Course Contents for the Study Plan of the Bachelor's Degree in Computer Science

Course Title and Code	PHYS115 - General Physics II

I. Course Identification and General Information:

Course Title	General Physics II	Course Code	PHYS115	Pre-requisite	PHYS110
Department	Computer Engineering	Course Level	2	Credit Hours	3 (2+1)

II. Course Description/Topics: The following course topics will be covered.

- Electromagnetism, Coulomb's Law and Electric Field, Gauss' Law, Electric work and Potential Energy
- Capacitors, Insulators and Current and resistance
- Electric energy and Power
- Direct current and Kirchhoff's Laws
- Magnetic Field and Electric charge movement in magnetic field
- Ampere's Law and Faraday's Law
- Alternative Current, Effective values and Resonance
- Nuclear Physics: Photoelectric Effect, Atomic spectra, Bohr's atomic model, Half-life, Radiation activity.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Indicate some reasons for studying physics
- Explain the purpose and role of physics in computer engineering
- Knowledge and understanding of most fundamental physical laws and principles and competence in the application of these principles to diverse areas of physics.
- Knowledge of the primary mathematical methods for the analysis of physical.
- Identify physical principles relevant to a problem and make necessary approximations to obtain solutions.
- Apply appropriate mathematical or computational tools to a physical problem.
- Apply such knowledge and understanding to the solution of qualitative and quantitative problems of a familiar and unfamiliar nature.
- Recognize and analyze novel problems and plan strategies for their solution.
- Evaluate, interpret and collate information and data.
- Illustrate scientific ideas concisely, accurately and informatively both in writing and orally.
- Illustrate scientific information in the form of clear and accurate scientific reports.
- Demonstrate the understanding of the fundamental concepts/laws in physics by setting up laboratory equipment safely and efficiently and planning and carrying out experimental procedures
- Demonstrate the ability to apply knowledge/skills to real world settings by identifying possible sources of error and implementing techniques that enhance precision
- Observe, accurately record and analyze, including estimates of accuracy, the results of experiments into physical processes.
- Conclude from results obtained through experiment or investigation and compare these with expected outcomes or published data.
- Demonstrate critical thinking/ analytical reasoning ability through analyzing and interpreting experimental data
- Demonstrate effective oral/written communication skills/ability by reporting verbally and in written language the experimental data, results, and assessment of reliability
- Demonstrate teamwork skills/ ability to collaborate by working in groups on a laboratory experiment
- Demonstrate ability to innovate and be creative in a laboratory experiment

IV. Required Text:

- Arthur Beiser, Physics, 5th Ed., The Benjamin/Cummings Publishing Co., 2005.

Course Title and Code	CS181- Programming II

V. Course Identification and General Information

Course Title	Computer Programming II	Course Code	CS 181	Pre-requisite	CSC111
Department	Computer Science	Course Level	3	Credit Hours	3(2+1)

VI. Course Description/Topics: The following course topics will be covered.

- Problem-solving, algorithmic design, Introduction to Programming, Structured Program Development, Program Control, Functions, recursion, Arrays, pointers, structures, unions, files. Assignments in algorithm design and translation of algorithms in high level language like C language.
- Effect-free programming
- Processing structured data (e.g., trees) via functions with cases for each data variant
- First-class functions (taking, returning, and storing functions)
- Function closures (functions using variables in the enclosing lexical environment)
- Defining higher-order operations on aggregates, especially map, reduce/fold, and filter.
- **PL/Basic Types**
- A type as a set of values together with a set of operations: Primitive types (e.g., numbers, Booleans), Compound types built from other types (e.g., records, unions, arrays, lists, functions, references)
- Association of types to variables, arguments, results, and fields
- Type safety and errors caused by using values inconsistently with their intended types
- Goals and limitations of static typing: Eliminating some classes of errors without running the program, Undecidability means static analysis must conservatively approximate program behavior
- Generic types (parametric polymorphism): Definition, Use for generic libraries such as collections, Comparison with ad hoc polymorphism (overloading) and subtype polymorphism
- • Complementary benefits of static and dynamic typing
- o Errors early vs. errors late/avoided
- o Enforce invariants during code development and code maintenance vs. postpone typing decisions while prototyping and conveniently allow flexible coding patterns such as heterogeneous collections
- o Avoid misuse of code vs. allow more code reuse
- o Detect incomplete programs vs. allow incomplete programs to run

VII. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Compare and contrast (1) the procedural/functional approach—defining a function for each operation with the function body providing a case for each data variant—and (2) Understand both as defining a matrix of operations and variants.
- Write basic algorithms that avoid assigning to mutable state or considering reference equality. [Usage]
- Write useful functions that take and return other functions. [Usage]
- Correctly reason about variables and lexical scope in a program using function closures. [Usage]
- Define and use iterators and other operations on aggregates, including operations that take functions as arguments.

VIII. Required Text:

- C How to Program, ISBN-10: 0-13-612356-2, Year: 2010, Author(s): Paul Deitel, Harvey M. Deitel

Course Title and Code	MATH115 - Integral Calculus

I. Course Identification and General Information:

Course Title	Integral Calculus	Course Code	MATH115	Pre-requisite	MATH105
Department	Computer Engineering	Course Level	3	Credit Hours	3 (3+0)

II. Course Description/Topics: The following course topics will be covered.

- The indefinite integral.
- The definite integral.
- Fundamental theorem of calculus.
- Area, volume of revolution.
- Work, Arc length.
- Differentiation and integration of inverse trigonometric functions, the logarithmic, exponential, hyperbolic and inverse hyperbolic functions.
- Techniques of integration: substitution, by parts, trigonometric substitutions, partial fractions, miscellaneous substitutions.
- Numerical integration.
- Improper integrals.
- Polar coordinates.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Explain indefinite and definite integration
- Evaluate integrals using Techniques of Integration such as substitution, Trigonometry substitution, Inverse substitution, Partial fraction, Integral by parts.
- Evaluate integrals by using Fundamental Theorem of Calculus.
- Describe the area and volume of revolution
- Apply Integration to compute areas and Volumes by slicing, Volume of revolution and surface of area of revolution.
- Explain the role of integration in the study of work and give examples.
- Locate the center of gravity by applying integrals.
- Define and calculate an integral numerically to a given degree of accuracy.
- Analyze and evaluate improper integrals.
- Describe polar coordinates.

IV. Required Text:

- Calculus, Howard Anton, John Wiley & Sons, 2009

V. References:

- Calculus: Early Transcendentals, James Stewart, Thomson Brooks/Cole, 2007.

Course Title and Code	IT131 **Database**

I. Course Identification and General Information:

Course Title	Database	Course Code	IT131	Pre-requisite	CSC111
Department	Information Technology	Course Level	3	Credit Hours	3(2+1)

II. Course Description/Topics: The following course topics will be covered.

- Data, Database, and DBMS definition.
- Database system.
- Overview of database management.
- Database system architecture.
- Database Languages and models.
- Relational databases and database Algebra.
- Structured Query Language (SQL), outer join, and views.
- Database Integrity.
- Entity Relationship Model.
- Database design.
- Relational Mapping.
- Normalization.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Differentiate and use key terms such as: information, data, database, database management system, and metadata and give a brief history of database models
- Explain the advantages of a database approach compared to traditional file processing.
- Describe the features of the relational model including relations, tuples, attributes, domains and operators.
- Formulate and test SQL queries using SELECT, FROM, WHERE, ORDER BY blocks, set operators, UNION, DISTINCT, LIKE, and BETWEEN operators, GROUP BY HAVING clause, sub-queries, VIEWS, INSERT, UPDATE and DELETE options, project, union, intersection, set difference, natural join, and outer join.
- Define entity integrity and referential integrity and give examples of user defined integrity constraints. Create, design, describe, and interpret Entity Relationship diagram.
- Explain the relationship between functional dependencies and keys and give examples.
- Label 1NF, some of 2NF, 3NF, or BCNF violations given a set of relations and a set of functional dependencies.

