# Savitribai Phule Pune University

*(Formerly University of Pune)*

Two - Year Post - Graduate Degree Program in

## MATHEMATICS

(Faculty of Science & Technology)

Syllabi for

## M. Sc. (Industrial Mathematics with Computer Applications)

(For Department of Mathematics, Savitribai Phule Pune University, Pune-411007)

Syllabus under National Education Policy (NEP)
With effect from Academic Year 2023-24

(1) **Title of the Program**: M. Sc. (Industrial Mathematics with Computer Applications) (IMCA)

(2) **Duration**: TWO years (Four semesters) Full-time Post - Graduate Degree Program

(3) **Intake Capacity**: 30 students

(4) **Total number of credits**: 88 credits (minimum)

(5) **Preamble**: The Department of Mathematics, Savitribai Phule Pune University, right from its inception in 1949, is recognized as a leading Department in teaching as well as research in Mathematics and allied areas. The Department was recognized as a Centre for Advanced Study (CAS-II) by the UGC (2016-21), Govt of India. Prior to CAS-II status, the Department was also recognized at the national level under various schemes, such as, DSA (UGC), COSIST(DST), FIST (DST), and NBHM. The UGC has identified three thrust areas for the Department, namely Algebra, Analysis and Discrete Mathematics for CAS. The Department offers M.A./M.Sc. (Mathematics), M.Sc. (Industrial Mathematics with Computer Applications) and Ph. D. (Mathematics) programs.

(6) **Program Objectives:** Some of the main objectives of M.Sc. (Industrial Mathematics with Computer Applications) Program are as follows:

(a) To provide students with an environment that is conducive to their academic development and overall progress
(b) To impart students the mathematical knowledge and necessary skills so that they develop a better appreciation and understanding of modern mathematics and current technical advancements.
(c) To motivate students to explore various opportunities in the field of academics and industrial sector
(d) To develop competence among the students so that they are able to apply mathematics and technical knowledge in various spheres of life

(7) **Program Outcome:** After successful completion of 2 years M.Sc. (Industrial Mathematics with Computer Applications) PG Programme from the Department of Mathematics, Savitribai Phule Pune University, a student is expected to have a proficiency in understanding and confidence in implementation of the knowledge gained, along with the following program-cum-learning outcomes.

(a) A competent student will be able to join Industry for a job.
(b) A student will be able to demonstrate mastery of subject material, as exhibited by quantitative and qualitative performance in core and elective courses, and as reflected in the grade sheet.
(c) A student with an interest in the teaching profession will firmly communicate, mathematical and technical components with context and interdisciplinary importance in written and oral form, as a persistent mathematics student.
(d) A student has gained a fundamental understanding of amalgamate courses, thereby becoming an ambassador for quality education in the Department of Mathematics, Savitribai Phule Pune University.
(e) A student with an inclination towards applicable mathematics will be able to explore and avail the opportunities in academia as well as industrial sectors.
(f) A student will be able to appear for NET/SET/PET/ equivalent exams in mathematical sciences, computer science. Further, all opportunities such as teaching/research/government jobs/ private sector jobs which are available for M.Sc. Mathematics students will also be available for M.Sc. (IMCA) students.

(8) **Program Structure of M.Sc. (Industrial Mathematics with Computer Applications)**:

For M.Sc. (IMCA) Degree, a student has to earn the minimum 88 credits from at least FOUR semesters. The structure of the program is as follows:

(a) In each of the four semesters I, II, III, and IV, the Department will offer at least 22 credits.

(b) In each semester, there will be three mandatory courses and one elective course of 4 credits each.

(c) In addition to (b), there will be compulsory courses as follows:
Sem I-Research Methodology (RM) - 4 credits,
Sem II-On Job Training (OJT) - 4 credits,
Sem III- Research Project/Industrial Training – 4 credits,
Sem IV- Research Project/Industrial Training – 6 credits.

(d) Each course of 4 credits, other than OJT and RP, comprises of 2T+2P or 3T+1P (T-Theory, P-Practical).

(e) Each course of 2 credits comprises of 2T or 2P or 1T+1P.

(f) A student has to attend 1-hour classroom teaching per week for one credit of theory and 2 hours lab work/problem-solving session/ related activities per week for one credit of practical.

(g) Practical sessions (lab work/problem-solving session/related activity) will be conducted in batches. A batch for such sessions will be of size 6-12 students.

(h) The modus-operandi for the conduct and evaluation of a Research Project/Industrial Training Course will be decided by the Departmental Committee from time to time as per the needs.

(i) The Department may conduct necessary lectures/workshops as a part of OJT which are relevant to industry requirement.

(9) **Exit Option:** After successful earning of 44 credits offered by the Department for the first two semesters (First year-I, II Sem), a student will have the option of exit from the program. In this case, such a student will be conferred with PG Diploma in Industrial Mathematics with Computer Applications**.**

(10)**Evaluation Rules:** The general CBCS policy and rules of Savitribai Phule Pune University, and UGC are to be followed.

(a) Each course of 4 credits (T + P) will carry 100 marks and the evaluation of the course will be carried out by considering T and P jointly. There will be Continuous Assessment (CA) of 50 marks and End Term Examination (ETE) of 50 marks for each course.

(a) The CA will be based on minimum two internal tests (IT). In addition, a teacher may consider one or more of the following.

I. Home Assignment(s)
II. Seminar/Presentation (Individual / Group)
III. Laboratory assignment
IV. Group Discussions / Oral
V. Research Paper Review
VI. Technology Demonstration
VII. A project (mini) by a group of 1 or 2 students

(b) For passing a course including OJT and RP, a student has to score a minimum of 30% marks in each of the CA and ETE separately and a minimum of 40% marks in the combined grading of CA and ETE. If a student fails to score a minimum of 30% marks in CA in a course, then the result of such a course will be FAIL.

(c) For both OJT and RP, the CA will be based on grades awarded by guide/mentor while the ETE will be based on presentation/oral/discussion/ any other criterion decided the departmental committee. Also, for RP, the dissertation will be compulsory.

(d) If a student fails in a course (with a minimum score of 30% marks in CA), then he/she can appear only for the End Term Examination (ETE) of the following semester. However, he/she can improve the performance in CA of a “failed course” in any of the forthcoming semesters in which the course is subsequently conducted and, in this case, the student will have to appear for the ETE also for the said course.

(e) Provision of (e) can be availed within the stipulated period as per UGC/ the University norms.

(11) **ATKT Rules**: Student who wishes to take admission to the second year (register for third or fourth semester) of M. Sc. (IMCA) program must have earned at least 22 credits from the total credits of two semesters of the first year of M.Sc. (IMCA).

**(12) Research Project/Industrial Training Course Outcome: (1 Batch: 2-4 Students)**

On successful completion of the course "Research Project/Industrial Training", a student is adept at:

1) getting jobs in industry
2) undertaking problem identification, formulation and devise a mechanism to find an approximate or best-fit solution for it.
3) taking steps to bespeak his/her knowledge gained during the project period and demonstrate it as and when required in the professional career.
4) analyzing, understanding, and formalizing a task in terms and language of mathematical modeling.
5) undertaking a research career in mathematics

**Procedures and guidelines for the conduct:**

1. A student is supposed to register for the course "Research Project/Industrial Training" with a suitable guide/mentor (in Industry/Institute).
2. A student will perform the academic activities throughout the semester for a course registered in that semester.
3. A teacher from the Department of Mathematics, Savitribai Phule Pune University will supervise the progress of students.
4. Every student will submit a dissertation at the end of the semester.

(13) **On Job Training (OJT)** In this course, the students are expected to do the On Job Training (OJT) in appropriate Industries/Government sectors/institutes etc. to get hands-on experience. The department may conduct necessary lectures/workshops/seminars as a part of OJT. The course will be conducted as per the guidelines of the Department/the University and Government of Maharashtra.

(14) **Completion of the Degree Program**:

(a) In order to qualify for the award of M.Sc. (IMCA) Degree, a student has to earn minimum 88 credits and also need to complete the compulsory audit courses as prescribed by the University from time to time.

(b) If a student fails in a course then the said course will not be considered for calculation of SGPA/CGPA and overall grade. Only those courses in which the student has passed will be considered for calculating the SGPA/CGPA and overall grade.

(c) The applicable policies and procedures laid down by SPPU will be followed for the conduct of examinations, evaluations and declaration of the results.

**(15) The Departmental Committee in its meeting with the majority may introduce/design additional course(s) and include/exclude/modify the existing course(s) to accommodate the then developments**

**from time to time.**

### Savitribai Phule Pune University
### FACULTY OF SCIENCE AND TECHNOLOGY
### M. Sc. (Industrial Mathematics with Computer Applications) Programme

| Year | Level | Semester | Course Type | Course Code | Course Title | Credit | Credit Distribution |
|---|---|---|---|---|---|---|---|
| 1 | 6.0 | I | Core | IMT-101 | Linear Algebra | 4 | (4T) |
| | | | Core | IMT-102 | Discrete Mathematics | 2 | (2T) |
| | | | Core | IMT-103 | Data Structures | 2 | (2T) |
| | | | Core | IMT-104 | Programming with Python | 2 | (2T) |
| | | | Core | IMT-105 | Lab on Data Structures and Python | 4 | (4P) |
| | | | Elective | | Choose any one from List B | 4 | (2T + 2P) |
| | | | RM | IMT-RM | Research Methodology | 4 | (2T + 2P) |
| | | II | Core | IMT-201 | Algebra | 4 | (4T) |
| | | | Core | IMT-202 | Foundations of Analysis | 4 | (4T) |
| | | | Core | IMT-203 | Programming with C++ | 2 | (2T) |
| | | | Core | IMT-204 | Lab on DBMS and C++ | 4 | (4P) |
| | | | Elective | | Choose any one from List B | 4 | (2T + 2P) |
| | | | OJT/FP | IMT-OJT | On Job Training | 4 | (4P) |

**Credit distribution structure for Two Years/One Year PG Degrees**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2 | 6.5 | III | Core | IMT-301 | Graphs and Applications | 4 | (4T) |
| | | | Core | IMT-302 | Design and Analysis of Algorithms | 4 | (4T) |
| | | | Core | IMT-303 | Differential Equations | 2 | (2T) |
| | | | Core | IMT-304 | Lab on JAVA | 4 | (4P) |
| | | | Elective | | Choose any one from List B | 4 | (2T + 2P) |
| | | | RP/ Elective | DSC-RP-1 | Research Project /<br>Choose any one from List B | 4 | (2T + 2P) |
| | | IV | Core | IMT-401 | Software Engineering | 4 | (3T + 1P) |
| | | | Core | IMT-402 | Computer Networks | 4 | (3T + 1P) |
| | | | Core | IMT-403 | Optimization Techniques | 4 | (3T + 1P) |
| | | | Elective/ RP | | Choose any one from List B /<br>Research Project | 4 | (2T + 2P) |
| | | | RP | DSC-RP-2 | Research Project | 6 | (4T + 2P) |

**Abbreviations**: **OJT: On Job Training: Internship/ Apprenticeship; RM: Research Methodology; Research Project: RP; T- Theory Course, P – Practical course.**

**1T means 1hr of teaching per week and 1P means 2hrs of teaching/practical /tutorial/lab per week for 15 weeks. Hence (3T+1P=75 hrs) and (2T+2P=90 hrs).**

**Note**:

1. **In any case, students must earn 10 credits for a Research project in the second year of a PG degree.**
2. **The courses which do not have practical 'P' will be treated as 'T'.**
3. **The provided syllabus and reference books are a general outline. They can be adapted and expanded based on the specific requirements of the Department and the expertise of the faculty members teaching the course.**

**List B**

| Subject Code | Subject Title | Credits (T + P) | Credits Distribution |
|---|---|---|---|
| IME-1 | Operating Systems | 4 | 2T + 2P |
| IME-2 | Dot NET | 4 | 2T + 2P |
| IME-3 | Computer Graphics | 4 | 2T + 2P |
| IME-4 | Advanced Databases and NoSQL | 4 | 2T + 2P |
| IME-5 | Web Technologies | 4 | 2T + 2P |
| IME-6 | Image Processing | 4 | 2T + 2P |
| IME-7 | Data Mining | 4 | 2T + 2P |
| IME-8 | Machine Learning | 4 | 2T + 2P |
| IME-9 | Artificial Intelligence | 4 | 2T + 2P |
| IME-10 | Programming with Advanced Java | 4 | 2T + 2P |
| IME-11 | Programming using Mobile Technologies | 4 | 2T + 2P |
| IME-12 | Cloud Computing | 4 | 2T + 2P |
| IME-13 | Object Oriented Modeling and Design | 4 | 2T + 2P |
| IME-14 | Compiler Construction | 4 | 2T + 2P |
| IME-15 | Quantum Computing | 4 | 2T + 2P |
| IME-16 | Statistics and Probability | 4 | 2T + 2P |
| IME-17 | Statistical Inference | 4 | 2T + 2P |
| IME-18 | Computational Geometry | 4 | 2T + 2P |
| IME-19 | Complex Analysis | 4 | 2T + 2P |
| IME-20 | Operations Research | 4 | 2T + 2P |
| IME-21 | Coding Theory | 4 | 2T + 2P |
| IME-22 | Rings and Fields | 4 | 2T + 2P |
| IME-23 | Applied Linear Algebra | 4 | 2T + 2P |
| IME-24 | Financial Mathematics | 4 | 2T + 2P |
| IME-25 | Integral Transforms | 4 | 2T + 2P |
| IME-26 | Differential Geometry | 4 | 2T + 2P |
| IME-27 | Numerical Analysis | 4 | 2T + 2P |
| IME-28 | Partial Differential Equations | 4 | 2T + 2P |
| IME-29 | Cryptography | 4 | 2T + 2P |
| IME-30 | Theory of Computer Science | 4 | 2T + 2P |
| IME-31 | Emerging Technologies | 4 | 2T + 2P |
| IME-32 | Mooc/NPTEL/Swayam/Equivalent courses | 4 | 2T + 2P |

# Syllabus for Core Courses

# IMT-101: Linear Algebra

**Course Description:**

The course "Linear Algebra" provides a comprehensive introduction to the fundamental concepts and applications of linear algebra. It covers topics such as vector spaces, linear transformations, eigenvalues and eigenvectors, orthogonality, positive definite matrices, and various applications in image processing, computer graphics, pattern recognition, and the Google PageRank algorithm. The course also includes numerical computations of determinants, solutions of linear systems, eigenvalues, eigenvectors, and diagonalizability using mathematical software tools like Scilab/Matlab.

**Course Objectives:**

- To develop a solid understanding of vector spaces, linear transformations, and their properties.
- To explore canonical forms of matrices, including eigenvalues, eigenvectors, minimal polynomials, and diagonalizability.
- To comprehend the concept of orthogonality and its applications, including projection methods and the Gram-Schmidt process.
- To study positive definite matrices, their properties, tests for positive definiteness, singular value decomposition, and quadratic forms.
- To analyze various applications of linear algebra in image processing, computer graphics, pattern recognition, and the Google PageRank algorithm.
- To gain practical experience in performing numerical computations of determinants, solutions of linear systems, eigenvalues, eigenvectors, and diagonalizability using software tools like Scilab/Matlab.

