronous transmission. The need for pulse shaping, synchronisation and line-coding. Calculation of bit-error probabilities when the channel is affected by the addition of Gaussian noise; Narrowband digital modulation. The need for modulation. Binary and multi-level (M-ary) amplitude-shift keying (ASK), frequency-shift keying (FSK) and phase-shift keying (PSK). Wideband modulation techniques to cope with inter-symbol interference direct sequence spread spectrum Adaptive Equalization Orthogonal frequency division multiplex; Medium Access Control (MAC). MAC protocols for digital cellular systems such as GSM. MAC protocols for wireless LANs such as IEEE802.11 and HIPERLAN I and II. The near far effect. Hidden and exposed terminals. Collision Avoidance (RTS-CTS) protocols; Protocols supporting mobility; Mobile network layer protocols such as mobile-IP, Dynamic Host Configuration Protocol (DHCP). Mobile transport layer protocols such as mobile-TCP, indirect-TCP. Wireless Application Protocol (WAP).</td></tr>
<tr><td><b>Mode of delivery</b></td><td colspan="4">Lectures, directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.</td></tr>
<tr><td><b>Instructional Material and/or Equipment</b></td><td colspan="4">Audi visual equipment, chalkboard, computer simulation software, computer programming tools.</td></tr>
<tr><td rowspan="4"><b>Course Assessment</b></td><td colspan="3">Type</td><td>Weighting (%)</td></tr>
<tr><td colspan="3">Examination</td><td>60</td></tr>
<tr><td colspan="3">Continuous Assessment</td><td>40</td></tr>
<tr><td colspan="3">Total</td><td>100</td></tr>
<tr><td rowspan="2"><b>Core Reading Material for the course</b></td><td>TITLE</td><td>AUTHOR</td><td>PUBLISHER</td><td>ISBN</td></tr>
<tr><td>Mobile communications</td><td>J. Schiller</td><td>Addison-Wesley</td><td>0-321-12381-6</td></tr>
<tr><td><b>Recommended reference material</b></td><td colspan="4">Computer Networks, A S. Tanenbaum, Prentice Hall PTR, 0130661023</td></tr>
</table>

COURSE NAME	**AUTOMATATHEORY**	
Credit Units	3	
Pre-requisite	Discrete Structures	
Purpose of the Course	To provide insight into the intrinsic nature of computational problems as well as possible solution techniques independent of programming language, programming paradigm, computer hardware, or any other implementation aspects.	
Expected Learning Outcomes	At the end of the course the students shall be able to: 1. Determine a language's location in the Chomsky hierarchy; 2. Identify the class for various programming languages; 3. Carry out conversion among equivalently powerful notations for language, including among DFAs, NFAs, and regular expressions, and between PDAs and CFGs; 4. Explain the Church-Turing thesis and its significance.	
Course Content	Deterministic finite automata (DFAs);Nondeterministic finite automata (NFAs); Equivalence of DFAs and NFAs; Regular expressions; The pumping lemma for regular expressions; Push-down automata (PDAs); Relationship of PDAs and context-free grammars; Properties of context-free grammars; Turing machines; Nondeterministic; Turing machines; Sets and languages; Chomsky hierarchy; The Church-Turing thesis.	
Mode of delivery	Lectures , directed reading.	
Instructional Material and/or Equipment	Audi visual equipment, chalkboard	
Course Assessment	Type	Weighting (%)
	Examination	60
	Continuous Assessment	40
	Total	100

COURSE NAME	**COMPILER CONSTRUCTION AND DESIGN**
Credit Units	3
Pre-requisite	1. Discrete Structures 2. Automata Theory 3. Computer Architecture
Purpose of the Course	To study fundamental concepts of programming language design, specification, implementation and translation through the very practical exercise of designing, writing, documenting and testing a compiler; to survey the state of the art in compiler theory and design.

Expected Learning Outcomes

At the end of the course the students will be able to:

1. Define the grammar and semantics of a language;
2. Design and implement finite state machines appropriate for use a lexical scanner;
3. Given the definition of an appropriate context free grammar, design either a bottom- up or top-down parser for the grammar;
4. Given the semantic definitions for an appropriate language, implement the semantic routines for a top-down or bottom up parser;
5. Perform code generation at the tuple level.