IV. Required Text:

- Database System Concepts, 6/E edition, Henry F. Korth, McGraw-Hill, 2011.

V. References:

- Fundamentals of Database Systems, 6/E, Elmasri , Addison-Wesley, 2011.
- An Introduction to Database Systems, 8/E, Date, Addison-Wesley, 2004.

Course Title and Code	CS182 - Programming III

I. Course Identification and General Information:

Course Title	Computer Programming III	Course Code	CS 182	Pre-requisite	CS181
Department	Computer Science	Course Level	4	Credit Hours	3(2+1)

II. Course Description/Topics: The following course topics will be covered.

- Object-oriented design
- ○ Decomposition into objects carrying state and having behavior
- ○ Class-hierarchy design for modeling
- Definition of classes: fields, methods, and constructors
- Subclasses, inheritance, and method overriding
- Dynamic dispatch: definition of method-call
-
- Subtyping (cross-reference PL/Type Systems)
- ○ Subtype polymorphism; implicit upcasts in typed languages
- ○ Notion of behavioral replacement: subtypes acting like supertypes
- ○ Relationship between subtyping and inheritance
- Object-oriented idioms for encapsulation
- ○ Privacy and visibility of class members
- ○ Interfaces revealing only method signatures
- ○ Abstract base classes
- Using collection classes, iterators, and other common library components
-

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Compare and contrast (1) the procedural/functional approach—defining a function for each operation with the function body providing a case for each data variant—and (2) the object-oriented approach—defining a class for each data variant with the class definition providing a method for each operation. Understand both as defining a matrix of operations and variants.
- Use subclassing to design simple class hierarchies that allow code to be reused for distinct subclasses.
- Correctly reason about control flow in a program using dynamic dispatch. [Usage]
- Explain the relationship between object-oriented inheritance (code-sharing and overriding) and subtyping (the idea of a subtype being usable in a context that expects the supertype). [Familiarity]
- Use multiple encapsulation mechanisms, such as function closures, object-oriented interfaces, and support for abstract datatypes, in multiple programming languages. [Usage]
- Define and use iterators and other operations on aggregates, including operations that take functions as arguments, in multiple programming languages, selecting the most natural idioms for each language. [Usage]
-

IV. Required Text:

- C How to Program, 4/E edition, Deitel & Deitel. ISBN-10: 0-13-612356-2, 2010

V. References:

- An Introduction to Object Oriented Programming with Java, 5th Edition, Thomas Wu, 2009, ISBN: 978-0-07352-330-9

Course Title and Code	COE121 – Logic Design

I. Course Identification and General Information:

Course Title	Logic Design	Course Code	COE121	Pre-requisite	CSC111
Department	Computer Engineering	Course Level	4	Credit Hours	3 (3+0)

II. Course Description/Topics: The following course topics will be covered.

- Historical background and contributors of digital logic
- Number systems (binary, decimal, hexadecimal), codes, base conversions
- Boolean algebra, minimization of Boolean functions
- Basic logic gates, Physical properties of logic gates; realizations of functions
- Combinational circuits: multiplexers, decoders, encoders, comparators
- Arithmetic functions (adder, subtractor); ALUs
- Latches, basic flip flops, flip flops with reset, enable; data registers
- Sequential Logic; Design of Finite State Machines (Mealy and Moore)
- Digital System modeling using Hardware Description Languages (Verilog/VHDL)

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Identify the importance of digital logic design
- Work with various number systems and convert between bases
- Reduce Boolean functions and realize the functions using logic gates
- Analyze and explain uses of small- and medium-scale logic functions as building blocks
- Design and describe the operation of basic memory elements
- Analyze and synthesize synchronous sequential machines
- Apply digital system design principles and descriptive techniques
- Model and simulate a digital system using a hardware description language, such as VHDL or Verilog

IV. Required Text:

- Digital Design and Computer Architecture, 2/e, David Money Harris and Sarah L. Harris, Morgan Kaufmann, MA 02451, USA, 2013, ISBN: 978-0-12-394424-5

V. References:

- Digital Design, With an Introduction to the Verilog HDL, 5/e, M. Morris Mano and Michael D. Ciletti, Prentice Hall,New Jersey 07458, 2013. ISBN: 978-0-13-277420-8

Course Title and Code	COE122 – Logic Design Lab

I. Course Identification and General Information:

Course Title	Logic Design Lab	Course Code	COE122	Co-requisite	COE121
Department	Computer Engineering	Course Level	4	Credit Hours	1 (0+1)

II. Course Description/Topics: The following course topics will be covered.

- Basic logic functions (AND/NAND, OR/NOR)
- Exclusive OR/NOR gate functions
- Transfer characteristics of gates
- Flip-flops (set/reset, D type)
- Tri-state buffer output control
- Digital system modeling using Verilog

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Identify the input/output relationships of basic logic gates
- Ability to connect basic gates as building blocks to create larger circuits
- Identify the difference of transfer characteristics of real gates with the ideal gates
- Work with various types of flip-flops
- Model and simulate a digital system using a hardware description language, such as VHDL or Verilog

IV. Required Text:

- Digital Design and Computer Architecture, 2/e, David Money Harris and Sarah L. Harris, Morgan Kaufmann, MA 02451, USA, 2013, ISBN: 978-0-12-394424-5

V. References:

- Lab-Volt Digital Logic Fundamentals, Student Workbook, 4/e, Lab-Volt Systems Inc., 2003. ISBN 0-86657-210-4

Course Title and Code	MATH116 - Linear Algebra and Multivariate Calculus

I. Course Identification and General Information:

Course Title	Linear Algebra and Multivariate Calculus	Course Code	MATH116	Pre-requisite	MATH115
Department	Computer Engineering	Course Level	4	Credit Hours	3 (3+0)

II. Course Description/Topics: The following course topics will be covered.

- Systems of linear equations, matrices, determinants, inverse of a matrix. Cramer's rule
- Vectors in two and three dimensions, scalar and vector products,
- Equations of lines and planes in space,
- Cylindrical and spherical coordinates
- Vector values functions, their limits continuity, derivatives and integrals
- Functions in two or three variables, partial derivatives ,differentials, chain rule,
- Directional derivatives, tangent planes and normal lines to the surfaces,
- Extrema of functions of several variables. Lagrange multipliers.
- Multiple integration: double and triple integrals; applications
- Vector fields; gradient, divergence, curl, and the del operator

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Solve linear system of equations.
- Perform operations on matrices.
- Evaluate the inverse of a matrix.
- Describe the concept of vector, how to add vectors, and multiply them with a scalar.
- Perform the dot and cross product of vectors.
- Use the properties for the dot and cross product.
- Calculate angle between vectors.
- Explain the parametric form of a line and the equation of a plane, and by using these concepts be able to solve simple geometrical problems concerning planes and lines as e.g. the distance between lines, the intersection point between a line and a plane, the distance between a point and a plane, the projection of vectors on lines and planes.
- Demonstrate the basic theory of calculus of functions in several real variables;
- Evaluate partial derivatives and multiple integrals;(e.g. area and volume)
- Apply the knowledge to solve some practical problems, such as constrained optimization problems and other problems involving differentiation and integration of multivariable functions.

IV. Required Text:

- Calculus, by Howard Anton, © 2007, ISBN 0-471-15306-0, John Wiley Publishing.

V. References:

- Vector Calculus, Third Edition, by Susan Jane Colley, 2006, Pearson Prentice Hall.