**Course Contents:**

1) **Vector Spaces:** Vector spaces, subspaces, linear independence, basis and dimension, linear transformations, quotient Spaces, direct sum, the matrix of linear transformation
2) **Canonical Forms:** Eigenvalues and eigenvectors, minimal polynomial, Cayley-Hamilton theorem, diagonalizability, Jordan form (without proof).
3) **Orthogonality:** Orthogonal vectors and subspaces, cosines and projection onto lines, projections and least square methods, orthogonal basses and Gram-Schmidt process,
4) **Positive Definite Matrices:** Minima, maxima and saddle point, tests for positive definiteness, singular value decomposition, principal component analysis, quadratic forms
5) **Applications:** Image processing, **c**omputer graphics, pattern recognition, Google page rank algorithm
6) **Numerical Computations:** Numerical computations of determinants, solutions of linear systems, eigenvalues, eigenvectors, diagonalizability using mathematical softwares like Scilab/Matlab.

**Reference Books:**

- Linear Algebra by Vivek Sahay, Vikas Bist
- Linear Algebra and its applications, 5th Edition by Gilbert Strang.
- Linear Algebra, 5th Edition by Stephen Friedberg, Arnold Insel, Lawrence Spence
- Linear Algebra and Its Applications, 5th Edition by David Lay, Steven Lay, Judi McDonald
- Linear Algebra Done Right, 3rd Edition by Sheldon Axler

# IMT-102: Discrete Mathematics

## Course Description:

The course "Discrete Mathematics" introduces students to foundational mathematical concepts and structures that are crucial in various areas of computer science and beyond. The course covers order relations and structures, trees, graph theory, combinatorics, and generating functions. Students will learn about partially ordered sets, lattices, Boolean algebras, tree structures, graph properties, and combinatorial techniques. The course aims to equip students with the necessary mathematical tools and problem-solving skills applicable in algorithm design, network analysis, and other computational areas.

## Course Objectives:

- To provide a solid understanding of order relations and structures, including partially ordered sets, lattices, and finite Boolean algebras.
- To familiarize students with tree structures and their properties, such as labelled trees and minimal spanning trees.
- To explore various topics in graph theory, including Euler paths and circuits, Hamiltonian paths and circuits, transport networks, matching problems, and graph colouring.
- To develop proficiency in combinatorial techniques, including combinations, permutations, generating functions, and recurrence relations.
- To equip students with problem-solving skills using substitution method, characteristic method, generating function method, and principle of inclusion and exclusion.

## Course Contents:

1) **Order Relations and Structures:** Partially ordered set, Extremal Elements of Partially ordered sets, Lattices, Finite Boolean algebras, Functions on Boolean algebras, Circuit designs.

2) **Trees:** Trees, Labelled Trees, Tree Searching, Undirected Trees, Minimal Spanning Trees

3) **Topics in Graph Theory:** Graph, Euler Paths and Circuits, Hamiltonian Paths and Circuits, Transport Networks, Matching Problems, Colouring Graphs,

4) **Combinatorics:** Combination, Permutation, Generating Functions, Ordinary and Exponential Generating Functions, Recurrence Relation, Methods of Solution of Recurrence Relations, Substitution Method, Characteristic Method, Generating Function Method, Principle of Inclusion and Exclusion.

## Reference Books:

- Kolman, Busby, Ross, Discrete Mathematical Structures, Pearson Education, Fifth Edition, 2017.
- Purna Chandra Biswal, Discrete Mathematics and Graph Theory, PHI, Third Edition ,2012.
- Alan Tucker, Applied Combinatorics, Fourth Edition John Willey, 2003.
- John Clark, Derek Holton, A first look at Graph Theory, Allied Publishers, First Edition, 1995.

# IMT-103: Data Structures

## Course Description:

The course "Data Structures" introduces the fundamental concepts and techniques related to data structures and their applications. It covers various data structures such as stacks, queues, linked lists, trees, graphs, hash tables, and different searching and sorting techniques. The course emphasizes the implementation and analysis of data structures and algorithms using the C/C++ programming language. Students will gain a solid understanding of how to design, implement, and analyze efficient data structures to solve real-world problems.

## Course Objectives:

- To understand the fundamental concepts of data structures and their applications.
- To develop programming skills for implementing and manipulating various data structures.
- To analyze the efficiency and performance of different data structures and algorithms.
- To enhance problem-solving skills by applying appropriate data structures and algorithms.
- To apply data structures and algorithms to solve practical problems in computer science and related fields.

## Course Contents:

1) **Introduction to Data Structures**: Abstract Data Types, review of arrays and strings, structures and pointers concepts in C/C++, recursion and its efficiency.
2) **Stacks**: ADT Stack and its Operations, Memory representation and implementation, Applications of Stack- Expression Evaluation and Conversion, Polish notation and expression conversion, Need for prefix and postfix expressions, Postfix expression evaluation.
3) **Queues**: ADT Oueue and its Operations, Memory representation and implementation. Circular Queue and its advantages, and Operations. Deque-Basic concept, types (Input restricted and Output restricted), Priority Queue: Basic concept, types (Ascending and Descending), Applications of Queue
4) **Linked Lists**: Introduction, , differences with array, Dynamic implementation of Linked List, Types of Linked List: singly linked, linear and Circular Linked Lists, Doubly Linked List, Doubly Circular Linked List, Primitive Operations on Linked List : Create, Traverse, Search, Insert, Delete, Sort, Concatenate. Polynomial Manipulations-Polynomial addition. Generalized Linked List (GLL) concept, Representation of Polynomial using GLL.
5) **Trees**: Binary Trees, Binary Tree Representations, Operations (insert/delete), Tree Traversal Techniques, Threaded Binary Tree. Applications of Trees, Search Trees: AVL Tree (single and double rotations), B-Trees
6) **Graphs**: Representation (Matrix/Adjacency) and Traversal (Depth First Search/Breadth First Search), Spanning Trees, Minimal Spanning Tree (Prim’s and Kruskalss algorithm), Dijkstras Shortest Path algorithm.

7) **Hash Table**: Hash Table: Hash Function - properties of good hash function, division, multiplication, extraction, mid-square, folding and universal, Collision and its Resolution, Separate Chaining, Open Addressing (linear probing, quadratic probing, double hashing), Rehashing, Extendible Hashing
8) **Searching Techniques**: Linear Search, Binary Search (array/ binary tree) methods.
9) **Sorting Techniques**: General Background, Sorting Techniques: Bubble Sort, Insertion Sort, Selection Sort, Quick sort, Merge sort, Heap sort and Radix Sort

**Reference Books:**

- ADTs, Data Structures, and Problem Solving with C++, Author: Larry R Nyhoff, ISBN: 9788131764701, Pearson Education
- Data Structures and Algorithms in C++, 2nd Edition Michael T. Goodrich, Roberto Tamassia, David M. Mount, Wiley
- Algorithms and Data Structures: The Basic Toolbox, Mehlhorn, Kurt, Sanders, Peter, Springer, ISBN:9783540779773
- Data Structures Using C and C++ 2 Edition, (Paperback), Yedidyah Langsam, Aaron M. Tenenbaum, Moshe J. Augenstein, PHI Learning ISBN:9788120311770

## IMT-104: Programming with Python

### Course Description:

The course "Programming with Python" introduces students to the fundamentals of programming using the Python programming language. The course covers the basics of Python syntax, data types, control structures, functions, file handling, object-oriented programming, web scraping, statistics, and data manipulation using libraries such as Matplotlib and Pandas. Students will learn how to write Python programs, solve problems, handle exceptions, work with web data, and perform data analysis using Python.

### Course Objectives:

- To provide students with a solid foundation in Python programming, including syntax, variables, operators, and built-in functions.
- To familiarize students with essential data structures in Python, such as strings, lists, tuples, sets, and dictionaries, and their manipulation techniques.
- To introduce students to control structures, including conditional statements and loops, and their applications in solving programming problems.
- To teach students the concept of user-defined functions, modules, and libraries, enabling them to write modular and reusable code.
- To explore advanced topics such as exception handling, regular expressions, file handling, and package management in Python.
- To introduce students to object-oriented programming, including classes, objects, inheritance, and encapsulation.
- To provide students with an introduction to web scraping using Python and libraries such as Beautiful Soup for extracting data from websites.
- To familiarize students with data visualization using Matplotlib and data manipulation and analysis using Pandas.
- To introduce students to working with databases, specifically MongoDB, and performing CRUD operations using Python.
- To provide students with a basic understanding of web development using Python, including HTML templates, handling forms, user authentication, and interacting with databases.

### Course Contents:

1) Introduction to Python, Variables, Built-in Functions, Operators

2) Strings - Strings, Python String Index, Slicing String in Python, String Methods,
Lists – Lists, Python List Index, Slicing List in Python, List Methods,
Tuples – Tuples, Modifying Tuple Elements, Tuple Methods,
Sets- Sets, Modifying Set Items, Set Methods, Set Union & Intersection, Subset, SuperSet, Disjoint, Set difference & symmetric_difference,
Dictionaries- Dictionary, Dictionary using for loop, Modifying Dictionary Items, Dictionary Methods,
Conditionals Statements - Decision Making In Python, If, If Else, Nested If Else nested If
Loops - Why Loops?, For Loop, Range Function, While Loop, Nested Loop, Break Statement, Continue Statement, Else Class On Loop, Pass Statement

User defined Functions - Need For Functions, Function Definition, Function Arguments, Return Statement, Recursion In Python, Lambda Function,
Python Modules, List Comprehension, Higher Order Functions, Python Type Errors, Python Date time,

3) Exception Handling, Regular Expressions, File Handling, Python Package Manager

4) Classes and Objects, Inheritance

5) Introduction to Web Scraping using python - Beautiful Soup, Parsing our soup, Directional navigation, Image scraper, Improvements to our web scraper

6) Virtual Environment in python,

7) Statistics – Mathplotlib

8) Pandas – Introduction, Dataframes, Read CSV, Read JSON, Cleaning data, Corrections, Plotting

9) Python with MongoDB - Introduction and setup, Inserting documents, Bulk inserts, Counting documents, Multiple find conditions, Datetime and keywords, Indexes

10) Python web – Introduction, HTML templates, Building a MVC, Importing static files, Setting up a register form, Posting data to web.py, Creating users, Hashing passwords, Login logic, Web.py sessions, Logout functionality, Posting microblogs, Retrieving post objects, User settings and updating Mongo, Relative datetimes, Making our post dates pretty, Adding post comments, Image uploads and avatars

**Reference Books:**

- John V Guttag, Introduction to Computation and Programming Using Python, Prentice Hall of India, 2013.
- R. Nageswara Rao, Core Python Programming, Dreamtech Press, 2016.
- Wesley J. Chun(2006), Core Python Programming - Second Edition, Prentice Hall, 2006.
- Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser(2013), Data Structures and Algorithms in Pyhon, Wiley, 2013.
- Kenneth A. Lambert, Fundamentals of Python First Programs, CENGAGE Publication,2011.
- Luke Sneeringer, Professional Python, Wiley Inc.,2015.
- Mark Lutz, Learning Python, 3rd Edition, OReilly Media, Inc., 2007.
- Katharine Jarmul & Richard Lawson, Python Web Scraping - Paperback, (Packt Publishing Limited; 2nd Revised edition).

## IMT-RM: Research Methodology

**Course Description:**

The Research Methodology course is designed to equip M. Sc. (IMCA) students with the essential skills and knowledge required to conduct rigorous and effective research in their field. This course provides an overview of various research methods, techniques, and tools commonly used in mathematical research, with an emphasis on developing critical thinking, problem-solving abilities, and research ethics. Students will also gain hands-on experience in formulating research questions, designing experiments, analyzing data, and presenting and writing research findings.

**Course Objectives:**

- To comprehensively understand different research methodologies and their applications in mathematics.
- To cultivate critical thinking and analytical skills necessary for identifying research problems and formulating research questions.
- To provide practical experience in designing experiments, collecting and analyzing data, and interpreting research results.
- To foster effective communication skills for presenting research findings orally and in written form.
- To promote ethical research practices and awareness of responsible conduct in mathematical research.

**Course Contents:**

1) **Foundations of Research:**
Meaning, Objectives, Motivation, Utility, Concept of theory, Research Problem Identification, Developing a Research Plan – Exploration, Description, Diagnosis, Experimentation, Determining Experimental and Sample Designs.

2) **Research Design:**
Defining research objectives and questions, Analysis of Literature Review – Primary and Secondary Sources, Web sources for critical Literature Review such as MathSciNet, ZMATH, Scopus, Web of Science, reviewing literature and identifying research gaps.

3) **Research Methods:**
Scientific methods, Logical Methods: Deductive, Inductive, logical methods. Quantitative research methods, Qualitative research methods, Data Collection Techniques, Surveys and questionnaires, Interviews and focus groups, Observations and case studies, Experimental methods, Data Analysis and Interpretation, Statistical analysis techniques in mathematics, Qualitative data analysis methods, Visualization and interpretation of results.

4) **Research Writing and Presentation:**
Scientific/ technical Writing Structure and Components, Importance of Effective Communication. Preparing Research papers for journals, Seminars and Conferences – Design of paper using TEMPLATE, Calculations of Impact factor of a journal, citation Index, ISBN & ISSN. Preparation of Project Proposal – Time frame and work plan – Budget and Justification – Preparation and Publication of Research paper, Thesis writing. Project Reports for various funding, Writing Statement of Purpose for PhD/Post Doc etc, Writing a review of paper, Presenting research findings orally and visually, Research Collaboration and Communication, Collaborative research practices, Effective communication in mathematical research, Participating in conferences

and seminars,

5) **Research Ethics and Responsible Conduct:**
Ethics and Ethical Issues – Ethical Committees – Commercialization – copy right – royalty – Intellectual Property rights and patent law – Track Related aspects of intellectual property Rights – Reproduction of published material – Plagiarism and software to detect plagiarism– Citation and Acknowledgement – Reproducibility and accountability.

6) **Mathematical Software and Paraphrasing Software:**
Basic Latex, Beamer, Overleaf, Grammarly, QuillBot, ChatGPT, and SAGE. Particularly, introduction to SAGE: Overview of the SAGE software, installation, and user interface. Basic Algebraic Manipulations: Symbolic algebra, equations, simplifications, and algebraic manipulations. Calculus Computations: Differentiation, integration. Linear Algebra with SAGE: Matrix operations, solving linear systems, eigenvalue calculations. Discrete Mathematics with SAGE: Combinatorics, graph theory, number theory, and cryptography.

7) **Start-ups, Funding and Patents:** Basics of Start-ups, Innovation in Start-ups, Scalability of Start-ups, Entrepreneurship Business Model for Start-ups, Funding options for Start-ups, Crowdfunding, Online Platforms, Types of Crowdfunding models, Basics of Patents, Definition and Types of Patents, Patent Application Process, Patent Rights and Duration.

**Course Assessment:**

The course assessment will include but not be limited to a combination of the following methods:

- Research/ Technical proposals and progress reports
- Research/Technical presentations
- Critical analysis of published research papers
- Participation in class discussions and activities
- Final research/ technical project or paper

**Note:** The syllabus provided above is a general outline and can be adapted and expanded based on the specific requirements of the institution offering this subject in the M. Sc. (IMCA) program and the expertise of the instructor.