Course Content

Introduction: Programs, interpreters, and translators; Analysis-Synthesis model of translation; Examples of translators; Why study compilers?; Issues in compiler design; Programming Language Specification: Definitions of syntax, semantics and pragmatics; In-depth study of syntactic specifications; brief look at semantics; Case studies: the definition of the languages like minijava ; Lexical And Syntactic Analysis: Foundations: Set theory and formal language theory; Syntax and microsyntax; Regular expressions and context free grammars; Scanning and parsing fundamentals; Bottom-up vs. top-down parsing; Recursive descent parsing; LL(k) Parsing - Parse tables, properties of LL(1) grammars, transformations to make grammars LL(1) Shift-Reduce Paradigm; LR(k) Parsing: LR, SLR and LALR Implementations; Context free language hierarchy; Semantics And Semantic Analysis: Static vs. dynamic semantics; Names, scopes and bindings; Object lifetimes; Run-time system organization (the stack, the heap, garbage collection); Classification of semantic objects for an intermediate representation; Symbol tables and their relationship to semantic objects; Symbol table implementation: binary trees vs. hashing; Attribute Grammars; Code Generation: Back-end compiler architecture; Code generation issues (Basic blocks, Traces, Liveness Analysis, Register Allocation); Construction of executable code and libraries; Code Optimization: Overview of optimization; Data Flow Analysis; Peephole Optimizations; Constant Folding, Common Subexpression Elimination, Copy Propagation, Strength Reduction. Global Optimization: Loop optimizations; Induction Variable elimination, Optimizing procedure calls - inline and closed procedures. Machine-Dependent Optimization: Pipelining and Scheduling; Pentium and Pentium-Pro specifics.

Mode of delivery

Lectures , directed reading, and hands-on laboratory sessions and projects.

Instructional Material and/or Equipment

Audi visual equipment, chalkboard, computer simulation software, computer programming tools

Course Assessment

Type	Weighting (%)
Examination	60
Continuous Assessment	40
Total	100

Core Reading Material for the course

TITLE	AUTHOR	PUBLISHER	ISBN
1. Compilers: Principles, Techniques, and Tools	Aho, Sethi, and Ullman.	Addison-Wesley	

<table>
<tr><td><b>COURSE NAME</b></td><td colspan="4"><b>COMPUTER GRAPHICS</b></td></tr>
<tr><td><b>Credit Units</b></td><td colspan="4">3</td></tr>
<tr><td><b>Pre-requisite</b></td><td colspan="4">Principles of Operating Systems</td></tr>
<tr><td><b>Purpose of the Course</b></td><td colspan="4">To enhance understanding of fundamental concepts underlying current developments in computer graphics.</td></tr>
<tr><td><b>Expected Learning Outcomes</b></td><td colspan="4">At the end of the course the students will be able to:
1. Distinguish the capabilities of different levels of graphics software and describe the appropriateness of each;
2. Create images using a standard graphics API;
3. Use the facilities provided by a standard API to express basic transformations such as scaling, rotation, and translation;
4. Implement simple procedures that perform transformation and clipping operations on a simple 2-dimensional image;
5. Discuss the 3-dimensional coordinate system and the changes required to extend 2D transformation operations to handle transformations in 3D.</td></tr>
<tr><td><b>Course Content</b></td><td colspan="4">Hierarchy of graphics software,Using a graphics API,Simple color models (RGB, HSB, CMYK); Homogeneous coordinates; Affine transformations: scaling, rotation, translation; Viewing transformation; Clipping; Raster and vector graphics systems; Video display devices; Physical and logical input devices; Issues facing the developer of graphical systems</td></tr>
<tr><td><b>Mode of delivery</b></td><td colspan="4">Lectures , directed reading, practical demonstrations of parallel computation, and hands-on laboratory sessions and projects.</td></tr>
<tr><td><b>Instructional Material and/or Equipment</b></td><td colspan="4">Audi visual equipment, chalkboard, computer simulation software, computer programming tools.</td></tr>
<tr><td rowspan="4"><b>Course Assessment</b></td><td colspan="3">Type</td><td>Weighting (%)</td></tr>
<tr><td colspan="3">Examination</td><td>60</td></tr>
<tr><td colspan="3">Continuous Assessment</td><td>40</td></tr>
<tr><td colspan="3">Total</td><td>100</td></tr>
<tr><td rowspan="2"><b>Core Reading Material for the course</b></td><td>TITLE</td><td>AUTHOR</td><td>PUBLISHER</td><td>ISBN</td></tr>
<tr><td>Computer graphics: Principles and Practice</td><td>James D. Folley, Andries Van Dam, Stephen K. Veiner and John F. HughesBischof, C., Bucker M., and Gibbon P.</td><td>Addison Wesley</td><td>0-201-84840-6</td></tr>
<tr><td><b>Recommended reference material</b></td><td colspan="4">Real-Time Rendering. Thomas Akenaine Moller, Eric Haines and Nafty Hoffman. A.K. Peters Ltd. ISBN 978-1-56881-424-7</td></tr>
</table>