Course Title and Code	STAT126 - Probability and Statistics

I. Course Identification and General Information:

Course Title	Probability and Statistics	Course Code	STAT126	Pre-requisite	STAT100
Department	Computer Engineering	Course Level	4	Credit Hours	3 (3+0)

II. Course Description/Topics: The following course topics will be covered.

- Introduction on Thinking for Decision-making.
- Organizing and graphing Data.
- Numerical Descriptive measures.
- Hypothesis tests about the mean and proportion.
- Applications of the Chi-square statistic.
- Regression modeling and analysis.
- Index numbers with applications.
- Probability, discrete probability distribution, and continuous probability densities.
- Combinatorics.
- Conditional probability, important distributions, and densities.
- Expected value and variance.
- Sums of Random variables.
- Law of large number, central limit theorem.
- Generating functions, Markov chains, random walk.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Describe the basic concepts of statistics and probability.
- Create and interpret data using various methods of displaying circle graphs, histograms, and frequency curves, and make predictions about outliers.
- Determine possible outcomes using tree diagrams and the counting principles of permutations and combinations.
- Express the chances of events occurring in terms of either a probability or odds.
- Interpret and calculate measures of central tendency (mean, median, and mode) from data presented in a variety of forms such as charts, tables, and graphs or from data created through experimentation.
- Interpret and calculate measures of dispersions (range and standard deviation) from data presented in a variety of forms such as charts, tables and graphs or from data created through experimentation.
- Describe individual performances in terms of percentiles,
- Test the validity of a hypothesis using appropriate statistical concepts.
- Calculate the Chi-Square values for a given population.
- Perform a t-test for a designated set of data, and use the results to test the validity of a hypothesis.
- Perform a regression analysis on a set of data, either given or created through experimentation, and use the results to predict specific values of a variable. Identify the regression equation.
- Calculate the index number.
- Identify the concepts and application of central limit theorem, Markov chains and Random Walks

IV. Required Text:

- Introduction to Probability and Statistics, by Grinstead and Snell, © 2006 | ISBN-10: 0821807498 | ISBN-13: 9780821807491 ,American mathematical society

V. References:

- Introduction to Statistics, 7th edition by Prem S. Mann, © 2011 John wiley and sons ,ISBN:978-0-470-50583-0

Course Title and Code	CS211 - Concepts of Algorithms

I. Course Identification and General Information:

Course Title	Concepts of Algorithms	Course Code	CS211	Pre-requisite	CS181
Department	Computer Science	Course Level	5	Credit Hours	3(3+0)

II. Course Description/Topics: The following course topics will be covered.

- The Role of algorithms in computing - Algorithms as a technology - Analyzing and designing algorithms
- Growth of functions - Asymptotic notation - Standard notations and common functions
- Recurrences - The substitution method - The recursion-tree method - Proof of the master theorem
- Probabilistic analysis and randomized algorithms - The hiring problem - Indicator random variables - Randomized algorithms - Probabilistic analysis and further uses of indicator random variables
- Sorting: Bubble sort - Heap sort –- Quick sort – Sorting in linear time
- Medians and Order Statistics - Minimum and maximum - Selection in expected linear time - Selection in worst-case linear time
- Hash tables – Direct-address tables - Hash functions - Open addressing - Perfect hashing
- Binary search trees – Querying a binary search tree Insertion and deletion – Randomly-built binary search trees
- Red-Black trees - Properties of red-black trees - Rotations - Insertion – Deletion
- Augmenting data structures - Dynamic order statistics - How to augment a data structure - Interval trees

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

By the end of the course, one should be able to:

- Develop an ability to write a pseudo code for a given problem.
- Have facility in mapping a pseudo code into implementation of examples of algorithmic strategies from scratch, and applying them to specific problems.
- Implement basic numerical algorithms.
- Determine informally the time and space complexity of simple algorithms.
- Solve elementary recurrence relations, *e.g.,* using some form of a Master Theorem.
- Use a heuristic approach to solve an appropriate problem.
- Describe the trade-offs between brute force and other strategies.
- Implement simple search algorithms and explain the differences in their time complexities.
- Be able to implement common quadratic and O(N log N) sorting algorithms.
- Be able to write efficient programs through the use of concise and efficient algorithms.
- Tackle any programming problem by breaking it into its component parts.
- Implement any algorithm using any programming language of one's choice.

IV. Required Text

- Cormen, T., C. Leiserson, R. Rivest and Clifford Stein, "*Introduction to Algorithms*", 2nd Edition MIT Press, 2002

V. Reference

- Goodrich, M.T. and R. Tamassia, "*Algorithms Design, Foundations, Analysis and Internet Examples*", John Wiley & Sons, 2002

Course Title and Code	Math212 – Discrete Mathematics

I. Course Identification and General Information:

Course Title	Discrete Mathematics	Course Code	MATH212	Pre-requisite	MATH116
Department	Computer Science	Course Level	5	Credit Hours	3(3+0)

II. Course Description/Topics: The following course topics will be covered.

- Sets, Relations, and Functions
- Basic Logic
- Proof Techniques
- Basics of Counting
- Graphs and Trees
- Discrete Probability

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Apply counting arguments, including sum and product rules, inclusion-exclusion principle and arithmetic/geometric progressions.
- Apply the pigeonhole principle in the context of a formal proof.
- Compute permutations and combinations of a set, and interpret the meaning in the context of the particular application.
- Map real-world applications to appropriate counting formalisms, such as determining the number of ways to arrange people around a table, subject to constraints on the seating arrangement, or the number of ways to determine certain hands in cards (e.g., a full house).
- Solve a variety of basic recurrence relations.
- Analyze a problem to determine underlying recurrence relations.
- Perform computations involving modular arithmetic.

IV. Required Text:

- Discrete Mathematics and its Applications, 7/e, Kenneth H. Rosen, McGraw-Hill Education, 2012, ISBN: 978-0-07-338309-5, MHID 0-07-338309-0

V. References:

- Discrete Mathematics and its Applications: with combinatorics and graph theory, Kenneth H. Rosen, McGraw-Hill Education - Europe; 7th edition (2011), ISBN13: 978-0070681880

Course Title and Code	CS213 Programming Languages Concepts

I. Course Identification and General Information:

Course Title	Programming Languages Concepts	Course Code	CS213	Pre-requisite	CS182
Department	Computer Science	Course Level	5	Credit Hours	3(3+0)

II. Course Description/Topics: The following course topics will be covered.

- Introduction to Programming Languages Concepts.
- Evolution of the Major Programming Languages.
- Describing Syntax and Semantics.
- Imperative Languages.
- Functional Programming Languages.
- Logic Programming Languages
- Names, Bindings, Type Checking, and Scopes
- OOP Languages.
- Data Types.
- Expressions and Assignment Statements.
- Statement-Level Control Structures
- Subprograms and Implementing Subprograms.
- Abstract Data Types.
- Concurrency.
- Exception Handling and Event Handling.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Recognize how a language's underlying computation model can impact how one writes programs in that language.
- Quickly learn new programming languages and how to apply them to effectively solve programming problems.
- Understand how programming language features are implemented.
- Reason about the tradeoffs among different languages and use a variety of programming languages with some proficiency.
- Introduction to Criteria and categories of programming languages.
- History of programming languages.
- Syntax and semantics.
- Logic programming languages (Prolog).
- Functional programming Languages (Lisp)
- Names, Data types and binding.
- Control statements.

IV. Required Text:

- Robert W.Sebesta., Concepts of programming languages(6th edition), 2008,Addison-Wesley, ISBN: 0-321-19362-8.