**References:**

- Kothari, C.R.(2008), Research Methodology: Methods and Techniques. Second Edition. New Age International Publishers, New Delhi.
- Dilip Datta, LaTeX in 24 Hours, A Practical Guide for Scientific Writing, Springer
- Eva O. L. Lantsoght, The A-Z of the PhD Trajectory -A Practical Guide for a Successful Journey, Springer Cham, 2018.
- NPTEL: SAGE https://onlinecourses.nptel.ac.in/noc22_ma24/preview

## IMT-201: Algebra

**Course Description:**

The course "Algebra" introduces students to the fundamental concepts and structures of algebraic systems, with a focus on groups. It covers topics such as group theory, subgroup and cyclic groups, permutation groups, group homomorphisms and isomorphisms, external direct products, normal subgroups, factor groups, and the fundamental theorem of finite abelian groups. The course aims to develop students' understanding of abstract algebraic structures and their applications in various areas of mathematics.

**Course Objectives:**

- To provide students with a solid foundation in group theory and its key concepts, including groups, subgroups, cyclic groups, and permutations.
- To develop students' understanding of group homomorphisms and isomorphisms, and their role in preserving the structure of groups.
- To introduce students to external direct products and their properties, and their applications in solving problems related to groups.
- To familiarize students with the concept of normal subgroups and factor groups, and their significance in the study of group structures.
- To explore the fundamental theorem of finite abelian groups and its implications, including the classification of abelian groups.
- To enhance students' problem-solving skills by applying algebraic concepts and techniques to various mathematical problems and applications.

**Course Contents:**

1) **Introduction to Groups:** Symmetries of a Square, The Dihedral Groups, Definition and Examples of Groups, Elementary Properties of Groups.
2) **Subgroups and Cyclic Groups:** Terminology and Notation, Subgroup Tests, Examples of Subgroups, Properties of Cyclic Groups, Classification of Subgroups of Cyclic Groups, Properties of Cosets, Lagrange’s Theorem and Consequences.
3) **Permutation Groups:** Definition and Notation, Cycle Notation, Properties of Permutations, An application of Cosets to Permutation Groups, The Rotation Group of a Cube and a Soccer Ball.
4) **Group Homomorphism and Isomorphism:** Definition and Examples of Homomorphism, Properties of Homomorphism. Definition and Examples of Isomorphism, Properties of Isomorphism, Cayley’s Theorem, The First Isomorphism Theorem, Automorphism.
5) **External Direct Products:** Definition and Examples, Properties of External Direct Products, The Group of Units Modulo n as an External Direct Product, Applications.
6) **Normal Subgroups and Factor Groups:** Normal Subgroups, Factor Groups, Applications of Factor Groups, Internal Direct Products.
7) **Fundamental Theorem of Finite Abelian Groups:** The Fundamental Theorem, The Isomorphism Classes of Abelian Groups. Reference

**Reference Books:**

- Joseph A. Gallian, Contemporary Abstract Algebra (Fourth Ed.), Narosa, 1999.
- D. S. Dummit and R. M. Foote, Abstract Algebra (Third Edition), Wiley, 2011.
- P. B. Bhattacharya, S. K. Jain and S. R. Nagpaul, Basic Abstract Algebra (Second Ed.), Cambridge Univ. Press (Indian Ed. 1995).
- I. S. Luthar and I. B. S. Passi, Algebra-Vol. 1: Groups, Narosa, New Delhi, 1996

## IMT-202: Foundations of Analysis

### Course Description:

The course "Foundations of Analysis" provides students with a rigorous introduction to the fundamental concepts and techniques of mathematical analysis. It covers topics such as limits and continuity, sequences, completeness, compactness, connectedness, uniform continuity, differentiability, partial derivatives, Taylor's theorem, implicit functions, multiple integrals, line integrals, Green's theorem, surface integrals, vector derivatives, the divergence theorem, and Stokes' theorem. Through theoretical discussions, proofs, and problem-solving exercises, students will develop a solid foundation in mathematical analysis and enhance their analytical and problem-solving skills.

### Course Objectives:

- To explore the notions of completeness, compactness, and connectedness, and their role in characterizing properties of sets and functions.
- To investigate differentiability in one and several variables, including the chain rule, mean value theorem, higher-order partial derivatives, Taylor's theorem, and critical points.
- To understand vector-valued functions and their derivatives, and their applications in studying curves in the plane and surfaces in space.
- To study the implicit function theorem and its applications in analyzing functional relations and implicit functions.
- To analyze transformations and coordinate systems and their impact on curves and surfaces.
- To explore the concept Lebesgue of integration, including line integration, integration in higher dimensions, multiple integrals, and change of variables for multiple integrals.
- To study functions defined by integrals and improper integrals, and their properties.

### Course Contents:

1) A Taste of Topology: Metric space concepts, Compactness, Connectedness, Coverings, Cantor sets.
2) Functions of a Real Variable: Differentiation, Reimann integration, Series.
3) Function Spaces: Uniform convergence and $C^0[a, b]$, Power series, Compactness and equicontinuity in $C^0$, Uniform Approximation in $C^0$, Contractions and ODE's, Analytic Functions.
4) Multivariable Calculus: Higher derivatives, Smoothness Classes, The Mean Value Theorem, Implicit and Inverse Functions, Multiple Integrals, Differential Forms, The General Stroke's Formula
5) Lebesgue Theory: Outer measure, Measurability, Regularity, Lebesgue integrals, Lebesgue integrals as limits, Lebesgue's Fundamental Theorem of Calculus.

### Reference Books:

- Real Mathematical Analysis, by C. C. Pugh, Springer, New Delhi, 2004
- Functions of Several Real Variables, by Martin Moskowitz and Fotios Paliogiannis.

- Methods of Real Analysis, by R. R. Goldberg, Oxford & IBH Publishing Company, 2019
- Advanced Calculus, by Gerald B. Folland, Pearson, 2012.
- N. L. Carothers, Real analysis, Cambridge University Press India, 1999.
- Foundations of Analysis, by Joseph L. Taylor, AMS, 2012.

## IMT-204: Programming with C++

### Course Description:

The course "Programming with C++" introduces students to the fundamentals of object-oriented programming using the C++ programming language. It covers various aspects of C++ programming, including basic syntax, functions, object and classes, arrays and string arrays, operator overloading, inheritance, pointers, memory management, virtual functions, streams and files, and templates and exceptions. Through hands-on exercises and practical examples, students will gain proficiency in C++ programming and develop a strong understanding of object-oriented concepts.

### Course Objectives:

- To understand the concepts and characteristics of object-oriented programming and the importance of using object-oriented languages like C++.
- To familiarize students with the basics of C++ programming, including output and input operations, type conversions, and manipulators.
- To explore functions in C++, including returning values, reference arguments, function overloading, inline functions, and default arguments.
- To provide a thorough understanding of objects and classes in C++, covering encapsulation, abstraction, polymorphism, class implementation, constructors, object as function arguments, copy constructors, and static class data.
- To introduce arrays and string arrays in C++, including their fundamentals and their usage as class member data.
- To explore the concept of operator overloading in C++, covering unary and binary operators, data conversion, pitfalls, and conversion keywords.
- To understand the concept of inheritance in C++, including derived classes, base classes, constructors, member functions, class hierarchies, and aggregation.
- To develop a strong understanding of pointers in C++, including addresses, pointer operations, arrays, fractions, and C-style strings.
- To cover memory management techniques in C++, including dynamic memory allocation using the new and delete operators, pointers to objects, and debugging pointers.
- To introduce virtual functions, friend functions, static functions, assignment and copy initialization, this pointer, and dynamic type information in C++.
- To explore streams and file handling in C++, including stream classes, disk file I/O, error handling, overloading extraction and insertion operators, handling command line arguments, and printer output.
- To introduce templates and exceptions in C++, covering function templates, class templates, and exception handling techniques.

### Course Contents:

1) **Introduction:** What is object-oriented programming? Why do we need object oriented?

Programming characteristics of object-oriented languages C and C++.

2) **C++ Programming basics:** Output using cout. Directives. Input with cin. Type bool. The setw manipulator. Type conversions.
3) **Functions:** Returning values from functions. Reference arguments. Overloaded function. Inline function. Default arguments. Returning by reference.
4) **Object and Classes:** Making sense of core object concepts (Encapsulation, Abstraction, Polymorphism, Classes, Messages Association, Interfaces) Implementation of class in C++, C++ Objects as physical object, C++ object as data types constructor. Object as function arguments. The default copy constructor, returning object from function. Structures and classes. Classes objects and memory static class data. Const and classes.
5) **Arrays and string arrays fundamentals. Arrays as class Member Data:** Arrays of object, string, The standard C++ String class
6) **Operator overloading:** Overloading unary operations. Overloading binary operators, data conversion, pitfalls of operators overloading and conversion keywords. Explicit and Mutable.
7) **Inheritance:** Concept of inheritance. Derived class and based class. Derived class constructors, member function, inheritance in the English distance class, class hierarchies, inheritance and graphics shapes, public and private inheritance, aggregation - Classes within classes, inheritance and program development.
8) **Pointer:** Addresses and pointers. The address of operator and pointer and arrays. Pointer and Faction pointer and C-types string.
9) **Memory management:** New and Delete, pointers to objects, debugging pointers.
10) **Virtual Function:** Virtual Function, friend function, Static function, Assignment and copy initialization, this pointer, dynamic type information.
11) **Streams and Files:** Streams classes, Stream Errors, Disk File I/O with streams, file pointers, error handling in file I/O with member function, overloading the extraction and insertion operators, memory as a stream object, command line arguments, and printer output.
12) **Templates and Exceptions:** Function templates, Class templates Exceptions

**Reference Books:**

- C++ Primer Plus, Stephen Prata, Pearson, ISBN 9788131786987,
- Programming in C++, Ashok Kamthane, Pearson, ISBN 9788131791448
- Introduction to Programming with C++, Y. Daniel Liang, Pearson, ISBN 9788131760659,
- The C++ Programming Language, B. Stroustrup Addison-Wesley ISBN 978-0321563842

## IMT-204: Lab on Database Management Systems

**Course Description:**

This course provides a comprehensive introduction to Database Management Systems (DBMS) and their role in modern business systems. Students will learn the fundamental concepts and principles of DBMS, including data modeling, database design, normalization, query processing, and optimization. The course also covers important topics such as data integrity, security, indexing, and hashing techniques for efficient data retrieval. Additionally, students will be introduced to decision support systems, data analysis, data warehousing, data mining, information retrieval, and the basics of NoSQL databases, focusing on MongoDB.

**Course Objectives:**

- Explain various data models and their application in representing business entities and relationships. Create and interpret entity-relationship (ER) models and ER diagrams for effective data representation.
- Apply principles of database design and normalization to ensure efficient and well-structured databases. Gain proficiency in SQL and its various constructs for querying and manipulating relational databases.
- Discuss the concepts of data integrity and security, and explore techniques to enforce them.
- Introduce XML as a method for handling unstructured data and discuss its applications.
- Explore indexing and hashing techniques for efficient data retrieval in database systems.
- Understand query processing and optimization techniques for improving the performance of database systems.
- Study transactions, concurrency control, and recovery mechanisms to ensure data consistency and reliability.
- Introduce the concept of NoSQL databases, with a focus on MongoDB, and understand their advantages and use cases in modern data management.

**Course Contents:**

1) Nature of Business Systems and Data Processing.
2) Data Models, ER Model, ER Diagrams
3) Database design and normalization.
4) Introduction to SQL and various construct
5) Integrity and Security.
6) Introduction to XML (unstructured data)
7) Indexing and Hashing Techniques.
8) Query processing and optimization, transactions, concurrency control and recovery.
9) Introduction to decision support and data analysis, data warehousing and data mining.
10) Information Retrieval.
11) Introduction to NoSQL e.g. MongoDB,

**Important Note:**

Teacher is supposed to take the practical's related to SQL query writing using any of the open source RDBMS engine e.g. MySQL/PostgreSQL. Some part of evaluation should be reserved for the practical aspects

**References Books:**

- Stones Richards, Beginning Databe with Poster SQL from Novice to Professionals, Academic Press
- Abraham Silberschatz,,Henry F. Korth and S. Sudarshan, Database System Concepts
- Raghu Ramakrishnan, Johannes Gehrke, Database Management Systems
- Ramez Elmasri and Shamkant Navathe, Fundamentals of Database Systems
- Korry Douglas and Susan Douglas, Postgre SQL, (second edition, Sams Publication)

## IMT-301: Graphs and Applications

**Course Description:**

The fundamental ideas of graph theory as well as some of its applications are covered in this course. Numerous real-world issues can be studied and modelled using graph theory, which is used in many different academic fields. Graph theory is used in computer science to represent networks and communications; it is used by Google search, Google Maps, and social media. It is used to investigate genomes in biology and to model molecules in chemistry. Even the social sciences and linguistics employ it. One of the newest trends is the use of graph theory in machine learning and neural networks.

**Course Objective:**

- Introduce the elements of a graph and compute its parameters
- Understand the connectivity of a graph
- See the importance of Eulerian and Hamiltonian graphs
- Solve some scheduling problems using graph colouring
- Represent a graph using a matrix
- Define matching in a graph
- Give real world examples of graph theory applications
- Model some simple real-world problems using graphs

**Course Contents:**

1) **Graph Basics:** Basic definitions and examples of paths, cycles, walks,trails etc, Bipartite graphs and characterization, Digraphs, Planarity and Graphs on Surfaces
2) **Connectivity and Trees:** Vertex Connectivity and Edge Connectivity, Trees and its properties, spanning trees, enumeration of trees, Matrix tree theorem, Minimum spanning tree, Kruskal's algorithm and Dijkstra's algorithm, Trees in Computer Science.
3) **Eulerian and Hamiltonian Graphs:** Eulerian graphs and characterization, Degree sum formula, counting and bijection, hypercubes, directed graphs, orientations
4) **Matching and its Applications:** Notions of matching, perfect matching, Halls theorem, Independent sets and covers
5) **Coloring and Applications:** Vertex Coloring and Upper Bounds, Brooks' Theorem and Color-Critical Graphs, Counting Proper Colorings.
6) **Algebraic Graph Theory**: Laplacian Matrices, Regular graphs, Quotient Graphs
7) **Graph Theory and Machine Learning:** Use of graphs in the different machine Learning algorithms like K-means, K-nearest neighbors, Decision tress, Random forests, Markov chains

**Reference Books:**

- D.B. West, Introduction to Graph Theory, Prentice Hall, 2001

- Jonathan Gross, Jay Yellen, and M. Anerson, Graph Theory and Its Applications, Taylor and Francis series.
- Jon Kleinberg and Eva Tardos, Algorithm Design, Addison-Wesley, 2005
- J.A.Bondy and U.S.R.Murty: Graph Theory, Springer, 2008.
- R.Diestel: Graph Theory, Springer( low price edition) 2000.
- F.Harary: Graph Theory, Narosa, (1988)
- C. Berge: Graphs and Hypergraphs, North Holland/Elsevier, (1973)

## IMT-302: Design and Analysis of Algorithms

### Course Description:

The course "Design and Analysis of Algorithms" provides students with a comprehensive understanding of fundamental algorithms and their analysis techniques. The course covers various topics including mathematical foundations, algorithm analysis, divide and conquer strategies, greedy algorithms, dynamic programming, graphs, sorting networks, parallel algorithms, number theoretic algorithms, NP completeness, and approximation algorithms. Through theoretical discussions and practical implementations, students will learn how to design efficient algorithms and analyze their time and space complexities.