COURSE NAME	**DISTRIBUTED SYSTEMS**	
Credit Units	3	
Pre-requisite	1. Principles of Operating systems 2. Data communications 3. Computer Networks	
Purpose of the Course	To introduce the principles and concepts involved in the design of distributed systems and enable students to realize simple distributed systems.	
Expected Learning Outcomes	At the end of the course the students will be able to: 1. Discuss the recent developments in distributed systems; 2. Investigate problems of timing, naming and location; 3. Carry out inter-process communication in a distributed environment; 4. Operate distributed file systems.	
Course Content	Overview of distributed computing. Communication: remote procedure call, remote object invocation, message and stream oriented communication. Naming: name resolution, mobile entities; Synchronization: physical and logical clocks, global state, election algorithms, mutual exclusion, distributed transactions; Data replication: consistency models and protocols. Fault tolerance: reliable p2p and group communication, distributed commit recovery; Basis of CORBA; Distributed file systems. Session management, cookies, server-side and Client-side scripting, Client-server	
Mode of delivery	Lectures, directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.	
Instructional Material and/or Equipment	Audi visual equipment, chalkboard, computer simulation software, computer programming tools	
Course Assessment	Type	Weighting (%)
	Examination	60
	Practical lab-Continuous Assessment	40
	Total	100
Recommended reference material	1. Distributed Systems: Principles and Paradigms, A.S. Tanenbaum, M. Steen, Pearson/Prentice Hall, ISBN: 0-13-121786-0 2. Distributed Systems, Concepts and Design, G. Coulouris, J. Dollimore and T. Kindburg, Addison Wesley,ISBN: 0-201-61918-0 3. The Essence of Distributed Systems, J.M. Crichlow, Prentice Hall, ISBN: 0-130-15167- 4. Elements of Distributed Computing by Vijay K. Garg, Wiley & Sons, 2002	

<table>
<tr><td><b>COURSE NAME</b></td><td colspan="2"><b>SYSTEMS ANALYSIS AND DESIGN</b></td></tr>
<tr><td><b>Credit Units</b></td><td colspan="2">3</td></tr>
<tr><td><b>Pre-requisite</b></td><td colspan="2">Fundamentals of Computing</td></tr>
<tr><td><b>Purpose of the Course</b></td><td colspan="2">To expose the students the principles of system analysis and design, feasibility studies, data gathering, system design and software documentation.</td></tr>
<tr><td><b>Expected Learning Outcomes</b></td><td colspan="2">At the end of the course students should be able to:
1. Elaborate on the concept of feasibility studies and data gathering;
2. Analysis process techniques for effective software design;
3. Propose a system based on the concepts the system development lifecycle;
4. Formulate data dictionaries, process specifications and structured decisions.</td></tr>
<tr><td><b>Course Content</b></td><td colspan="2">Introduction to systems analysis and design. Introduction to general systems theory. Players in the systems game. Information systems building blocks. Information systems development: Software development life cycle; specification, analysis, design, implementation and testing. Project Management: project management tools and source code control systems (SCCS). Feasibility analysis and system proposal. Systems Analysis; Requirements discovery; deliverables; CASE tools for system analysis and design, data modeling and process modeling tools (data flow diagrams, entity relationship diagrams), traditional and prototyping approaches; Data modeling and analysis; Process modeling. Systems Design: Applications architecture and modeling; Database design; I/O design, input validation and user interface design (GUI). Systems Construction.</td></tr>
<tr><td><b>Mode of delivery</b></td><td colspan="2">Lectures , directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.</td></tr>
<tr><td><b>Instructional Material and/or Equipment</b></td><td colspan="2">Audi visual equipment, chalkboard, computer simulation software, computer programming tools.</td></tr>
<tr><td rowspan="4"><b>Course Assessment</b></td><td>Type</td><td>Weighting (%)</td></tr>
<tr><td>Examination</td><td>60</td></tr>
<tr><td>Practical lab-Continuous Assessment</td><td>40</td></tr>
<tr><td>Total</td><td>100</td></tr>
</table>