V. References:

- John Mitchell, Concepts in Programming Languages", 2002, ISBN: 9780521780988.
- Multiple Web sites dedicated to Concepts of Programming Languages.

Course Title and Code	COE223 – Computer Organization

I. Course Identification and General Information:

Course Title	Computer Organization	Course Code	COE223	Pre-requisite	COE121
Department	Computer Engineering	Course Level	5	Credit Hours	3 (3+0)

II. Course Description/Topics: The following course topics will be covered.

- History and overview.
- Fundamentals of computer architecture.
- Computer arithmetic (adders, subtractors, comparators, multiplication, division, ALU's).
- Number systems (fixed & floating point).
- Sequential Building Blocks (counter, shift register); Memory Arrays (RAMs, ROMs); Logic Arrays (PLAs, FPGAs).
- MIPS Instruction Set and Registers; Branches & Procedure Calls, Addressing Modes; Linking & Launching Applications.
- Single-Cycle Processor Data-path; Single-Cycle Processor Control.
- The CPU interface: clock, control, data and address buses.
- Address decoding and memory interfacing
- Main memory organization and its characteristics and performance.
- Cache memories (address mapping, line size, replacement and write-back policies).
- Virtual memory systems.
- Memory-mapped I/O; Memory system performance & hierarchy: Caches; Memory system optimization; Virtual Memory.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Identify some of the components of a computer.
- Describe how computer engineering uses or benefits from computer architecture.
- Explain how a computer fetches from memory and executes an instruction.
- Explain the relationship between the representation of machine level operation at the binary level and their representation by a symbolic assembler.
- Explain why a designer adopted a given different instruction formats, such as the number of addresses per instruction and variable length vs. fixed length formats.
- Write small programs and fragments of assembly language code to demonstrate an understanding of machine level operations.
- Implement some fundamental high-level programming constructs at the machine-language level.
- Appreciate the effect of a processor's arithmetic unit on its overall performance.
- Identify the main types of memory technology.
- Understand how a CPU chip becomes a complete system.

IV. Required Text:

- Harris, D. and Harris, S., "Digital Design and Computer Architecture", (2nd Edition), Morgan Kaufmann, August, 2012, ISBN: 0123944244.

V. References:

- M. Morris Mano, "Computer System Architecture", (3rd Edition).

Course Title and Code	CS221- Assembly Language

I. Course Identification and General Information:

Course Title	Assembly Language	Course Code	CS221	Pre-requisite	COE223
Department	Computer Science	Course Level	6	Credit Hours	3(2+1)

II. Course Description/Topics: The following course topics will be covered.

- Basic Features of PC Hardware.
- Instruction Addressing and Execution.
- Examining Computer Memory and Executing Instruction.
- Requirement for Coding in Assembly Language.
- Assembling Linking and Executing Programs.
- Symbolic instruction and addressing.
- Program Logic and Control.Processing String Data.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course

Understanding the following Subjects:

- Basic Features of PC Hardware.
- Instruction Addressing and Execution.
- Examining Computer Memory and Executing Instruction.
- Requirement for Coding in Assembly Language.
- Assembling Linking and Executing Programs.
- Symbolic instruction and addressing.
- Program Logic and Control.
- Processing String Data.

IV. Required Text:

- Text book: "IBM PC Assembly Language and Programming, Fifth Edition, Peter Abel.

V. References:

- Text book: Intel microprocessor, Fifth Edition, Barry B. Bray.

Course Title and Code	CS222-Operating Systems

I. Course Identification and General Information:

Course Title	Operating Systems	Course Code	CS222	Pre-requisite	CS182
Department	Computer Science, IT. and COE.	Course Level	6	Credit Hours	3(2+1)

II. Course Description/Topics: The following course topics will be covered.

- Overview of operating systems principles and concepts, components, structure.
- Operating systems services, system calls, APIs, interrupts, system mode and user mode, system programs, security, protection, OS design and implementation, processes & threads.
- Process, operation on process, PCB, process synchronization (concurrency).
- CPU Scheduling: Basic concepts, scheduling criteria, states diagram, dispatching, context switching, CPU performance, resource allocation, and schedulers.
- Scheduling algorithms, preemptive and non-preemptive scheduling, SJF, Round Robin.
- Main memory, virtual memory, algorithms, swapping, contiguous memory allocation, segmentation, paging, page replacement, allocation of frames, memory-mapped files.
- File-Systems, File Concept, naming, searching, access, backups, File Sharing, protection, directories, contents and structures.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Learn and understand the OS concepts and fundamentals, OS structure, I/O system, CPU management, process management, memory management, virtual memory, file system.
- Install, apply basic theoretical concepts using the Linux operating system.
- Study and analyze various algorithms of CPU management and memory management.
- Improving student's skills in thinking based on discussions and problem solving.
- Learn and discover the main features, the main functionalities and services of OS.
- Describe the reasons for using interrupts, dispatching, context switching and describe the difference between processes and threads.
- Study the process scheduling types such as short-term, medium-term, long-term.
- Create state and transition diagrams for simple problem domains.
- Describe relationships between scheduling algorithms and application domains.
- Understand and describe the mechanism of swapping, contiguous memory allocation, segmentation, memory paging.

IV. Required Text:

- Operating System Concepts, 9th Edition, (2012), AviSilberschatz, Peter Baer Galvin, Greg Gagne, ISBN-13: 9781118063330.

V. References:

- Operating Systems: Internals and Design Principles, William Stallings, 7th Edition, (2011), ISBN-13: 9780132309981.
- Operating Systems Design and Implementation, 3rdEdition, (2011), Kindle Edition, Andrew S. Tanenbaum, Albert S Woodhull, ISBN: o-13-142938-8.

Course Title and Code	CS214 - Data Structures

I. Course Identification and General Information:

Course Title	Data Structures	Course Code	CS214	Pre-requisite	CS 182
Department	Computer Science	Course Level	6	Credit Hours	3(2+1)

II. Course Description/Topics: The following course topics will be covered.

- Data Structures? (Linear and non-linear), built-in and abstract data structures.
- Data Structure Operations (Insertion, Deletion, Traversing/Visiting, Searching, Sorting etc.)
- Records/structs (heterogeneous aggregates)
- Algorithms: Complexity, Time-Space tradeoff. Basics of algorithm and Algorithmic Notations
- Introduction to Array, Linear Array, Representation of linear array in memory.
- Traversing, Inserting, Deleting into/from an array
- Sorting: (Bubble Sort) and Searching: (Linear and Binary Search)
- Multidimensional array and its presentation in memory. Matrices:
- Linked Lists: Presentation of Linked Lists in Memory.
- Linked List Operations: (Traversing, Searching, Insertion, Deletion)
- Circular, Header Linked and Two-way Lists.
- Stack?, Representation of stack using Array and Linked list.
- Arithmetic Expressions: Polish Notation (Infix to pre-fix-fix and post-fix notation Conversion and its evaluation using stack.), Recursion(Quick Sort, Find Factorial, Tower of Hanoi)
- Queues: Representation of queue using Array and Linked list. Circular Queue, Priority Queue.
- Introduction to Tree. Binary Trees (2-tree, complete tree), Presentation Binary Tree in Memory
- Terminologies (Parent, Child, Brothers, Siblings, Levels, Height/Depth, Path/Brach, etc.)
- Traversing Binary Tree (Pre-Order, In-Order, Post-Order), Recursion in processing of Binary Tree.
- Binary Search Tree. (Searching, inserting, and deleting in a Binary Search Tree).
- Sorting: (Insertion Sort, Bubble Sort, Selection Sort, Merging, Merge-sort, Radix Sort, Quick Sort)
- Graph theory and Terminologies, (Sequential Directed and Undirected graphs)
- Traversing a Graph. Minimum-Cost Spanning Tree.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

1. Discuss the appropriate use of built-in data structures.
2. Describe common applications for each data structure in the topic list.
3. Write programs using:(arrays, strings, linked lists, stacks, queues, Trees, and graphs).
4. Compare alternative implementations of data structures with respect to performance.
5. Compare and contrast the costs and benefits of dynamic and static data structure implementations.
6. Choose the appropriate data structure for modeling a given problem.