### Course Objectives:

- To develop a solid foundation in algorithm analysis, including growth functions, summations, recurrences, and counting techniques.
- To familiarize students with fundamental sorting algorithms and their time complexities, such as insertion sort, heap sort, and bubble sort.
- To understand and apply the divide and conquer strategy to solve problems, including binary search, merge sort, quick sort, and Strassen's matrix multiplication.
- To explore greedy algorithms and their applications in solving optimization problems, such as the fractional knapsack problem, job sequencing, minimum-cost spanning trees, and Huffman coding.
- To introduce dynamic programming as a problem-solving technique, covering topics such as matrix chain multiplication, longest common subsequence, 0/1 knapsack problem, and all pairs shortest path.
- To study graph algorithms, including traversals, topological sort, minimum spanning trees, shortest path algorithms (Dijkstra's algorithm and Bellman-Ford algorithm), and maximum flow problems.
- To understand sorting networks and their applications in parallel algorithms, including bitonic sort and merge sort networks.
- To explore parallel algorithms and their efficiency in solving problems related to sorting, matrix operations, Strassen's algorithm, and matrix inversion.
- To introduce number theoretic algorithms, including Rabin-Karp, Knuth-Morris-Pratt (KMP), and Boyer-Moore algorithms.
- To provide an overview of NP completeness, nondeterministic algorithms, P and NP classes, and the concept of reducibility.
- To study approximation algorithms and their techniques for solving optimization problems with guaranteed near-optimal solutions.

### Course Contents:

1) **Mathematical Foundation and Algorithms Analysis**: Growth Functions, Summations, Recurrences Substitutions, Iterations, Master Methods, Counting and probability, Sorting algorithms (insertion sort, heap sort, bubble sort), Sorting in linear time: counting sort, concept of bucket and radix sort.

2) **Divide and conquer strategy**: Control abstraction, Binary search with time complexity. Merge sort, Quick sort with best, average and worst-case time complexity, Strassen's Matrix Multiplication with time complexity
3) **Greedy Algorithm**: Control abstraction, Fractional Knapsack problem, Job sequencing with deadlines, Minimum-cost spanning trees: Kruskal's and Prim's algorithm, Optimal storage on tapes, Optimal merge patterns, Huffman coding
4) **Dynamic Programming**: Matrix chain Multiplication, Longest common subsequence, Travelling Salesperson problem, 0/1 Knapsack Problem by Merge & Purge and Functional Method , All pairs Shortest Path (Floyd Warshall Algorithm), optimal polygon triangularization.
5) **Graphs**: Traversals, Topological sort, Minimum spanning trees, single source shortest path : Dijkstra's algorithm and Bellman- Ford algorithm, All pair shortest path : Floyd-Marshall algorithm, Connected components and strongly connected components, Maximum flow problems.
6) **Sorting Networks**: Comparison, bitonic sort and merge sort networks.
7) **Parallel Algorithms**: CRCW, EREW algorithms efficiency sorting linear system problem, Matrix Operations, Strassens Algorithm and matrix inversion
8) **Number Theoretic Algorithm**: Rabin - Karp, KMP, Bower - Moore algorithms.
9) **NP Completeness**: Nondeterministic algorithm, P and NP classes, NP completeness and reducibility.
10) **Approximation Algorithms**: Vertex cover problem, traveling salesman problem, set covering and subset sum problems.

**Reference Books:**

- T. H Coreman, Leiserson, Rivest: Introduction to Algorithms.
- Ellis Horowitz, Sartaj Sahani, Sanguthevar Rajasekaran: Computer algorithms by, Galgotia Publication

## IMT-303: Differential Equations

### Course Description:

The course "Differential Equations" focuses on the study of different types of linear and nonlinear differential equations of one and two dimensions. It explores the Lorenz model, quadratic ordinary differential equations, and other relevant topics. The course also covers the existence and uniqueness theorem for solutions of differential equations.

### Course Objectives:

- To understand the concept of different types of differential equations.
- To explore the fundamental solution theorem.
- To study the existence and uniqueness theorem for solutions of differential equations.
- To learn and apply the method of successive approximations to solve differential equations.
- To learn and apply the method of Frobenius to solve differential equations.
- To solve particular examples and exercises of differential equations.

### Course Contents:

(1) **Prerequisites:** Linear equations of the first order, Linear equations with constant coefficients,

(2) **Linear equations with variable coefficients:** Initial value problems, Solutions of the homogeneous equation, Wronskian and linear independence, Reduction of order, Non-homogeneous equations, Homogeneous equations with analytic coefficients, Legendre equation.

(3) **Linear Equations with regular singular points:** Euler equation, Second order equation with regular singular points, Exceptional cases, Bessel’s equation.

(4) **Existence and uniqueness of solutions to first order equations:** Separation of variables, exact equations, Method of successive approximations, Lipschitz condition, approximation to and uniqueness of solutions.

(5) **Existence and uniqueness of solutions to systems and n-th order equations:** Complex n-dimensional space, Systems as vector equations, Existence and uniqueness of solutions to systems, Uniqueness for linear systems and equations of order n.

### Reference Books:

- E. A. Coddington, An Introduction to Ordinary Differential Equations (Prentice- Hall).
- G. F. Simmons and S. G. Krantz, Differential Equations (Tata McGraw-Hill).

## IMT-304: Lab on JAVA

**Course Description:**

This course provides a comprehensive introduction to Java programming. It covers the foundational concepts and techniques of Java, including programming fundamentals, object-oriented programming, exception handling, input/output operations, string handling, GUI development with Swing, database connectivity with JDBC, and multithreading. Through hands-on programming exercises and projects, students will gain practical experience in Java programming and develop the skills necessary to build robust and efficient Java applications.

**Course Objectives:**

- Gain an overview of Java programming, including an understanding of Java tools and the concept of Java Byte Code.
- Master elementary programming concepts such as variables, identifiers, Java keywords, data types, operators, expressions, constants, statements, and arrays.
- Learn how to define classes, utilize static members, work with packages, understand access specifiers, constructors, and finalizers, and reference objects.
- Explore inheritance, nested classes, and inner classes, including extending classes, working with abstract classes and interfaces, utilizing the super keyword, understanding final classes, constructors, dynamic binding, and overriding methods.
- Understand exception handling and input/output operations, including working with byte streams, character streams, file I/O basics, try-catch-finally blocks, and inbuilt exceptions.
- Develop proficiency in string handling, including string operations, character extractions, data conversions, and modifying strings. Explore the Java.lang package.
- Learn about applet development, event handling, and controls for creating interactive Java applications.
- Understand input/output operations using object serialization, reader, and writer classes.
- Gain practical knowledge of Swing GUI development, including layout managers, swing controls, components organizers, JList, JTree, JTables, dialogues, file choosers, and color choosers. Develop skills in JDBC for database connectivity, including the design of JDBC, programming concepts, establishing connections, executing SQL commands, and executing queries. Further, to explore multithreading concepts, and preventing deadlocks.

**Course Contents:**

1) **Introduction to Java Programming:** Overview, Java Tools, Java Byte Code
2) **Elementary Programming Concepts:** Variables and Identifiers, Java keywords, Data Types, Operators, Expression, Constants, Statements, Arrays
3) **Classes and Packages:** Defining classes, Static Members, Using packages, Access Specifiers, Constructors, Finalisers referencing objects
4) **Inheritance, nested and inner class:** Extending classes, Abstract Class Interface, Super

Keyword, Final classes, Constructors and Inheritance, Dynamic Binding, Overriding Methods

5) **Exception and Input and Output:** Byte streams, Character streams, File i/o basics, Introduction to exception, Try and catch block and finally block, Inbuilt Exception.
6) **String Handling and Exploring Java.lang:** String Operations, Character Extractions, Data Conversions, Modifying strings. Applet and Event Handling and Controls
7) **Input and Output package:** Object serialization, reader and writer
8) **Swings:** Layout Manager Layout Manager swing Controls Components Organizers, Jlish, Jtree, Jtables, Dialogue, File chooser, color chooser.
9) **JDBC:** The design of JDBC, JDBC programming concepts making the connection, statement and result set class, Executing SQL commands, Executing Queries.
10) **Multithreading:** Running multiple threads, The runnable interface Threads priorities Daemon, Thread States, thread groups Synchronization and Interthread Communication Deadlocks.

**Practicals on each topic in the course syllabus.**

**References Books:**

- Herbert Schildt, A Complete Reference Java 2.
- Ken Arnold, James Gosling, David Holmes, The Java Programming Language
- Marc Loy, Patrick Niemeyer, Daniel Leuck, Learning Java: An Introduction to Real-World Programming with Java

# IMT-401: Software Engineering

## Course Description:

This course provides a comprehensive overview of software engineering principles and practices. Students will learn the fundamental concepts and techniques related to software management, software methodologies, software measurement, requirements analysis, software architecture, software design, implementation, user considerations, project management, maintenance, tools and environments for software engineering, and industry standards. The course also includes a practical component where students work in groups to develop a minor software project, which will be evaluated throughout the course.

## Course Objectives:

- Explore software engineering principles, including the distinction between programming in the small and programming in the large.
- Study software methodologies and processes, including the software life cycle, the waterfall model, variations, and an introduction to evolutionary and prototyping approaches.
- Gain knowledge of software measurement techniques and their importance in software engineering. Understand the process of requirements analysis, solicitation, analysis tools, requirements definition, and requirements specification.
- Learn about software architecture and its significance in software development.
- Develop skills in software design, including design for reuse, design for change, design notations, and design evaluation and validation.
- Understand the implementation phase of software development, including programming standards and procedures, modularity, data abstraction, testing techniques (unit testing, integration testing, regression testing), and tools for testing and fault tolerance.
- Understand the role of tools and environments in software engineering, the importance of programming paradigms, and the concept of process maturity.
- Introduce the Capability Maturity Model (CMM) and its variations, including People Capability Maturity Model (P-CMM), Software Acquisition Capability Maturity Model (SA-CMM), and Systems Engineering Capability Maturity Model (SE-CMM).
- Familiarize students with IEEE software engineering standards and their relevance to software development practices.
- Engage in a minor software project in groups of 2 to 3, applying the concepts and techniques learned throughout the course for continuous evaluation.

## Course Contents:

1) Concepts of software management, The software crisis, principles of software engineering, programming in the small Vs programming in the large.

2) Software methodologies/processes, The software life cycle, the waterfall model and variations, and introduction to evolutionary and prototyping approaches.

3) Software measurement, Requirements analysis, requirements.

4) Solicitation, analysis tools, requirements definition, requirements specification, static and dynamic specifications, and requirements review. (just revisited)

5) Software architecture.

6) Software design, Design for reuse, design for change, design notations, design evaluation and validation.

7) Implementation, Programming standards and procedures, modularity, data abstraction, static analysis, unit testing, integration testing, regression testing, tools for testing, fault tolerance.

8) User considerations, Human factors, usability, internationalization, user interface, documentation, user manuals Documentation, Documentation formats, tools.

9) Project management, Relationship to life cycle, project planning, project control, project organization, risk management, cost models, configuration management, version control, quality assurance, metrics.

10) Maintenance, The maintenance problem, the nature of maintenance, planning for maintenance.

11) Tools and environments for software engineering, role of programming paradigms, process maturity.

12) Introduction to Capability Maturity Model People Capability Maturity Model.

13) Software Acquisition Capability Maturity Model Systems Engineering Capability Maturity Model.

14) IEEE software engineering standards.

**Important Note:**

students who take this course are supposed to work in a group of 2 to 3 and are expected to carry out a minor project (software development) which is absolutely necessary and will be used for continuous evaluation by the respective faculty member teaching this course.

**References Books:**

- Software Engineering, Ian Sommerville, Addison Wesley, (Note : This is also the preferred textbook for the IEEE Software Engineering Certificate Program.)
- The Engineering of Software, Dick Hamlet, Joe Maybee, Addison Wesley
- Introduction to the Team Software Process, Watts S. Humphrey, Addison Wesley
- Software Engineering A Practitioner’s Approach European Adaption, Roger S. Pressman, McGraw Hill
- Software Engineering Theory and Practice, Shari Lawrence Pfleeger, Prentice Hall
- Practical Software measurement, Bob Huges, McGraw Hill
- Human Computer Interaction, Dix, Finlay, Abowd and Beale, Prentice Hall
- Software Project Management, Bob Huges & Mike Cotterell, McGraw Hill

## IMT-402: Computer Networks

### Course Description:

This course provides a comprehensive understanding of computer networks, including network hardware, network software, preference models, and network standardization. Students will explore various layers of the network stack, including the physical layer, data link layer, medium access sublayer, network layer, transport layer, and application layer. Topics covered include data communication theory, guided and wireless transmission media, error detection and correction, data link control and protocols, medium access protocols, network design issues, routing algorithms, congestion control, quality of service, transport protocols, and application layer services such as DNS, electronic mail, and the World Wide Web (WWW).

### Course Objectives:

- Gain an introduction to computer networks, including understanding network hardware, network software, preference models, and the importance of network standardization.
- Explore the physical layer of computer networks, including the theoretical basis for data communication, guided transmission media, and wireless transmission.
- Study the data link layer, including design issues, error detection and correction techniques, data link control, and protocols such as flow and error control, Stop-and-Wait ARQ, Go-Back-N ARQ, Selective Repeat ARQ, and HDLC.
- Understand the medium access sublayer, including the channel allocation problem, multiple access protocols, Ethernet (cabling and encoding), wireless LANs, Bluetooth architecture, Bluetooth applications, and data link layer switching using repeaters, hubs, bridges, switches, routers, and gateways.
- Explore the network layer, including design issues, routing algorithms, congestion control algorithms, and quality of service considerations.
- Study the transport layer, including the transport service and elements of transport protocols.
- Gain knowledge of the application layer, including DNS (Domain Name System), electronic mail, and the World Wide Web (WWW).

### Course Contents:

1) **Introduction:** Network Hardware, Network Software, Preference Models, Network Standardization.
2) **Physical Layer:** Theoretical basis for data communication, Guided Transmission Media, Wireless transmission.
3) **Data Link Layer:** Design issues, Error detection and Correction: Type of errors, detection and correction of errors Data Link Control & Protocol: Flow & error control, Stop And Wait ARQ, Go Back -N ARQ, Select Repeat ARQ, HDLC.
4) **The Medium Access Sublayer:** Channel Allocation Problem, Multiple Access Protocols,

Ethernet (Cabling, Encoding) Wireless LANs Bluetooth Architecture, Bluetooth Applications, Data link layer switching: repeaters, hubs, bridges, switches, routers, gateways.

5) **Network Layer:** Design issues, Routing algorithms, Congestion control algorithms, quality of service.
6) **Transport Layer:** Transport Service, Elements of Transport protocols
7) **Application Layer:** DNS, Electronic mail, WWW.

**Important Note:**

Students are expected to implement the algorithms taught in this course using libraries with C language on Linux Platform. Some marks are to be reserved in Continuous Evaluation/Assessment for the laboratory assignments/work.

**References Books:**

- Computer Networks, 5/e, Andrew S. Tanenbaum, David J Wetherall, Pearson Education
- Computer Networking: A Top-Down Approach, 5/e, James F. Kurose, Keith W. Ross, Pearson Education
- Data Communications and Networking, Forouzan, McGraw-Hill

## IMT-403: Optimization Techniques

**Course Description:**

The course "Optimization Techniques" provides a comprehensive study of various mathematical optimization methods used in solving optimization problems. Optimization plays a crucial role in numerous fields, including engineering, economics, operations research, and data science. This course introduces students to both linear and nonlinear optimization techniques, including linear programming, integer programming, nonlinear programming, and heuristic algorithms. The course covers theoretical foundations, algorithmic approaches, and practical applications of optimization techniques.