Recommended reference material

1. System Analysis and Design 6th Ed. Kenneth Kendall and Julie Kendall. Prentice Hall. ISBN. 0131454552
2. Systems Analysis and Design 2nd Edition. 2003. Donald Yeates, Tony Wakefield. ISBN 0273655361.
3. http://camden-www.rutgers.edu/dept-pages/business/kendall/sad.html
4. Systems Analysis and Design. 3rd Edition. Shelly Cashman, Rosenblatt. ISBN.0789542668.
5. http://www.scsite.com/sad3e/

COURSE NAME **SOFTWARE ENGINEERING**

Credit Units 3

Pre-requisite

1. Programming
2. Systems Analysis and Design

Purpose of the Course To introduce students to the fundamentals of the software engineering and a range of skills needed to undertake basic software engineering activities.

Expected Learning Outcomes At the end of the course the students will be able to:

1. Evaluate the overall structure of the software engineering process;
2. Criticize the theories underpinning the software engineering process;
3. Summarize the methodologies available to guide the software engineering process;
4. Apply case tools to software development life cycle.

Course Content Introduction: basic concepts, project planning and management. Software process models: waterfall model, incremental process models, evolutionary process models, specialized process models and agile process models. Software requirement analysis and specification. Software design: function-oriented design, object-oriented design, detailed design and programming methodologies. Software testing: testing approaches e.g. unit testing and integration testing; testing tactics e.g. white-box testing, black-box testing and object-oriented testing. Software evolution and CASE tools.

Mode of delivery Lectures, directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.

Instructional Material and/or Equipment Audi visual equipment, chalkboard, computer simulation software, computer programming tools

Course Assessment	Type	Weighting (%)
	Examination	60
	Practical lab-Continuous Assessment	40
	Total	100

Recommended reference material

1. Software Engineering, Sommerville, I., Addison-Wesley2004, ISBN: 0-201-56898-5
2. Software Engineering – A Practitioners Approach, Pressman, McGraw-Hill. 2005, ISBN: 0-07-050814-3
3. Fundamentals of software engineering, Ghezzi, New Delhi, India Prentice Hall of India 2003 , ISBN: 8120322428
4. Software Engineering. 2008. Ivan Marsick, http://www.caip.rutgers.edu/~marsic/books/SE/

	SOFTWARE PROJECT MANAGEMENT
Credit Units	3
Pre requisite	SOFTWARE ENGINEERING
Aim	To introduce the students to the fundamentals of IS software project management.
Learning Outcomes	At the end of the course, the students should be able to: 1. Discuss the nature, purposes, and significance of Information Technology Software Project Management; 2. Describe various techniques in managing projects and team members; 3. Make decisions on the feasibility of a chosen project proposal based on defined criteria and considerations; 4. Collect, analyze, consolidate and present data; 5. Manage change, risks, clients, teams and suppliers brought about by any IS project; 6. Explain the relationships between business strategy and information systems; 7. Discuss the importance of planning, monitoring, controlling, and reporting an IS project; 8. Write a convincing IS project estimate on time and budget.
Content	Definitions, Managing Change; Business Strategy and Information Systems. The Organizational Framework; The Profile of a Project - Justifying the Project; Project Planning – Understanding the Work; Project Planning – Estimating; Project Planning – Scheduling and Resourcing; Monitoring Progress; Exercising Control; Reporting Progress; Quality Management; Risk Management; Value Management; Selling the Project; Client Management Issues; Managing Suppliers; Leadership; Performance Management; Project Teams; The Working Environment; The Project Manager.

Learning and Teaching Methodologies	Lectures, small group discussion, case studies, individual projects and tutorials			
Assessment	**Type**		Weighting (%)	
	Examination		60	
	Continuous Assessment		40	
	Total		100	
Recommended Reading	TITLE	AUTHOR	PUBLISHER	ISBN
	1. Project Management: A Managerial Approach, 2nd ed.,	Meredith, JR. & Mantel, SJ., Jr.,	Wiley	1880410230
	2. Project Management for Information Systems	Don Yeates and James Cadle	Pitman Publishing T58.6Y43	0273620193
Support Materials and Resources	Texts, audio and video cassettes, computer software			