IV. Required Text:

- Data structures using C and C++ by YEDIDYAH LANGSAM

V. References:

- Theory and problems of data structures by Lipchuiz

Course Title and Code	CS224– Compiler Design

I. Course Identification and General Information:

Course Title	Compiler Design	Course Code	CS224	Pre-requisite	CS213
Department	Computer Science	Course Level	6	Credit Hours	3(3+0)

II. Course Description/Topics: The following course topics will be covered.

- Compiler introduction and background. Phases and cousins of compiler
- Lexical analysis: regular expressions, finite automata and its implementation.
- Syntax analysis: top-down and bottom up parsing. Derivation trees. Writing context-free grammar of a sample computer language. Recursive descent parsing. LL parsing. Non-recursive predictive parsing. SLR parsing.
- Syntax-directed translation. S-attributed and L-attributed grammars. Abstract syntax trees.
- Semantic analysis. Type checking.
- Runtime environments. Activation trees, activation records, calling sequence.
- Intermediate code generation. Three-address code. Flow-of-control statements translation.
- Code generation and optimization. Register allocation, basic block and peephole optimizations.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Identify the actual role of compiler in a language processing system.
- Distinguish syntax and parsing from semantics and implementation.
- Use formal grammars to specify the syntax of languages.
- Identify key issues in syntax: ambiguity, associativity, precedence.
- Applying data structures to various algorithms in the design of compiler phases.
- Describe semantic analysis using an attribute grammar.
- Implementing context sensitive, static analysis type checker for programming language constructs.
- Identify and fix memory leaks and dangling-pointer dereferences.
- Identify necessary steps for automatically converting code to assembly language.
- Generate the low-level code for assignment and flow of control statements.
- Analyze computer code for correctness and efficiency.

IV. Required Text:

- Compilers: Principles, techniques and tools. Aho, Lam, Sethi and Ullman.

V. References:

- Introduction to Compiler Design. T. Mogensen.

Course Title and Code	IT362 Communication skills& Ethics Issues

I. Course Identification and General Information:

Course Title	Communication skills & Ethics Issues	Course Code	IT362	Pre-requisite	
Department	Information Technology	Course Level	6	Credit Hours	2(2+0)

II. Course Description/Topics: The following course topics will be covered.

- Introduction to communication process
- The importance of communication
- Introduction to skills and goals of communication
- Types of communication
- Writing for business etiquette rules for email messages, correspondence letters, progress and activity reports
- Communication and leading at work
- Using verbal and nonverbal messages
- Listening and responding
- Overcoming listener barriers
- Adapting to differences: personality, culture and gender
- Cultural contexts powerful and powerless language
- Relating to others at work, understanding types of relationship at work
- Interviewing principles and skills. Interview types
- Collaborating in teams. Elements of team work, characteristics of effective teams.
- Enhancing team meetings, preparing for meetings

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Define communication and describe why communication is important
- Define the leadership qualities and how to be an excellent leader
- Acquaint the importance of listen effectively.
- interpret and explain their knowledge
- demonstrate interpersonal and impersonal skills
- identify and describe six approaches to leadership
- Articulate their presentations.
- interview people of high profiles
- apply their theoretical knowledge about communication skills
- Decide proper selection of words for the concerned audience.
- Adapt different cultures. like high or low context cultures
- Manage interviews as an interview and interviewer
- Present their cv in a professional way
- Implement the strategies of supportive communication
- Overcoming listener barriers

IV. Required Text:

- Business and professional communication, Steven A. Beebe, Timothy P. Mottet, Pearson first /E 2010

V. References:

Course Title and Code	CS342- Visual Programming

I. Course Identification and General Information:

Course Title	Visual Programming	Course Code	CS342	Pre-requisite	CS182
Department	Computer Science	Course Level	7	Credit Hours	3(2+1)

II. Course Description/Topics: The following course topics will be covered.

- Part I:
  1. Getting Visual J++ Set Up
  2. Classes, Interfaces and Packages
  3. The Java API
  4. Hand-Written UI Code
  5. Creating an AWT Workspace
  6. Writing the AWT Program
  7. Touring the UI API
  8. HTML and Applets
- Part II:
  1. The Visual J++ Developer Studio
  2. Visual J++, A Guided Tour
  3. The Friendly Wizards
  4. Events and Buttons
  5. Frames, Windows, Menus, and Containers
  6. Entering Data
  7. Grouping Controls
  8. Drawing Tools
  9. Java Graphics for Applets
- Part III:
  1. Advanced Java Programming
  2. Exceptions Are the Rule
  3. Multithreaded Programs

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Write event handlers for use in reactive systems, such as GUIs.
- Explain why an event-driven programming style is natural in domains where programs react to external events.
- Build robust code using exception handling mechanisms.

IV. Required Text:

- ***Beck, L. "System Software – An Introduction to Systems Programming." Addison-Wesley Longman, 3rd Edition, 1997***.

V. References:

- ***Knuth, D. E., "The Art of Computer Programming, Vol. 3: Sorting and Searching." Addison-Wesley Publishing Co., Reading, Mass., 1976***

Course Title and Code	MATH316 - Numerical Methods

I. Course Identification and General Information:

Course Title	Numerical Methods	Course Code	MATH316	Pre-requisite	MATH116
Department	Computer Science	Course Level	7	Credit Hours	3 (3+0)

II. Course Description/Topics: The following course topics will be covered.

- Errors in Computation
- Taylor Series representation of a function
- Representation of Numbers
- Finding Roots of Equations
- Interpolation
- Numerical Integration
- Systems of equations
- Splines for curve fitting
- Multiple integration: double and triple integrals; applications
- Vector fields; gradient, divergence, curl, and the del operator

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

Upon completion of this course, students will be able to :

- Explain the role of and the limitations of the computer in solving mathematical and engineering problems.
- Implement mathematical algorithms to:
  - Evaluate functions
  - Find approximate roots of equations
  - Solve systems of linear equations
  - Perform numerical differentiation and integration
  - Fit a curve to a set of data.
- Discuss selected numerical algorithms for solving a variety of commonly encountered mathematical problems.
- Analyze a computation for error and discuss the types and sources of errors involved.
- Explain how error accumulates and discuss the errors inherent in using standard floating point numbers.