**Course Objectives:**

- Understand the fundamental concepts and terminology of optimization.
- Learn different optimization models and their mathematical representations.
- Study various optimization algorithms and their applications in solving real-world problems.
- Develop skills in formulating optimization problems and selecting appropriate solution methods.
- Analyze the complexity and efficiency of optimization algorithms.
- Apply optimization techniques to solve practical problems in engineering, economics, and other fields.
- Enhance problem-solving and critical thinking skills through mathematical modeling and optimization analysis.
- Gain proficiency in using optimization software tools for solving optimization problems.

**Course Contents:**

Introduction to Optimization, Basic concepts and terminology, Types of optimization problems, Classification of Optimization Problems, Classical Optimization Techniques, Linear Programming, Linear optimization models, Simplex method, Duality theory, Sensitivity analysis, Karmarkar’s Method, Integer Programming: Integer optimization models, Branch and bound methods, Cutting plane methods, Applications of integer programming, Nonlinear Programming: Unconstrained optimization, Constrained optimization, Karush-Kuhn-Tucker conditions, Gradient-based methods Convex Optimization: Convex sets and functions, Convex optimization algorithms, Applications of convex optimization Optimization in Practice: Case studies and applications in engineering, economics, and other domains, Optimization software tools (e.g., MATLAB, Python libraries) Optimization Performance Analysis: Complexity analysis of optimization algorithms, Efficiency and scalability considerations

**Reference Books:**

- Engineering Optimization Theory and Practice by S.S.Rao, New Age International (P) Ltd, Publishers.

- Multi-objective optimization using evolutionary algorithms by Kalyanmoy Deb, John Wiley Publications .

# Syllabus for Elective Courses

## IME-1: Operating Systems

### Course Description:

This course provides a comprehensive understanding of operating systems, focusing on their functionalities, types, and the support provided by computer architecture. Students will learn about system calls and the services they provide, as well as the advantages and examples of system calls in both Windows and Unix environments. The course covers various aspects of process management, including process scheduling algorithms for uniprocessor and multiprocessor systems, real-time scheduling, process synchronization using algorithms such as Peterson's Solution and the Bakery Algorithm, hardware support for process synchronization, and concepts related to threads. Memory management topics include segmentation and space allocation, linking and loading, demand paging, page replacement algorithms, and analysis of page allocation policies such as the Working Set model. The course also covers file systems, including contiguous, sequential, and indexed allocation, file system interfaces, and implementation. Other topics include I/O system components such as disk scheduling and device drivers, protection and security measures, and a case study of the Unix/Linux operating system with a focus on process management, memory management, and file management.

### Course Objectives:

- Understand the basics of operating systems, including their functionalities and different types.
- Explore the support provided by computer architecture to operating systems.
- Learn about system calls, including their introduction, the services they provide, and their features. Understand the advantages and examples of system calls in Windows and Unix.
- Study process management, including process scheduling algorithms for uniprocessor and multiprocessor systems, real-time scheduling algorithms, process synchronization using algorithms like Peterson's Solution and the Bakery Algorithm, hardware support for process synchronization, and concepts related to threads.
- Gain knowledge of memory management, including segmentation and space allocation, linking and loading, demand paging, page replacement algorithms, and analysis of page allocation policies such as the Working Set model.
- Understand file systems, including contiguous, sequential, and indexed allocation, file system interfaces, and implementation.
- Explore the I/O system, including disk scheduling and device drivers for block and character devices, as well as streams and character and block device switch tables.
- Learn about protection and security mechanisms in operating systems, including accessibility and capability lists.
- Conduct a case study of the Unix/Linux operating system, focusing on process management, memory management, and file management.

### Course Contents:

1) **Basics:** Operating System Functionalities, Types of Operating Systems, Computer Architecture support to operating Systems
2) **System Calls:** Introduction to system calls**,** Services Provided by System Calls, Features of System Calls, Advantages of system calls, Examples of a System Call in Windows and Unix
3) **Process Management:** Process Scheduling - Uniprocessor scheduling algorithms, Multiprocessor and Real-time scheduling algorithms, Process Synchronization - Peterson's Solution, Bakery Algorithm, Hardware Support to Process Synchronization, Semaphores, Critical Regions, Monitors - Deadlock prevention, deadlock avoidance and Deadlock Detection and Recovery - Bankers Algorithm, Threads
4) **Memory Management:** Segmentation and space allocation, Basics of linking and loading, Demand Paging, Page replacement algorithms, Analysis of page allocation policies -Working Set
5) **File Systems:** Contiguous, Sequential and Indexed Allocation, File system interface, File System implementation,
6) **I/O System:** Disk Scheduling, Device drivers - block and character devices, streams, Character and Block device switch tables
7) **Protection and Security:** Accessibility and Capability Lists
8) **Case Study** of Unix/Linux Operating System with reference to Process Management, Memory Management and File Management

**Reference Books:**

- Operating System Concepts 8 Edition (Paperback), Peter B. Galvin, Abraham Silberschatz, Gerg Gagne,Wiley.
- Operating Systems, Nutt, Pearson Education.
- Operating Systems, William Stallings, Pearson.
- Operating Systems, Haldar & Aravind, Pearson.
- Understanding the Linux Kernel 3rd Edition (Paperback), Daniel P Bovet, Marco Cesati, O'Reilly, ISBN: 9788184040838
- The Design of the UNIX Operating System (Paperback), Maurice J. Bach, PHI Learning, ISBN-9788120305168

## IME-2: Programming with DOT NET

### Course Description:

This course focuses on programming with the .NET framework, specifically using C# as the programming language. Students will learn the fundamentals of .NET, including the framework itself, Visual Studio.NET, assemblies, and the distinction between managed and unmanaged code. The course covers various aspects of C# programming, including basic syntax, collections and generics, exception handling, I/O streams, unsafe code, and garbage collection. Advanced topics in C# include reflection and attributes, delegates and events, serialization and deserialization, and multithreading. Students will also learn how to develop Windows Forms applications using WinForms, work with controls, user controls, and custom controls. Database programming using ADO.NET, including data management with datasets, building N-tier layered architecture applications, packaging and deployment, and debugging and diagnostics, will be covered. The course also includes an introduction to Entity Framework, LINQ for querying data, and web development using ASP.NET, covering topics such as validation controls, themes and styles, page navigation, master pages, state management, databound controls, authentication and authorization, globalization, and localization.

### Course Objectives:

- Understand the fundamentals of the MS.NET framework, including the relationship between the framework and the Visual Studio.NET development environment.
- Explore the basics of the C# programming language, including syntax, collections, generics, exception handling, I/O streams, unsafe code, and garbage collection.
- Gain proficiency in advanced C# concepts such as reflection and attributes, delegates and events, serialization and deserialization, and multithreading.
- Learn how to develop Windows Forms applications using WinForms, including working with form controls, user controls, and custom controls.
- Develop database applications using ADO.NET, including managing data with datasets, implementing N-tier layered architecture, and understanding packaging, deployment, debugging, and diagnostics.
- Gain an introduction to Entity Framework and understand its usage with the DB-first and class-first approaches.
- Explore LINQ for querying data, including writing LINQ queries in C#, performing joins, ordering results, handling null values, and exceptions in query expressions.
- Learn web development using ASP.NET, including working with validation controls, applying themes and styles, understanding ASP.NET architecture, page navigation, master pages, user controls, state management, data-bound controls, web caching, authentication and authorization, globalization, and localization.

### Course Contents:

1) **Introduction to MS.NET**: MS.NET Framework Introduction, VS.NET Introduction, Assemblies and GAC, Manage and Unmanaged code

2) **C# Language:** Basics, Collections and Generics, Exception Handling, IO Streams, Unsafe Code, Garbage Collection
3) **C# Advanced:** Reflection and Attributes, Delegate and Events, Serialization and Deserialization, Multithreading
4) **Windows Form Applications:** Developing GUI Application Using WINFORMS, Form control, User Control and Custom Control
5) **ADO.NET**: Database Programming Using ADO.NET, Managing Data using DataSet, N-Tier Layered Architecture Application, Packaging and Deployment, Debugging and Diagnostics
6) **Entity Framework:** Introduction to Entity Framework, DB first approach, Class first approach
7) **LINQ:** Overview of LINQ, Write LINQ queries in C#, Query a collection of objects, Perform grouped joins, Perform left outer joins, Order the results of a join clause, Join by using composite keys, Perform custom join operations, Handle null values in query expressions, Handle exceptions in query expressions
8) **ASP**.**NET**: Introduction to ASP, ASP.NET Introduction & Sample Programs, Validation Controls, Applying Themes and Styles to Controls, ASP.NET Architecture, Page Navigation Options, Creating a Layout Using Master Pages, User Control, ASP.NET State Management, Databound Controls, Creating Virtual Directory & Web Application, Global.asax & HttpApplication, Understanding Configuration File - Web.Config, Web Caching, Authentication & Authorization, Globalization and Localization.

**References Books:**

- Kogent, .NET 4.5 Programming 6-in-1, Learning Solutions Inc., (Wiley India Pvt. Limited) 2011.
- Joseph Albahari, C# 10 in a Nutshell: The Definitive Reference (Grayscale Indian Edition)
- John Paul Mueller, Microsoft Ado Net Entity Framework Step By Step by John Paul Mueller, PHI Learning
- Richard Hundhausen, Steven Borg, Programming Ado.Net
- Onur Gumus, Mugilan T. S. Ragupathi, ASP.NET Core 2 Fundamentals
- Fabrice Marguerie, Steve Eichert and Jim Wooley, LINQ in Action

## IME-3: Computer Graphics

### Course Description:

This course provides a comprehensive introduction to computer graphics, covering both 2D and 3D graphics. Students will learn the basics of computer graphics systems, graphics primitives, color models, and programming essentials. The course will also cover popular libraries such as three.js and paper.js for creating 3D and 2D graphics, respectively. Students will gain hands-on experience in frontend development using HTML, CSS, and JavaScript. Additionally, the course will introduce the version control system Git and its usage with GitHub. Topics covered include 2D and 3D transformations, viewing pipelines, curves and surfaces, hidden surfaces elimination techniques, and color and shading models.

### Course Objectives:

- Understand the fundamentals of computer graphics, including graphics systems, representation, presentation, interaction, and transformations.
- Learn about the uses of computer graphics and different graphics primitives. Differentiate between raster and vector graphics and understand the RGB color model and intensity.
- Gain proficiency in programming essentials, including event-driven programming and the usage of libraries such as three.js and paper.js.
- Develop frontend skills using HTML, CSS, and JavaScript to create user interfaces and basic web development. Learn the key concepts of the Git version control system and understand the workflow for managing files with Git and GitHub.
- Explore three.js, a 3D graphics library, including scene setup, camera, renderer, geometries, animation loops, materials, mesh creation, and orbit controls.
- Understand the basics of paper.js, a 2D graphics library, including paths, projects, entities, smoothing, simplifying, flattening, vector geometry, mathematical operations, and animations.
- Learn 3D transformations and viewing concepts, including translation, rotation, scaling, parallel and perspective projections, and transformations about arbitrary axes and planes.
- Gain knowledge of curves and surfaces, including polygon meshes, representing polygons, parametric curves (Hermite, Bezier, B-spline), and hidden surfaces elimination techniques (Z-buffer algorithm, backface detection, BSP tree method, Painter's algorithm, scan-line algorithm,

hidden line elimination). To explore color and shading models, including light and color models, interpolative shading models.

**Course Contents:**

1) **Introduction to Computer graphics:** Introduction to computer graphics & graphics systems. Four components of Computer Graphics Representation. Presentation, Interaction and Transformations. Uses of Computer Graphics, Graphics Primitives Pixel/Point, Raster v/s Vector, RGB color model, intensity. Programming essentials event driven programming. three.js library. paper.js library.
2) **Introduction to the typescript:** Fundamentals of typescript programming language with basic data structures. Object oriented programming with typescript. Interview programming questions implementation with typescript. User Interface, basic frontend development with HTML, CSS and javascript.
3) **Introduction to the version control system GitHub:** Learn the key concepts of the Git source control system. Compare the different states in Git and compare between branches and commits. Create and fork repositories on GitHub and push changes back after working
4) after working on them locally. Step through the entire Git workflow. Manage files with Git (move, rename, delete) and update files managed outside Git. Create branches and resolve merge conflicts.
5) **Introduction to the 3D graphics library three.js:** Basics of Three.js with many demonstrations and example code three.js Scene, Camera and Renderer. Rendering pipeline. geometries such as Box, Sphere, Icosahedron, Plane, TorusKnot and more. Setting up a Development Environment using VSCode, Git and NodeJS. Create a three.js project using NPM and package json, animation loop, frame buffers. Object3D base class, and the Rotation, Position, Scale, Visibility and Matrix properties, three.js Materials, three.js mesh creation in three.js. orbit controls.
6) **Introduction to the 2D graphics library paper.js:** Basics of paper.js with demonstration and example code, Paper.js 2D paths, projects and 2D entities, Smoothing, simplifying and flattening the 2D shapes and curves, Vector geometry and mathematical operations on paper.js. Creating animations.
7) **2D Transformations and viewing:** Basic transformations: translation, rotation, scaling; Matrix representations & homogeneous coordinates, Reflection shear; Transformation of points, lines,

parallel lines, intersecting lines. Viewing pipeline; Window to viewport coordinate transformation, clipping operations, point clipping, line clipping; Cohen Sutherland algorithm, Midpoint subdivision algorithm, Cyrus beck algorithm; Polygon clipping, Sutherland Hodgman algorithm, Weiler-Atherton Algorithm.

8) **3D transformation & viewing:** 3D transformations, translation, rotation, scaling & other transformations; Rotation about an arbitrary axis in space, reflection through an arbitrary plane; general parallel projection transformation; Three dimensional viewing, Parallel and Perspective projections.
9) **Curves and Surfaces:** Polygon meshes, Representing polygons; Parametric curves, Hermite Curves, Bezier curves, B-spline curves.
10) **Hidden surfaces Elimination:** Depth comparison, Z-buffer algorithm, Back face detection, BSP tree method, the Painter's algorithm, scan-line algorithm, Hidden line elimination, wire frame methods, fractal geometry. Color & shading models Light & color model, interpolative shading model.

**Important Note:**

- Students are expected to implement the algorithms/assignments taught in this course using three.js/paper.js using javascript on windows platform.
- Some marks are to be reserved in Continuous Evaluation/Assessment for the laboratory assignments/work.