IV. Required Text:

- Cheney & Kincaid, Numerical Mathematics and Computing, 7 Edition, Brooks/Cole 678 pages, 2012. ISBN-13: 978-1-133-10371-4

V. References:

- Gerald, Applied Numerical Analysis, Pearson Education, 2007, 624 pages, ISBN-13: 9788131717400

Course Title and Code	CS315 -Algorithms Analysis and Design

I. Course Identification and General Information

Course Title	Algorithms Analysis and Design	Course Code	CS315	Pre-requisite	CS211
Department	Computer Science	Course Level	7	Credit Hours	3(3+0)

II. Course Description/Topics: The following course topics will be covered.

- Dynamic programming - Assembly-line scheduling - Matrix-chain multiplication - Longest common subsequence - Optimal binary search trees
- Greedy algorithms - An activity-selection problem - Elements of the greedy strategy - Huffman codes - Theoretical foundations for greedy methods - A task-scheduling problem
- Amortized analysis - Aggregate analysis - Accounting method - Potential method - Dynamic tables
- B-Trees - Definition of B-trees - Basic operations on B-trees - Deleting a key from a B-tree
- Binomial heaps - Binomial trees and binomial heaps - Operations on binomial heaps
- Fibonacci heaps - Structure of Fibonacci heaps - Mergeable-heap operations - Decreasing a key and deleting a node - Bounding the maximum degree
- Data structures for disjoint sets - Disjoint-set operations - Linked-list representation of disjoint sets - Disjoint-set forests - Analysis of union by rank with path compression
- Elementary graph algorithms - Representations of graphs - Breadth-first search - Depth-first search - Topological sort - Strongly connected components
- Minimum spanning trees - Growing a minimum spanning tree - Algorithms of Kruskal and Prim
- Single-source shortest paths - Bellman-Ford algorithm - Single-source shortest paths in directed acyclic graphs - Dijkstra's algorithm - Difference constraints and shortest paths - Proofs of shortest paths
- All-pairs shortest paths - Matrix multiplication - The Floyd-Warshall algorithm - Johnson's algorithm
- Maximum Flow - Flow networks - The Ford-Fulkerson method - Maximum bipartite matching - Push-relabel algorithms - The relabel-to-front algorithm

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

By the end of the course, one should be able to:

- Use dynamic programming and greedy to solve an appropriate problem - Eventual optimality.
- Understand the heap property and the use of heaps as an implementation of priority queues.
- Solve problems using fundamental graph algorithms, including depth-first and breadth-first search.
- Solve problems using graph algorithms, including single-source and all-pairs shortest paths, and at least one minimum spanning tree algorithm.
- Be able to implement a string-matching algorithm.
- Use recursive backtracking to solve a problem such as navigating a maze.
- Demonstrate the ability to evaluate algorithms, to select from a range of possible options, to provide justification for that selection, and to implement the algorithm in a particular context.

IV. Required Text

- Cormen, T., C. Leiserson, R. Rivest and Clifford Stein, "*Introduction to Algorithms*", 2nd Ed. MIT Press,2002

V. Reference

- Goodrich, M.T. and R. Tamassia, "*Algorithms Design, Foundations, Analysis and Internet Examples*", John Wiley & Sons, 2002

Course Title and Code	MATH329– Operations Research

I. Course Identification and General Information:

Course Title	Operations Research	Course Code	MATH 329	Pre-requisite	MATH116
Department	Computer Science	Course Level	7	Credit Hours	3(3+0)

II. Course Description/Topics: The following course topics will be covered.

- • How to use linear programming as one of the Operations Research techniques.
- • Applying of some practical applications that can be solved by linear programming and the ability to formulate linear programming problems.
- • Understanding of the mathematical properties of linear programming models, by graphical and algebraic concepts.
- • Learning how to solving linear programming problems by the Simplex method, then performing sensitivity analysis on optimal solution.
- • Understanding design the dual problems using Duality Theorize.
- • Design programming Projects for different LP application for real cases

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Understanding linear programming as one of the Operations Research techniques.
- The ability to formulate linear programming problems.
- Understanding of some practical applications that can be solved by linear programming
- Understanding of the mathematical properties of linear programming models, by graphical and algebraic concepts.
- Understanding how to solve linear programming problems by the different methods.
- Understanding the transportation problem as an example of LLP.
-

IV. Required Text:

- **Introduction – Operations Research – Taha Hamdy eighth edition 2008**

V. References:

- Hillier & Lieberman "Introduction to Operations Research", Tata,McGraw-Hill.
- www.w3schools.com – very good tutorial

Course Title and Code	CS341-Computer Graphics

I. Course Identification and General Information:

Course Title	Computer Graphics	Course Code	CS341	Pre-requisite	CS182
Department	Computer Science	Course Level	7	Credit Hours	3(2+1)

II. Course Description/Topics: The following course topics will be covered.

- Computer Graphics Applications Survey.
- **Color models** (chromaticity diagram, RGB, CMY, YIQ, HSV, and HLS color models).
- **Graphics Output Primitives** (coordinate frames, DDA, Bresenham's algorithm, circle-drawing, fill-area primitives algorithms).
- **2D Graphics** (2D Cartesian coordinates, curves and parametric equations, functions and transformations, inverse functions.
- **3D Graphics** (vectors in 3D, dot and cross product, homogeneous coordinates, correlation between Cartesian and homogeneous coordinates).
- **Geometric transformations** (2D geometric transformations, matrix representation and homogeneous coordinates, inverse transformations, 2D composite transformations).
- **Geometric representation** (Lagrange polynomials of degree n, Hermite cubic polynomial, Bernsteinpolynomial, interpolation problem, Spline interpolation, problem ofapproximation, Bezier-Bersnstein approximation, Bezier-B-Splineapproximation, quadric surfaces).

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Identify common uses of computer graphics.
- Describe the computer generation and manipulation of images.
- How humans use vision to perceive information.
- How information can be rendered on a display device
- How images can be represented by pixels.
- Describe color models and their use in graphics display devices.
- How animation can be created as a sequence of still images.
- Describe the basic process of producing continuous motion from a sequence of discrete frames.
- Describe how double-buffering can remove flicker from animation.

IV. Required Text:

- Hearn, D. and P. Baker "*Computer Graphics with OpenGL*". Pearson Prentice Hall, PearsonEducation Inc., Upper Saddle River, NJ07458, USA, 2004. PIE ISBN 0-13-120238-3.

V. References:

- Hill, F.S., "Computer Graphics using Open GL", 2nd ed., Prentice Hall, 2001
- Edward Angel, "Interactive Computer Graphics, A Top-Down Approach with OpenGL", Third Edition, Addison Wesley Longman, 2003

Course Title and Code	MATH313 – Advanced Discrete Mathematics

I. Course Identification and General Information:

Course Title	Advanced Discrete Mathematics	Course Code	MATH313	Pre-requisite	MATH212
Department	Computer Science	Course Level	8	Credit Hours	3 (3+0)

II. Course Description/Topics: The following course topics will be covered.

- 1. Number Theory and Cryptography
- 2. Induction and Recursion
- 3. Advanced Counting Techniques
- 4. Boolean Algebra
- 5. Modeling Computation

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Divisibility and Modular, Integer Representations and Algorithms, Primes and Greatest Common Divisors, Solving Congruences, Applications of Congruences, Cryptography
- Mathematical Induction, Strong Induction and Well-Ordering, Recursive Definitions and Structural Induction, Recursive Algorithms, Program Correctness.
- Applications of Recurrence Relations, Solving Linear Recurrence Relations,Divide-and-Conquer Algorithms and Recurrence Relations, Generating Functions, Inclusion–Exclusion, Applications of Inclusion–Exclusion.
- Boolean Functions, Representing Boolean Functions, Logic Gates, Minimization of Circuits.
- Languages and Grammars, Finite-State Machines with Output, Finite-State Machines with No Output, Language Recognition, Turing Machines.

IV. Required Text:

- Discrete Mathematics and its Applications, 7/e, Kenneth H. Rosen, McGraw-Hill Education, 2012, ISBN: 978-0-07-338309-5, MHID 0-07-338309-0

V. References:

- Discrete Mathematics and its Applications: with combinatorics and graph theory, Kenneth H. Rosen, McGraw-Hill Education - Europe; 7th edition (2011), ISBN13: 978-0070681880

Course Title and Code	CS383-Software Engineering

I. Course Identification and General Information:

Course Title	Software Engineering	Course Code	CS383	Pre-requisite	CS222
Department	Computer Science	Course Level	8	Credit Hours	3(3+0)

II. Course Description/Topics: The following course topics will be covered.

- Introduction to Software Engineering, FAQs in software engineering, software types, and software cost.
- Computer-Aided Software Engineering (CASE), software processes models, process iterations, and process activities.
- Project management, management tasks,risk management, and team management.
- Software requirements, functional and non-functional requirements, interface specification, software constraints, and stakeholders.
- System modeling, behavioral models, data models, object models.
- Software architecture, architecture types, system organization, and modular decomposition styles, control styles.
- Software design, object oriented approach, object oriented design process, generality, and a study of weather station system: an example, tools and environments, and use components in design.
- Software evolution, software change, software reuse, and systems reengineering.
- Software verification and validation: software testing types; component testing, system testing, integration testing, component or white box testing, black box testing, and functionality testing.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Learn fundamentals and concepts of software engineering, essential tasks and activities of S/W project management.
- Learn and understand S/W requirements engineering processes, modeling and S/W testing and evolution.
- Able to identify and collect software requirements, classify functional and none functional requirements.
- Study and differentiate between all S/W testing types, tools and environments.
- Able to identify user problems, analyze, design the required solution.
- Apply the learned concepts through a simple project in a group.
- Work effectively in teams to accomplish common tasks such as requirements gathering and system design.
- Acquire important skills for S/W requirements collection, S/W modeling, software design tasks, risks management, team work, S/W testing methods.

IV. Required Text:

- Software Engineering, Ian Sommerville, 9thEdition, (2010), ISBN-13: 978-0137035151.

V. References:

- Requirements Engineering for Software and Systems, Phillip A. Laplante, 2nd Edition, (Oct 17, 2013), Applied Software Engineering Series, ISBN-13: 978-1466560819.
- Object-Oriented Software Engineering: An Agile Unified Methodology , David Kung,1st Edition, (2013), ISBN-13: 978-0073376257
- Essentials Of Software Engineering, Frank Tsui, Orlando Karam, Barbara Bernal, 3rd Edition, (2013).

Course Title and Code	COE351 - Computer Networks

I. Course Identification and General Information:

Course Title	Computer Networks	Course Code	COE351	Pre-requisite	CS222
Department	Computer Engineering	Course Level	8	Credit Hours	3 (3+0)

II. Course Description/Topics: The following course topics will be covered.

- Introduction: What is the Internet, What is a protocol?, Network Edge, Network Core, Network Access, Physical Media, Delay and Loss in Packet-Switched Networks, Protocol Layers and their Service Models, Internet Backbones, NAPs and ISPs, Brief History of Computer Networking and the Internet.
- Application Layer: Principles of Application Layer Protocols, HTTP, FTP, and Electronic Mail in the Internet, DNS and P2P File Sharing.
- Transport Layer: Services and Principles, Multiplexing and De-multiplexing Applications, UDP, Principles of Reliable of Data Transfer: TCP case study, Principles of Congestion Control.
- Network Layer: Service Models, What is Inside a Router? IP: the Internet Protocol, Routing Algorithms, Hierarchical Routing, Routing in the Internet.
- Link Layer & LANs: Link Layer: Services, Multiple Access Protocols and LANs, LAN Addresses and ARP, Ethernet, Hubs, Bridges and Switches, PPP.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Understand the technical literature on computer networks.
- Demonstrate an insight into the design, organization, operation of state of the art, widely used networks and protocol suites.
- Learn various network protocols and algorithms
- Acquire the required skill to design simple computer networks
- Identify different layers of network reference model and the functions of each layer.
- Apply Internet Protocol Addressing and subnetting in designing networks.
- Describe background and history of networking and the Internet
- Explain how a network can detect and correct transmission errors
- Familiar with network application architectures
- Classify the different network technologies
- To be familiar with data transferring techniques using TCP
- To understand the Principles of Congestion Control
- Introducing routers and their applications in network layer
- Understanding about Addressing, Ethernet and PPP

IV. Required Text:

- Computer Networking: A Top-Down Approach Featuring the Internet. By James F. Kurose, Keith W. Ross. 6th edition, Addison-Wesley, 2012.

V. References:

- Data and Computer Communications, 9/E Author: William Stallings, Andrew S.

Course Title and Code	COE352 -Computer Networks Lab

I. Course Identification and General Information:

Course Title	Computer Networks Lab	Course Code	COE352	Co-requisite	COE351
Department	Computer Engineering	Course Level	8	Credit Hours	1 (0+1)

II. Course Description/Topics: The following course topics will be covered.

- Operating Systems and LAN Implementation
- Web and FTP Services
- DNS, SMTP, and POP3
- Remote Access Service
- IP Address Classes and DHCP
- IP Addresses subnetting and CIDR
- Network Protocol Analysis
- Protocol Analysis of TCP, UDP, and IP using TTCP tool
- Routing Between LANs (Design and configuration)
- Routing Between LANs: Static, RIP, and RIPv2
- WAN Connection using PPP With Performance

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Construct a local area network mainly Ethernet network.
- Illustrate how to install client and server Operating Systems.
- Utilize different server roles in a local area network.
- Design Internet Protocol Addressing scheme for a subnet.
- Build and design different types of networks.
- Solve computer networks problems.
- Configure different network devices.
- Configure routers and switches for security
- Simulate lab experiments using the packet tracer.

IV. Required Text:

- Computer Networking: A Top-Down Approach Featuring the Internet. By James F. Kurose, Keith W. Ross. 6th edition, Addison-Wesley, 2012.
- Computer Networks Experiments sheets.

V. References:

- Cisco website (www.cisco.com) for technical data sheets of devices.

Course Title and Code	CS348 – Optimization Techniques

I. Course Identification and General Information:

Course Title	Optimization Techniques	Course Code	CS348	Pre-requisite	MATH329
Department	Computer Science	Course Level	8	Credit Hours	3 (3+0)

II. Course Description/Topics: The following course topics will be covered.

- Unconstrained optimization theory. Convex functions and convex sets.
- Algorithms for unconstrained optimization (Steepest descent; Newton method; Conjugate gradient methods).
- Constrained optimization theory (Kuhn-Tucker conditions). Special problems: Linear programming Quadratic Programming. Algorithms for constrained optimization.
- Since this is a course on optimization techniques, you will need to be able to program with high level programming languages (*e.g.*, C/C++, Java, C#).

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Show the ability to solve optimization problems for both constrained and unconstrained cases.
- Explain the differences between constrained and unconstrained optimization problems.
- State the standard forms of the main optimization problems such as unconstrained case, linearly-constrained case, non-linearly-constrained case, quadratic case.
- Define and properly apply the conditions of optimality such as First-Order Necessary Conditions (FONC), Second-Order Necessary Conditions (SONC), Second-Order Sufficient Conditions (SOSC) and First-Order Sufficient Conditions (FOSC) in both constrained and unconstrained case.
- Apply mathematical procedures to recursively solve unconstrained optimization problems using the steepest (or gradient) method, newton method and conjugate gradient method.
- Discuss and interpret results obtained for each optimization problem solution.
- Develop computer programs written in high-level language such as C/C++, Java or C# to solve simple optimization problems and compare results given by MATLAB Optimization Toolbox™.

IV. Required Text

- Baldick, R. "Applied Optimization Formulation and Algorithms for Engineering Systems", Cambridge University Press, 2006.