**References Books:**

- Hearn, Baker Computer Graphics ( C version 2nd Ed.) Pearson education
  https://r105.threejsfundamentals.org/threejs/lessons/threejs-fundamentals.html
  https://paperjs.org/tutorials/
- Foley, Vandam, Feiner, Hughes Computer Graphics principles (2nd Ed.) Pearson Education. 26
- W. M. Newman, R. F. Sproull Principles of Interactive computer Graphics TMH.
- D. F. Rogers, J. A. Adams Mathematical Elements for Computer Graphics (2nd Ed.) TMH
- F S. Hill, Stephen Kelly, Computer Graphics using OpenGL, PHI Learning
- Z. Xiang, R. Plastock Schaum's outlines Computer Graphics (2nd Ed.) TMH

## IME-4: Advanced Databases and NoSQL

**Course Description:**

This course focuses on advanced concepts in databases, including NoSQL databases and their usage in modern applications. Students will learn about procedural language support for SQL using PostgreSQL/MySQL for writing functions and triggers. The course provides an overview of NoSQL databases, comparing them to the relational model, discussing ACID properties, distributed databases, consistency, the CAP theorem, and different NoSQL data models. Students will also gain an understanding of Hadoop and HDFS, including deployment, core services, federated and high availability HDFS, and multi-node clusters with Docker. The course covers MongoDB, covering the document data model, document manipulation language, replication via replica sets, design considerations, transactions, durability, indexing, auto-sharding, and MongoDB as a file system. Additionally, students will explore column store databases using Cassandra, including the column-family data model, data manipulation language, architecture, key spaces, replication, consistent hashing, and cluster node management. The course also introduces Neo4j, a graph database, covering graph theory, the graph data model, relationships as first-class citizens, graph database use cases, Neo4j design, ACID properties, transaction management, CRUD operations, graph traversal, the Neo4j REST API, and the Cypher data manipulation language.

**Course Objectives:**

- Understand procedural language support for SQL using PostgreSQL/MySQL for writing functions and triggers.
- Gain an overview of NoSQL databases, including comparing them to the relational model, understanding ACID properties, distributed databases, consistency, the CAP theorem, and various NoSQL data models.
- Learn about Hadoop and HDFS, including deployment, core services, check pointing, federated and high availability HDFS, and multi-node clusters using Docker.
- Study MongoDB, including the document data model, collections, MongoDB use cases, embedded data models, replication via replica sets, MongoDB design considerations, the CAP theorem, the MongoDB document manipulation language, transactions, durability and journaling, batch processing and aggregation, indexing, auto-sharding, shard keys, horizontal scalability, and using MongoDB as a file system.

- Explore column store databases using Cassandra, databases, tables, columns, types, replication, the CAP theorem, consistent hashing, and managing cluster nodes.
- Gain knowledge of Neo4j, a graph database, including an overview of graph theory, the graph data model, relationships as first-class citizens, graph database use cases, Neo4j design (standalone and cluster), ACID properties, transaction management with JTA, CRUD operations with the Neo4j Core API, graph traversal, the Neo4j REST API, and the Cypher data manipulation language.

**Course Contents:**

1) **Introduction to Procedural Language Support to SQL:** Writing Functions/Triggers using PostgreSQL/MySQL
2) **An Overview of NoSQL:** Review of the Relational Model, ACID Properties, Distributed Databases: Sharding and Replication, Consistency, The CAP Theorem, NoSQL Data Models.
3) **Introduction to Hadoop and HDFS:** Overview of Hadoop and DFS, HDFS Deployment, Core HDFS Services, Check Pointing, Federated and High Availability HDFS, Multi-node, Cluster with Docker.
4) **Introduction to MongoDB:** The Document Data Model, Documents and Collections, MongoDB, Use Cases, Embedded Data Models, Normalized Data, Replication via Replica Sets, MongoDB Design, MongoDB and the CAP Theorem, The MongoDB Data Manipulation Language, Transactions, Atomicity, and Documents, Durability and Journaling, Batch Processing and Aggregation, Indexing, Auto-Sharding, Shard Keys, and Horizontal Scalability, Writing to Shards, MongoDB as a File System
5) **Introduction to Column Store Databases (Cassandra):** The Column-Family Data Model, Databases and Tables, Columns, Types, and Keys, The Data Manipulation Language, Architecture, Key Spaces, Replication, and Column-Families, The CAP Theorem, Consistent Hashing, Managing Cluster Nodes
6) **Introduction to Neo4j :** Overview of Graph Theory, The Graph Data Model, Relationships as First-Class Citizens, Graph Database Use Cases, Neo4j Design: Standalone and Cluster, ACID Properties and the CAP Theorem, Transaction Management with JTA, CRUD Operations with the Neo4j Core API, Navigating Graphs with the Traversal API, The Neo4j REST API, The Cypher Data Manipulation Language, Querying as Graph Traversal

**Important Note:**

- Teacher/Students are expected to visit the websites for the above NoSQL databases and read the documentation instead of relying on books.

**Reference Books:**

- Beginning Neo4j by Chris Kemper, Apress
- Learning Neo4j 3.x - Jerome Baton, Packt Publishing Limited
- Practical MongoDB by Shakuntala Gupta Edward, Apress
- The Definitive Guide to MongoDB, by David Hows , Eelco Plugge, Peter Membrey, Apress
- Hadoop: The Definitive Guide by Tom White, Shroff Publishers & Distributers Private Limited
- Hadoop from the Beginning: The Basics by Nicholas Brown Createspace Independent Pub

## IME-5: Web Technologies

**Course Description:**

This course focuses on web technologies, with a specific emphasis on Node.js and its applications in web development. Students will learn the foundation of Node.js, including installation and executing scripts. The course covers HTTP and HTTPS protocols, server creation, ports, listening, handling requests and responses, headers and bodies, and building a basic HTTP server with static files. Students will explore file system operations, including synchronous and asynchronous I/O, path and directory operations, and working with modules using exports. The course also covers buffers, streams, and events in Node.js, including using buffers for binary data, flowing vs. non-flowing streams, asynchronous processing, and configuring event handlers. Students will learn Express, a web application framework, covering the model-view-controller pattern, front-end controllers, routes, actions, RESTful architecture, reading POST data, Handlebars helpers, and middleware. The course delves into connecting Node.js to databases, both relational (RDBMS) and NoSQL, performing CRUD operations, and building client requests to web services. Additionally, students will learn about Angular components, including their life cycle, services, single-page applications, directives, forms, pipes, and communication between components.

**Course Objectives:**

- Understand the foundation of Node.js, including installation and executing scripts.
- Gain knowledge of HTTP and HTTPS protocols, including creating a server, handling ports, listening, processing requests and responses, and working with headers and bodies. Build a simple HTTP server with static file serving capability.
- Explore file system operations in Node.js, including synchronous and asynchronous I/O, path and directory operations, and working with modules using exports. Understand module scope, construction, and package creation.
- Learn about buffers, streams, and events in Node.js, including using buffers for binary data, flowing and non-flowing streams, processing streams asynchronously, and configuring event handlers.
- Gain proficiency in Express, a web application framework, including the model-view-controller pattern, building front-end controllers, defining routes and actions, utilizing REST architecture, reading POST data, creating Handlebars helpers, and adding middleware for enhanced functionality.
- Understand how Node.js connects to databases, including both relational (RDBMS) and NoSQL databases. Learn to perform CRUD operations and build client requests to web services.

- Study Angular components, including their life cycle, services, single-page applications, directives, forms, pipes, and communication between components.

**Courses Contents:**

1) **Foundation:** The Node.js framework, Installing Node.js, Using Node.js to execute scripts
2) **HTTP and HTTPs:** Making a simple server, when to use HTTP and HTTPs, Server ports and listening, HTTP requests and responses, Request and response headers and body, Creating a response to incoming requests, Building a simple HTTP server with static files
3) **File System & Modules:** Synchronous vs. asynchronous I/O, Path and directory operations, dirname and –filename, Asynchronous file reads and writes, Defining modules with exports, Modules are singletons, Creating a package, Module scope and construction
4) **Buffers, Streams, and Events:** Using buffers for binary data, flowing vs. non-flowing streams, Streaming I/O from files and other sources, processing streams asynchronously, Configuring event handlers
5) **Express:** The model-view-controller pattern, Building a front-end controller, Defining routes, Creating actions, Using REST, Reading POST data, Building Handlebars helpers, Adding middleware
6) **Data Sources:** How Node.js connects to databases, RDBMS databases and NoSQL databases, Connecting to RDBMS and NoSQL databases, Performing CRUD operations, Building client requests to web services,
7) **Angular Components:** Component Life Cycle, Services, Single Page Applications, Directives, Forms, Pipes, Communication Between Component

**Reference Book/ URL:**

- Aristeidis Bampakos, Pablo Deeleman, Learning Angular: A no-nonsense beginner's guide to building web applications with Angular 10 and TypeScript, 3rd Edition
- Jeremy Wilken, Angular in Action
- Frank Zammetti, Modern Full-Stack Development: Using TypeScript, React, Node.js, Webpack, Python, Django, and Docker
- https://docs.angularjs.org/
- https://nodejs.org/en/docs/

# IME-6: Image Processing

**Course Description:**

Image Processing is an advanced course exploring the principles and techniques of digital image processing. The course covers a wide range of topics, including evolutionary computing and its applications, as well as the fundamentals of digital image processing. Students will learn about image acquisition, sampling, quantization, and the relationships between pixels. They will also study various image enhancement techniques in spatial and frequency domains. Also, the course covers morphological image processing, image segmentation, and representation and description methods for shapes and textures.

**Course Objectives:**

- Understand the principles and applications of evolutionary computing and its various techniques, such as genetic algorithms, ant colony optimization, Monte Carlo methods, and simulated annealing.
- Learn about selection strategies and search operators, including crossover and mutation, and their role in evolutionary algorithms.
- Study the principles and techniques of digital image processing, including image acquisition, sampling, and quantization.
- Gain practical knowledge of image enhancement techniques such as the discrete Fourier transform (DFT), frequency-domain filters like Ideal, Butterworth, and Gaussian filters, and morphological image processing operations such as dilation, erosion, opening, closing, and hit-or-miss transform.
- Learn about image segmentation techniques, including discontinuity detection, point, line, edge detection, and region-based segmentation methods.
- Understand the representation and description of shapes in digital images, including chain codes, polygonal approximations, signatures, skeletons, Fourier descriptors, statistical moments, and texture descriptors.
- Gain practical knowledge of image analysis techniques, such as boundary extraction, region filling, and regional and topological descriptors.
- Explore the concept of texture in images and learn how to analyze and describe texture using appropriate techniques.
- By the end of this course, students will have a strong foundation in evolutionary computing and digital image processing techniques. They can apply various image enhancement, segmentation, and analysis techniques to solve real-world problems. Additionally, students will gain practical

skills in using software tools and libraries commonly used in image processing applications, enabling them to pursue careers in computer vision, medical imaging, and multimedia processing.

**Course Contents:**

1) **Introduction:** What Is Digital Image Processing?, MATLAB and the Image Processing Toolbox
2) **Fundamentals:** Digital Image Representation, Reading Images, Displaying Images, Writing Images, Classes, Image Types, Converting between Classes, Array Indexing, Indexing Vectors, Some Important Standard Arrays, Introduction to M-Function Programming
3) **Intensity Transformations and Spatial Filtering:** Intensity Transformation Functions, Histogram Processing and Function Plotting, Spatial Filtering, Image Processing Toolbox, Standard Spatial Filters, Using Fuzzy Techniques for Intensity Transformations and Spatial Filtering
4) **Filtering in the Frequency Domain:** The 2-D Discrete Fourier Transform, Computing and Visualizing the 2-D OF T in MATLAB, Filtering in the Frequency Domain, Obtaining Frequency Domain Filters from Spatial Filters, Generating Filters Directly in the Frequency Domain, Highpass (Sharpening) Frequency Domain Filters, Selective Filtering
5) **Image Restoration and Reconstruction:** A Model of the Image Degradation/Restoration Process, Noise Models, Restoration in the Presence of Noise Only-Spatial Filtering, Periodic Noise Reduction Using Frequency Domain Filtering, Modeling the Degradation Function, Direct Inverse Filtering, Wiener Filtering, Constrained Least Squares (Regularized) Filtering, Iterative Nonlinear Restoration Using the Lucy-Richardson Algorithm, Blind Deconvolution, mage Reconstruction from Projections,
6) **Geometric Transformations and Image:** Transforming Points, Affine Transformations, Projective Transformations, Applying Geometric Transformations to Images, Image Coordinate Systems in MATLAB, Image Interpolation, Image Registration
7) **Colour Image Processing:** Colour Image Representation in MATLAB, Converting Between Colour Spaces, The Basics of Colour Image Processing, Colour Transformations, Spatial Filtering of Colour Images, Working Directly in RGB Vector Space
8) **Wavelets:** Background, The Fast Wavelet Transform, Working with Wavelet, Decomposition Structures, The Inverse Fast Wavelet Transform, Wavelets in Image Processing
9) **Image Compression:** Background, Coding Redundancy, Spatial Redundancy, Irrelevant Information, JPEG Compression, Video Compression,
10) **Morphological Image Processing:** Preliminaries, Dilation and Erosion, Combining Dilation and Erosion, Labeling Connected Components, Morphological Reconstruction, Gray-Scale Morphology

11) **Image Segmentation:** Point, Line, and Edge Detection, Line Detection Using the Hough , Transform, Thresholding, Region-Based Segmentation, Segmentation Using the Watershed Transform

**References Books:**

- Gonzalez, R. C., Woods, R. E., and Eddins, S. L. [2009], Digital Image Processing using MATLAB, 2nd ed., Gatesmark Publishing, Knoxville, TN
- Gonzalez, R. C. and Woods, R. E. [2002/2008] Digital Image Processing, 2nd/3rd ed., Prentice Hall
- Sonka, M., Hlavac, V., and Boyle, R. [1999], Image Processing, Analysis and Machine Vision (2nd edition), PWS Publishing, or (3rd edition) Thompson Engineering, 2007
- Anil K. Jain [2001] Fundamentals of digital image processing (2nd Edition), Prentice- Hall, NJ

## IME-7: Data Mining

**Course Description:**

Data Mining is a course that introduces the fundamental concepts, techniques, and applications of data mining. The course covers various topics such as data preprocessing, data warehousing, data mining techniques, including association rule mining and classification, accuracy measures, software for data mining, clustering, and an overview of advanced techniques in data mining. The course aims to give students the necessary knowledge and skills to analyze and extract valuable insights from large datasets.

**Course Objectives:**

- To introduce the concepts, tasks, and issues related to data mining.
- To understand the architecture of data warehousing and the use of OLAP, data cubes, and data preprocessing techniques, including data cleaning, integration, transformation, and reduction.
- To study data mining techniques such as frequent item-set, association rule and sequence mining.
- To understand classification and prediction techniques, including decision tree learning, Bayesian classification, and linear regression.
- To learn about accuracy measures for evaluating data mining models.
- To familiarize students with software tools used for data mining, such as R and Weka, and an overview of clustering techniques, including k-means, expectation-maximization, and hierarchical clustering.
- To introduce advanced topics in data mining such as active learning, reinforcement learning, text mining, graphical models, and web mining.

1) **Introduction to Data Mining:** Basic Data Mining Tasks, DM versus Knowledge Discovery in Databases, Data Mining Issues, Data Mining Metrics, Social Implications of Data Mining, Overview of Applications of Data Mining

2) **Introduction to Data Warehousing:** Architecture of DW, OLAP and Data Cubes, Dimensional Data Modeling-star, snowflake schemas, Data Preprocessing Need, Data Cleaning, Data Integration & Transformation, Data Reduction, Machine Learning, Pattern Matching
3) **Data Mining Techniques:** Frequent item-sets and Association rule mining: Apriori algorithm, Use of sampling for frequent item-set, FP tree algorithm, Graph Mining: Frequent sub-graph mining, Tree mining, Sequence Mining
4) **Classification & Prediction:** Decision tree learning, Construction, performance, attribute selection Issues: Over-fitting, tree pruning methods, missing values, continuous classes, Classification and Regression Trees (CART), Bayesian Classification, Bayes Theorem, Nave Bayes classifier, Bayesian Networks, Inference, Parameter and structure learning, Linear classifiers, Least squares, logistic, perceptron and SVM classifiers, Prediction, Linear regression, Non-linear regression.
5) **Accuracy Measures:** Precision, recall, F-measure, confusion matrix, cross-validation, bootstrap.
6) **Software for data mining and applications of data mining:** R, Weka, Sample applications of data mining.
7) **Clustering:** k-means, Expectation Maximization (EM) algorithm, Hierarchical clustering, Correlation clustering.
8) **Brief overview of advanced techniques:** Active learning, Reinforcement learning, Text mining, Graphical models, Web Mining.