V. References

- Wright, S. and J. Nocedal "Numerical Optimization", 2nd Edition, Springer, 2007

Course Title and Code	CS451- computer security

I. Course Identification and General Information:

Course Title	Introduction to computer security	Course Code	CS451	Pre-requisite	CS315
Department	Computer Science	Course Level	9	Credit Hours	3(3+0)

II. Course Description/Topics: The following course topics will be covered.

- Nature of the Threats (e.g. natural, intentional, accidental)
- Definition and need for Information Assurance
- Basic Information Assurance Concepts that should be recognized. (Confidentiality, Integrity, Availability).
- Industry, Government, and Cultural Guidelines, Standards, and Differences including topics such as HIPAA, ISO 27002, Safe Harbor, and data protection laws.
- Legal, Ethical, and Social Issues (cross-reference SP).
- Threats and Vulnerabilities.
- Motivation of Attackers.
- Protection Mechanisms.
- Incident Response.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Describe the types of threats to data and information systems.
- Describe why processes and data need protection.
- Describe the context in which Confidentiality, Integrity, and Availability are important to given processes or data.
- Describe the significant national/international level laws affecting the obligation for the protection of data.
- Describe the impact of ethics and social issues in information assurance and security.
- Describe the major vulnerabilities present in systems today and the types of attacks.
- Define the fundamental motivations for intentional malicious exploitation of vulnerabilities.
- Define the protection mechanisms that can be used to detect or mitigate malicious activity in information systems.
- Define an incident and evaluate the roles and actions taken in response to an incident.

IV. Required Text:

- William Stallings, Cryptography and Network Security: Principles and Practice (5th
- Edition), Prentice Hall, 2011.
- Charles P. Pfleeger and Shari L. Pfleeger. Security in Computing (3rd edition). Prentice-Hall. 2003. ISBN: 0-13-035548-8.

V. References:

- Mark Stamp, Information Security : Principles and Practice, Second Edition, Wiley-Interscience, 2011.
- Anderson, Ross. Security Engineering -- A Guide to Building Dependable Distributed Systems. John Wiley & Sons, 2008, Second Edition.
- Matt Bishop, Introduction to Computer Security, 1st Ed., Addison Wesley, 2005.

Course Title and Code	CS423- Systems Programming

I. Course Identification and General Information:

Course Title	Systems Programming	Course Code	CS423	Pre-requisite	CS 224
Department	Computer Science	Course Level	9	Credit Hours	(2+1) = 3

II. Course Description/Topics: The following course topics will be covered.

- Systems Programming Background – CISC and RISC machines.
- Implementation examples (VAX, Pentium Pro, UltraSparc, PowerPC, Cray T3E) and update using the Internet.
- Simplified Instructional Computer (SIC) and SIC Extended (SIC/XE).
- Assemblers for SIC.
- Assemblers for SIC/XE.
- Machine-independent assembler features (literals, symbol definitions, expressions), Control blocks and Control sections.
- One-pass and multi-pass assemblers. Implementation examples (MASM, SPARC, AIX).
- Basic loader functions, machine-dependent loader functions.
- Machine-independent loader features, Loader design options (bootstrap loader, Relative loader, Linking loader).
- Implementation examples (MS-DOS Linker, SunOS Linker, Cray MPP Linker) and update using the Internet.
- Macro processors, basic functions, machine-dependent features.
- Machine-independent features, Implementation examples (MASM macro processor, ANSI C macro Language, ELENA macro processor) and update using the Internet.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Basic systems programming concepts.
- Systems software and machine architecture.
- Design and implementation of assemblers, loaders and linkers, and macro processors.
- Machine-independent features of these software components.
- Existing implementation examples of these software components.

IV. Required Text:

- Leland Beck "An introduction to Systems programming", Addison Wesley, 1990.

V. References:

- Knuth, D. E., "The Art of Computer Programming, Vol. 3: Sorting and Searching." Addison-Wesley Publishing Co., Reading, Mass., 1976.
- Maciaszek, L. "Practical Software Engineering." Addison Wesley, 2005.
- Maciaszek, L."Requirements Analysis and System Design." Pearson Education, 2005.

Course Title and Code	CS471- Web Technologies

I. Course Identification and General Information:

Course Title	Web Technologies	Course Code	CS471	Pre-requisite	CS342
Department	Information Technology	Course Level	10	Credit Hours	3(2+1)

II. Course Description/Topics: The following course topics will be covered.

- Web Essentials: Clients, Servers, and Communication.
- The Internet-Basic Internet Protocols -The World Wide Web-HTTP request message-response message-Web Clients Web Servers-Case Study.
- Web Essentials: Clients, Servers, and Communication
- Markup Languages: XHTML
- Style Sheets: CSS
- Client-Side Programming: The JavaScript Language
- Host Objects: Browsers and the DOM
- Server-Side Programming: Java Servlets
- Representing Web Data: XML
- Separating Programming and Presentation: JSP Technology
- Web Services: JAX-RPC, WSDL, XML Schema, and SOAP
- Databases and Java Servlets

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Students will demonstrate the ability to create web pages.
- Students will demonstrate the ability to create images for web pages.
- Students will demonstrate the ability to integrate multimedia into web pages.
- Students will demonstrate their understanding of the SQL Language.
- Students will recognize the proper way of structuring a fully functional website.
- Students will utilize their design skills to create a professional website.
- Students will have an understanding of Client/Server databases.

IV. Required Text:

- Jeffrey C. Jackson, Web Technologies: A Computer Science Perspective, Prentice Hall,2007, ISDN: 9780131856035

V. References:

- Ramesh Bangia , Web Technology (including HTML,CSS,XML,ASP,JAVA),Firewall Media, 2008
- Robert. W. Sebesta, "Programming the World Wide Web", Fourth Edition, PearsonEducation, 2007.
- Deitel, Deitel, Goldberg, "Internet & World Wide Web How To Program", 3rd Edition, Pearson Education, 2006.

Course Title and Code	CS432 - Artificial Intelligence

I. Course Identification and General Information:

Course Title	Artificial Intelligence	Course Code	CS432	Pre-requisite	MATH 329
Department	Computer Science	Course Level	10	Credit Hours	3(2+1)

II. Course Description/Topics: The following course topics will be covered.

Artificial Intelligence - Introduction :
What is AI.
A brief history.
The state of the art.

Intelligent Agents:
Agents and environments.
Rationality.
PEAS(Performance measure, Environment, Actuators, Sensors).
Environment types Agent types.

Problem solving and search:
Problem-solving agents.
Problem types.
Problem formulation.
Example problems.
Basic search algorithms.

Informed search algorithms:
Best-first search.
A∗ search.
Heuristics

Local search algorithms :
Hill-climbing.
Simulated annealing.
Genetic algorithms.

Constraint Satisfaction Problems :
CSP examples.
Backtracking search for CSPs.
Problem structure and problem decomposition.
Local search for CSPs.

Game playing :
Games.
Perfect play :
minimax decisions.
α–β pruning.
Resource limits and approximate evaluation.
Games of chance.
Games of imperfect information.

Planning :
Search vs. planning
STRIPS operators
Partial-order planning

First-order logic:
Why FOL.
Syntax and semantics of FOL.
Fun with sentences.
Wumpus world in FOL.

Uncertainty :
Probability .
Syntax and Semantics .
Inference .
Independence and Bayes' Rule.

Machine learning:
Classification by examples.
Decision trees induction.
Artificial Neural networks.
Naïve Bayes Classifier.
Unsupervised learning approach- clustering.

III. Course Outcomes: Summary of the main learning outcomes for students enrolled in the course.

- Describe the nature of AI using rational act and think concepts.
- Comparing and differentiate between the concepts of optimal reasoning/behavior and human-like reasoning/behavior.
- Describe a given problem domain using the characteristics of the environments in which a better intelligent agent design can be achieved.
- Comparisons between different search algorithms in terms of their efficiency, and the optimality.
- Experience with the concept of learning from the past to predict a better future.

IV. Required Text:

Artificial Intelligence: A Modern Approach(3rd Edition), Stuart Russell, and Peter Norvig.

V. References:

- Machine learning : A Probabilistic Perspective , Kevin P. Murphy