**References Books:**

- Data Mining: Concepts and Techniques, Han, Elsevier
- Margaret H. Dunham, S. Sridhar, Data Mining Introductory and Advanced Topics, Pearson Education
- Tom Mitchell, Machine Learning, McGraw-Hill, 1997
- R.O. Duda, P.E. Hart, D.G. Stork. Pattern Classification. Second edition. John Wiley and Sons, 2000.
- Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer 2006
- Raghu Ramkrishnan, Johannes Gehrke, Database Management Systems, Second Edition, McGraw Hill International
- Ian H. Witten, Eibe Frank Data Mining: Practical Machine Learning Tools and Techniques, Elsevier/(Morgan Kauffman),

## IME-8: Machine Learning

**Course Description:**

Machine Learning is a course that introduces the principles, algorithms, and techniques used in machine learning. The course covers various topics such as basic definitions in machine learning, types of learning, hypothesis space, evaluation methods, linear regression, decision trees, overfitting, instance-based learning, feature reduction, collaborative filtering, probability and Bayes learning, logistic regression, support vector machines, neural networks including perceptron and backpropagation, deep neural networks, and clustering techniques such as k-means and Gaussian mixture models. The course aims to provide students with a solid foundation in machine learning and its practical applications.

**Course Objectives:**

- To introduce the basic concepts and definitions in machine learning.
- To understand the different types of learning and the notion of hypothesis space and inductive bias, evaluation methods in machine learning, including cross-validation.
- To study linear regression and its applications in modeling relationships between variables, decision trees and overfitting in machine learning models.
- To study probability and Bayes learning and their applications in machine learning.
- To explore logistic regression and support vector machines (SVM) for classification tasks.
- To study neural networks, including perceptron, multilayer networks, and backpropagation algorithm, deep neural networks and applications.
- To explore recurrent neural networks (RNN) and long short-term memory (LSTM) networks, clustering techniques such as k-means and adaptive hierarchical clustering.
- To study Gaussian mixture models for clustering.

**Course Contents:**

**1)** Basic definitions, types of learning, hypothesis space and inductive bias, evaluation, cross validation.

**2)** Linear regression, Decision trees, overfitting.

**3)** Instance based learning, Feature reduction, Collaborative filtering based recommendation.

**4)** Probability and Bayes learning.

**5)** Logistic Regression, Support Vector Machine, Kernel function and Kernel SVM.

**6)** Neural network: Perceptron, multilayer network, backpropagation, introduction to deep neural network, RNN and LSTM.

**7)** Clustering: k-means, adaptive hierarchical clustering, Gaussian mixture model.

**References Books:**

- Tom Mitchell, Machine Learning. First Edition, McGraw- Hill,(1997). (ISBN 10: 0070428077 (ISBN 13: 9780070428072)
- Ethem Alpaydin, Introduction to Machine Learning, Edition 2, The MIT Press.(2009). (ISBN 978-0-262-01243-0)

## IME-9: Artificial Intelligence

**Course Description:**

Artificial Intelligence is a course that introduces the fundamental concepts, techniques, and applications of artificial intelligence. The course covers various topics such as the introduction to artificial intelligence, problems and search algorithms, heuristic search techniques, knowledge representation, slot and filler structures, game playing, planning, and learning. The course aims to provide students with a comprehensive understanding of artificial intelligence and its practical applications.

**Course Objectives:**

- To introduce the concept of artificial intelligence and its significance in various fields, different search and control strategies for problem-solving.
- To study heuristic search techniques such as generate-and-test, hill climbing, best-first search, problem reduction, constraint satisfaction, and mean-ends analysis.
- To understand knowledge representation methods and different approaches to representing knowledge, logical reasoning and representation use propositional and predicate logic.
- To learn about slot and filler structures such as weak structures, semantic networks, frames, strong structures, conceptual dependencies, and scripts.
- To explore game-playing algorithms, including minimax search procedures and alpha-beta cutoffs, planning techniques and their applications, using examples such as the Blocks world.
- To study different learning approaches in artificial intelligence, including rote learning, learning by taking advice, learning from examples, and explanation-based learning.

**Course Contents:**

1) Introduction to Artificial Intelligence: What is AI? Early work in AI, AI and related fields AI problems and Techniques

2) Problems, Problem Spaces and Search: Defining AI problems as a State Space Search: example Production Systems Search and Control Strategies Problem Characteristics Issues in Design of Search Programs Additional Problems

3) Heuristic Search Techniques Generate-and-test, Hill Climbing, Best First Search, Problem Reduction, Constraint Satisfaction, Mean-Ends Analysis

4) Knowledge Representation Representations and Mappings, Approaches to Knowledge Representation, Knowledge representation method, Propositional Logic, Predicate logic, Representing Simple facts in Logic, Representing Instances and Isa relationships, Computable Functions and Predicates, Resolution, Forward and backward chaining

5) Slot and Filler Structures: Weak Structures, Semantic Networks, Frames, Strong Structures, Conceptual Dependencies, Scripts

6) Game Playing: Minimax Search Procedures, Adding alpha-beta cutoffs

**7)** Planning: An example Domain: The Blocks world, Component of a planning system, Goal stack planning, Nonlinear planning, Hierarchical Planning

8) Learning: What is learning, Rote Learning, Learning by taking advice, Learning in problem solving, Learning from examples, Explanation based learning Important Note: Teacher is supposed to take the practical implementation of the some of concepts in AI using Prolog language. Some marks are to be reserved in Continuous Evaluation/ Assessment for the laboratory assignments/work.

**References Books:**

- Artificial Intelligence, Tata McGraw Hill, 2nd Edition, by Elaine Rich and Kevin Knight
- Artificial Intelligence: A Modern Approach by Stuart Russell, Peter Norvig, Prentice Hall, ISBN 0-13- 103805-2
- Introduction to Artificial Intelligence and Expert System, Prentice Hall of India Pvt. Ltd., New Delhi, 1997,2nd Printing, by Dan Patterson.
- Introduction to TURBO PROLOG, BPB Publication, by Carl Townsend

## IME 10: Programming with Advanced JAVA

**Course Description:**

This course focuses on advanced Java programming concepts and their application in various domains. Students will review the basics of Java, including streams, networking, event handling, multithreading, byte code interpretation, customizing applications, data structures, and collection classes. The course covers distributed computing, including custom sockets, remote method invocation (RMI), activation, object serialization, distributed garbage collection, RMI-IIOP, interface definition language, and an overview of the CORBA and JINI frameworks. Students will explore Java Beans and Swing, including bean concepts, events in the bean box, bean customization, persistence, application deployment using Swing, advanced Swing techniques, and JAR file handling. The course also covers Java enterprise applications, including Java Native Interface (JNI), servlets, JavaServer Pages (JSP), Java Database Connectivity (JDBC), session beans, entity beans, programming and deploying enterprise JavaBeans (EJB), and Java transactions. Additionally, students will learn related Java techniques, including Java Media Framework, 3D graphics, internationalization, and case studies involving deploying n-tier applications and e-commerce applications.

**Course Objectives:**

- Review and reinforce the basics of Java, including streams, networking, event handling, multithreading, byte code interpretation, customizing applications, data structures, and collection classes.
- Gain an understanding of distributed computing concepts, including custom sockets, remote method invocation (RMI), activation, object serialization, distributed garbage collection, RMI-IIOP, interface definition language, and an overview of the CORBA and JINI frameworks.

- Learn about Java Beans and Swing, including bean concepts, event handling in the bean box, customizing beans, persistence, application deployment using Swing, advanced Swing techniques, and handling JAR files.
- Explore Java enterprise applications, including Java Native Interface (JNI), servlets, JavaServer Pages (JSP), Java Database Connectivity (JDBC), session beans, entity beans, programming and deploying enterprise JavaBeans (EJB), and Java transactions.
- Gain knowledge of related Java techniques, such as Java Media Framework, 3D graphics, internationalization, and case studies involving deploying n-tier applications and e-commerce applications.

**Course Contents:**

1) **JAVA Basic Reviews:** Java streaming - Networking - Event handling - Multithreading - Byte code Interpretation - Customizing application - Data Structures - Collection classes.

2) **Distributed Computing:** Custom sockets - Remote Method Invocation - Activation - Object serialization -Distributed garbage collection - RMI - IIOP - Interface definition language - CORBA - JINI overview.

3) **JAVA Beans And Swing:** Bean concepts - Events in bean box - Bean customization - Persistence - Application - deployment using swing - Advanced swing techniques – JAR file handling.

4) **JAVA Enterprise Applications:** JNI - Servlets - Java Server Pages - JDBC - Session beans - Entity beans - Programming and deploying enterprise Java Beans - Java transactions.

5) **Related JAVA Techniques:** Java Media Frame work - 3D graphics – Internationalization - Case study - Deploying n-tier application, E- commerce applications.

**References Books:**

- Deitel and Deitel , Java How to program, Prentice Hall , 4 th Edition, 2000.
- Gary Cornell and Cay S. Horstmann, Core Java Vol 1 and Vol 2 , Sun Microsystems Press, 1999.
- Stephen Asbury, Scott R. Weiner, Wiley, Developing Java Enterprise Applications, 1998

## IME-11: Programming using Mobile Technologies

**Course Description:**

This course focuses on programming using mobile technologies, with a specific emphasis on Android and Windows Phone app development. Students will learn the fundamentals of Android, including its architecture, development environment setup, creating Android applications, and working with the Android software stack, Linux kernel, Dalvik virtual machine, core libraries, and application framework. The course covers topics such as Android application components, activities, services, broadcast receivers, content providers, intent objects, Android manifest XML, views, view groups, layouts, graphical user interfaces, and handling user input. Students will also explore displaying pictures, working with files, databases (SQLite), intents, intent filters, threading, messaging services, location-based services, multimedia (audio, video, camera), and an introduction to Windows Phone app development, including installation of the Windows Phone SDK and creating XAML-based Windows Phone apps.

**Course Objectives:**

- Understand the basics of Android app development, including the Android architecture, versions, development environment setup, and creating Android virtual devices.
- Gain knowledge of the Android software stack, Linux kernel, Dalvik virtual machine, core libraries, Java interoperability, and application framework. Create example Android applications, configure project settings, work with layouts and resources, and modify applications.

- Learn about the Android software development platform, including Java SE, the Dalvik virtual machine, directory structure of Android projects, common default resources folders, leveraging Android XML, and launching Android applications using the AndroidManifest.xml file.
- Explore the Android framework, including application components (activities, services, broadcast receivers, content providers), Android activities for defining the user interface, Android services for background processing, intent objects for inter-component messaging, and declaring components using the Android manifest XML.
- Understand Android views, view groups, and layouts, designing user interfaces for different Android devices, working with view hierarchy, and designing user interfaces using the graphical layout tool.
- Explore files, content providers, and databases in Android, including saving and loading files, working with SQLite databases, Android database design, exposing data sources through content providers, and content provider registration.
- Understand intents and intent filters, including implicit and explicit intents, using intents with activities and broadcast receivers.
- Gain knowledge of Android threads and thread handlers, including an overview of threads, the application main thread, creating and implementing threads, and passing messages to thread handlers.
- Learn about messaging services in Android, including sending SMS messages programmatically, receiving and sending emails, and introduction to location-based services, configuring the Android emulator for location-based services, and geocoding and map-based activities.
- Gain an introduction to Windows Phone app development fundamentals, including installing the Windows Phone SDK and creating XAML-based Windows Phone apps.

**Course Contents:**

1) **Introduction:** What is Android, Android versions and its feature set The various Android devices on the market, The Android Market application store, Android Development Environment- System Requirements, Android SDK, Installing Java, and ADT bundle – Eclipse Integrated Development Environment (IDE), Creating Android Virtual Devices (AVDs)

2) **Android Architecture Overview and Creating an Example Android Application:** The Android Software Stack, The Linux Kernel, Android Runtime - Dalvik Virtual Machine, Android Runtime Core Libraries, Dalvik VM Specific Libraries, Java Interoperability Libraries, Android Libraries, Application Framework, Creating a New Android Project ,Defining the Project Name and SDK Settings, Project Configuration Settings, Configuring the Launcher Icon, Creating an Activity, Running the Application in the AVD, Stopping a Running Application, Modifying the Example Application, Reviewing the Layout and Resource Files,

3) **Android Software Development Platform:** Understanding Java SE and the Dalvik Virtual Machine , The Directory Structure of an Android Project , Common Default Resources Folders , The Values Folder , Leveraging Android XML, Screen Sizes , Launching Your Application: The AndroidManifest.xml File , Creating Your First Android Application

4) **Android Framework Overview:** Android Application Components, Android Activities: Defining the UI, Android Services: Processing in the Background, Broadcast Receivers:

Announcements and Notifications Content Providers: Data Management, Android Intent Objects: Messaging for Components Android Manifest XML: Declaring Your Components

5) **Understanding Android Views, View Groups and Layouts:** Designing for Different Android Devices, Views and View Groups, Android Layout Managers, The View Hierarchy, Designing an Android User Interface using the Graphical Layout Tool

6) **Graphical User Interface Screen with views:** Displaying Text with TextView, Retrieving Data from Users, Using Buttons, Check Boxes and Radio Groups, Getting Dates and Times from Users, Using Indicators to Display Data to Users, Adjusting Progress with SeekBar, Working with Menus using views

7) **Displaying Pictures:** Gallery, ImageSwitcher, GridView, and ImageView views to display images, Creating Animation

8) **Files, Content Providers, and Databases:** Saving and Loading Files, SQLite Databases, Android Database Design, Exposing Access to a Data Source through a Content Provider, Content Provider Registration, Native Content Providers

9) **Intents and Intent Filters:** Intent Overview, Implicit Intents, Creating the Implicit Intent Example Project, Explicit Intents, Creating the Explicit Intent Example Application, Intents with Activities, Intents with Broadcast Receivers

10) **A Basic Overview of Android Threads and Thread handlers:** An Overview of Threads, The Application Main Thread, Thread Handlers, A Basic Threading Example, Creating a New Thread, Implementing a Thread Handler, Passing a Message to the Handler

11) **Messaging and Location-Based Services:** Sending SMS Messages Programmatically, Getting Feedback after Sending the Message Sending SMS Messages Using Intent Receiving, sending email, Introduction to location-based service, configuring the Android Emulator for Location-Based Services, Geocoding and Map-Based Activities

12) **Multimedia:** Audio, Video, Camera Playing Audio and Video, Recording Audio and Video, Using the Camera to Take and Process Pictures

13) **Windows Phone App Development Fundamentals:** Introduction to Windows Phone App Development, Installing the Windows Phone SDK, Creating Your First XAML for Windows Phone App

14) **Fundamental Concepts in Windows Phone Development:** Understanding the Role of XAP Files, the Windows Phone Capabilities Model, the Threading Model for XAMLBased Graphics and Animation inWindows Phone, Understanding the Frame Rate Counter, The Windows Phone Application Analysis Tool, Reading Device Information, Applying the Model-View-ViewModel Pattern to a Windows Phone App, Property Change Notification, Using Commands

**References Books:**

- Bill Phillips, Chris Stewart, Brian Hardy, and Kristin Marsicano, Android Programming:
- The Big Nerd Ranch Guide, Big Nerd Ranch LLC, 3rd edition, 2017;
- Rajiv Ramnath, Roger Craws, and Paolo Sivilotti, Android SDK 3 for Dummies, Wiley.
- B. Phillips et al., Android Programming: Big Nerd Ranch Guide;
- Christian Keur and Aaron Hillegass, iOS Programming: The Big Nerd Ranch Guide, 6th edition, 2016;
- Valentino Lee, Heather Schneider, and Robbie Schell, Mobile Applications: Architecture,

- Design and Development, Prentice Hall, 2004;
- Tomasz Nurkiewicz and Ben Christensen, Reactive Programming with RxJava, OReilly Media, 2016;
- Raoul-Gabriel Urma, Mario Fusco, and Alan Mycroft, Java 8 in Action: Lambdas, Streams, and Functional-Style Programming, Manning Publications, 2015;

## IME 12: Cloud Computing

**Course Description:**

This course provides a comprehensive understanding of cloud computing and its key concepts. Students will learn about web services, including XML-based web services, SOAP, WSDL, and UDDI. The course covers virtualization and resource provisioning, exploring virtual machine technology, hypervisors, and multi-tenant software. Students will gain an introduction to cloud computing, including its definition, characteristics, components, cloud providers, and different service models (SAAS, PAAS, IAAS). The course also covers administering and monitoring cloud services, benefits and limitations of cloud computing, deployment of applications over the cloud, and a comparison of different cloud computing platforms. Additionally, the course focuses on security in clouds, covering cloud security fundamentals, privacy, and security considerations, identity management, access control, and challenges in cloud computing security.

**Course Objectives:**

- Understand the basics of web services, including XML, SOAP, WSDL, and UDDI, and their role in enabling interoperability between systems.
- Gain knowledge of virtualization and resource provisioning, including virtual machine technology, hypervisors, multi-tenant software, and data access control in cloud environments. Further, develop an understanding of cloud computing, including its definition, characteristics,

components, and service models (SAAS, PAAS, IAAS). Compare and contrast different cloud computing platforms.

- Learn about administering and monitoring cloud services, including deployment of applications over the cloud and the benefits and limitations of cloud computing.
- Explore security fundamentals in cloud computing, including vulnerability assessment tools, privacy, and security considerations.
- Understand cloud computing security architecture, including architectural considerations, trusted cloud computing, secure execution environments, and communications.
- Gain knowledge of identity management and access control in cloud environments, including best practices for managing identities and controlling access to resources.
- Examine the challenges and security management in virtualized environments, including virtual threats, VM-specific security techniques, and secure execution environments and communications in the cloud.

**Course Contents:**

1) **Basics of Web Services:** Extensible Mark-up Language XML Introduction, some key aspects of XML, Document-centric XML Data-centric XML, XML-based Web Services, Simple Object Access Protocol (SOAP),Web Service Definition Language (WSDL), UDDI (Universal Description Discovery and Integration) discovery that form a basis forWeb Services, exploring JAXR, jUDDI, UDDI4J etc. Technologies include HTML, HTTP, XML, SOAP, and WSDL, Development of Java Web Services.

2) **Virtualization and Resource Provisioning:** Introduction to Cloud Technologies, Study of Hypervisors Virtualization Technology: Virtual machine technology, virtualization applications in enterprises, Pitfalls of virtualization Multitenant software: Multi-entity support, Multi-schema approach, Multi-tenancy using cloud data stores, Data access control for enterprise applications

3) **Introduction to Cloud Computing:** Definition, Characteristics, Components, Cloud provider, SAAS, PAAS, IAAS and Others, Organizational scenarios of clouds, Administering & Monitoring cloud services, benefits and limitations, Deploy application over cloud, Comparison among SAAS, PAAS, IAAS Cloud computing platforms: Infrastructure as service: Amazon EC2,Platform as Service: Google App Engine, Microsoft Azure, Utility Computing, Elastic Computing

4) **Security in Clouds :** Cloud security fundamentals, Vulnerability assessment tool for cloud, Privacy and Security in cloud computing security architecture: Architectural Considerations-General Issues, Trusted Cloud computing, Secure Execution Environments and Communications, Micro-architectures; Identity Management and Access control-Identity management, Access control, Autonomic Security Cloud computing security challenges: Virtualization security management- virtual threats, VM Security Recommendations, VMSpecific Security techniques, Secure Execution Environments and Communications in the cloud.

**Important Note:**

- The teacher may take some practical demonstration of creating cloud if necessary infrastructure is made available in the laboratory (Amazon Cloud/Azure Cloud/ Google Cloud Platform)

**References Books:**

- Beginning Java web services, Henry Bequet et. al.,

- Programming web services with SOAP, James Snell et. al., O Reilly publisher
- Mastering Cloud Computing: Foundations and Applications Programming, Rajkumar Buyya, Christian Vecchiola, and Thamarai Selvi , Morgan Kaufmann Publication ISBN-10: 9780124114548 ISBN-13: 978-0124114548
- Distributed and Cloud Computing: From Parallel Processing to the Internet of Things Kai Hwang , Jack Dongarra, Geoffrey C. Fox, Elsevier; First edition ISBN-10: 9789381269237 ISBN-13: 978-9381269237

## IME-13: Object Oriented Modeling and Design

**Course Description:**

This course focuses on object-oriented modeling and design techniques for software development. Students will learn the principles and concepts of modeling as a design technique, with a specific emphasis on the Object Modeling Technique (OMT). The course covers various aspects of object modeling, including objects and classes, links and associations, generalization and inheritance, and advanced concepts such as aggregation, abstract classes, and multiple inheritance. Students will also explore dynamic modeling, functional modeling, and their relation to object models. The course provides an overview of the OMT methodology as a software engineering approach and examines the impact of an object-oriented approach on the development process. Practical analysis techniques will be applied to real-world examples, allowing students to gain hands-on experience with object-oriented modeling and design.

**Course Objectives:**

- Understand the role of modeling as a design technique in software development.
- Gain proficiency in object modeling concepts, including objects, classes, links, associations, and generalization.
- Explore advanced object modeling topics, such as aggregation, abstract classes, multiple inheritance, and metadata.

- Learn dynamic modeling techniques, including events, states, operations, and concurrency.
- Develop skills in functional modeling, including data flow diagrams, specifying operations, and constraints.
- Preview the OMT methodology as a comprehensive software engineering approach.
- Analyze real-world problems and apply object-oriented modeling techniques to create object, dynamic, and functional models.
- Understand the interrelation between object, dynamic, and functional models in the software development process.

**Course Contents:**

1) **Modeling as a Design Technique:** Modeling, The Object Modeling Technique.

2) **Object Modeling:** Objects and Classes, Links and Associations, Advanced Link and Association Concepts,Generalization and Inheritance, Grouping Constructs,A Sample Object Model.

3) **Advanced Object Modeling:** Aggregation, Abstract Classes, Generalization as Extension and Restriction, Multiple Inheritance, Metadata, Candidate Keys, Constraints.

4) **Dynamic Modeling:** Events and States, Operations, Nested State Diagrams, Concurrency, Advanced Dynamic Modeling Concepts, A Sample Dynamic Model, Relation of Object and Dynamic Models.

5) **Functional Modeling:** Functional Models, Data Flow Diagrams, Specifying Operations, Constraints, A Sample Functional Model, Relation of Functional to Object and Dynamic Models.

6) **Methodology Preview:** OMT as a Software Engineering Methodology, The OMT Methodology, Impact of an Object-Oriented Approach.

7) **Analysis:** Overview of Analysis, Problem Statement, Automated Teller Machine Example, Object Modeling, Dynamic Modeling, Functional Modeling, Adding Operations, Iterating the Analysis.

**References Books:**

- Object Oriented Modeling and Design By James Rumbaugh, Michael Blaha
- The Unified Modeling Language User Guide By Grady Booch, James Rumbaugh.
- The Unified Modeling Language Reference Manual By James Rumbaugh, Ivar Jacobson
- Object-Oriented software Engineering By Timothy, Robert Lagantere.

## IME-14: Compiler Construction

**Course Description:**

This course provides an in-depth study of compiler construction, focusing on the various phases involved in transforming source code into executable programs. Students will learn about the different analysis and synthesis tasks performed by compilers and gain hands-on experience with compiler construction tools such as Lex and Yacc. The course covers topics including lexical analysis, syntax analysis, symbol table construction and management, syntax-directed translation, abstract syntax tree construction, semantic analysis, intermediate code generation, and code optimization. Throughout the course, students will work on practical exercises and projects to reinforce their understanding of the concepts and techniques involved in compiler construction.

**Course Objectives:**

- Understand the overall process of compilation and the role of compilers in software development. To gain proficiency in lexical analysis techniques, including token specification and recognition, regular expressions, and finite automata.
- Explore different parsing methods, such as top-down parsing, bottom-up parsing, and operator-precedence parsing.

- Learn about symbol table construction, organization, and operations, including scope and overloading issues, syntax-directed translation and the use of attributes for propagating information through the syntax tree.
- Study semantic analysis techniques, including type checking rules and handling of issues like type equivalence and function/operator overloading.
- Gain proficiency in intermediate code generation, including the generation of three-address code for expressions and statements.
- Explore code optimization techniques and understand the principles of data flow analysis and code transformations.

**Course Contents:**

1) **Introduction to Compilers:** Overview of Compilation, Analysis of source program, phases of compiler, compiler construction tools namely Lex,Yacc etc.

2) **Lexical Analysis:** Specification and recognizing of tokens, Regular Expressions, Finite Automata (NFA, DFA), Algorithms for conversion from regular expression to NFA and from NFA to DFA. Implementation of Lexical Analyzer from DFA

3) **Syntax Analysis:** Context-free grammars, ambiguity specifying operator precedence, Overview of Parsing, Types of parsing including Top-down parsing, Bottom-up parsing, Operator-precedence parsing, LR parsers etc.

4) **Symbol Table Construction and Issues:** Organization, operations issues such as scope and overloading and their effect on symbol table design, implementation, and operations

5) **Syntax Directed Translation** Syntax-directed definitions, translation schemes, synthesized and inherited attributes, propagation of attribute values through syntax tree

6) **Construction of Abstract Syntax Tree:** Binary tree representation of expressions and statements, construction of binary tree representation using a stack

7) **Semantic Analysis:** Rules that cannot be described using a context-free grammar, Type checking rules for expressions and statements and issues such as type equivalence, overloading of functions and operators

8) **Intermediate Code Generation:** Three-address code for expressions and statements including assignments, conditionals, loops, procedure calls and generation of temporary variables and tables.

9) **Code Optimization:** Principal sources of optimization, Introduction to data flow analysis and equations, code improving transformations.

**References Books:**

- Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman, Compilers: Principles, Techniques, & Tools, (Second Edition, Addison-Wesley).
- David Galles, Modern Compiler Design, (First Edition,Addison-Wesley), 2005.
- Thomas Pittman and James Peters, Art of Compiler Design, The: Theory and Practice, (First Edition, Prentice Hall), 1992.

- Cooper and Torczon, Engineering a Compiler, (First Edition, Elsevier), 2004.

## IME-15: Quantum Computing

**Course Description:**

This course introduces the principles and applications of quantum computing. Students will learn the foundational concepts of quantum mechanics, explore quantum algorithms, and understand the potential of quantum technologies. The course covers topics such as quantum circuits, entanglement, quantum algorithms, quantum information, and quantum cryptography. Students will gain practical knowledge of quantum computing through theoretical discussions and hands-on exercises.

**Course Objectives:**

- To introduce students to the fundamental principles and concepts of quantum computing.
- To develop an understanding of the circuit model of quantum computation.
- To explore quantum algorithms and their applications.
- To equip students with the knowledge and skills to apply quantum algorithms to real-world problems.
- To familiarize students with quantum information theory and quantum cryptography.
- Course Contents:

**Course Contents:**

1) **Introduction to Quantum Computing:** Quantum Mechanics, Church-Turing Thesis, The Circuit Model of Computation, Linear Algebra Formulation of the Circuit Model, Reversible Computation,

2) **Quantum Computation:** Fundamentals of Quantumness, No-Cloning Theorem, Quantum Entanglement, Bell States and Bell Inequalities, Quantum Circuits, Pauli, Hadamard, Phase, CNOT, Toffoli Gates, Quantum Teleportation, Universality of Two-Qubit Gates, Reversible Computing,

3) **Quantum Algorithms:** Probabilistic Versus Quantum Algorithms, Phase Kickback, The Deutsch Algorithm, The Deutsch-Jozsa Algorithm, Simon's Algorithm, Quantum Phase Estimation and Quantum Fourier Transform, Grover's Quantum Search Algorithm, Shor's Period Finding Algorithm.

4) **Quantum Information:** Quantum Error Correction, Shannon Entropy, Von Neumann Entropy, Classical Cryptography, RSA Algorithm, Quantum Cryptography, BB84 Protocol, B92 and Eckart Protocol.

**Reference Books:**

- Phillip Kaye, Raymond Laflamme, and Michele Mosca, An Introduction to Quantum Computing, Oxford University Press, 2007.
- Michael A. Nielsen, and Isaac L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2000.
- David McMahon, Quantum Computing Explained, John Wiley & Sons, Inc., 2008.
- Mermin N. David, Quantum Computer Science: An Introduction, Cambridge University Press, 2007.

## IME 16: Statistics and Probability

**Course Description:**

The course "Statistics and Probability" introduces statistical analysis and probability theory. It covers various topics including descriptive statistics, probability concepts, random variables, discrete and continuous probability distributions, functions of random variables, hypothesis testing, and practical applications using statistical software.

**Course Objectives:**

- To understand the fundamentals of descriptive statistics and data analysis techniques.
- To introduce the concept of probability and its applications in real-world scenarios.
- To explore different types of random variables and their distributions.
- To understand the properties of discrete and continuous probability distributions.
- To study the concepts of moments and moment-generating functions.
- To learn hypothesis testing techniques and their applications in statistical inference.
- To gain practical skills in statistical analysis using the R software.

**Course Contents:**

1) **Descriptive Statistics:** Measure of central tendency, Measure of dispersion, Graphical representations of data and its interpretation, Exploratory data analysis, correlation, covariance.

2) **Introduction to Probability:** Intuitive concepts: Sample space, events, probability of an event, additive rules, conditional probability, multiplicative rule, Bayes' rule.

3) **Random Variable:** Concept of a random variable, discrete probability distribution, continuous probability distribution, joint probability distribution, independent random variables, Chebyshev's theorem. Mean of a random variable, variance and covariance, mean and covariance of linear combinations of random variables.

4) **Some Discrete Probability Distributions:** discrete uniform distribution, binomial and multinomial distributions, hypergeometric distribution, negative binomial and geometric distribution, Poisson distribution and Poisson process.

5) **Some Continuous Probability Distributions:** continuous uniform distribution, normal distribution, area under the normal curve, applications of the normal distribution, normal approximation to the binomial distribution, gamma and exponential distribution, chi-squared distribution, lognormal distribution, Weibull distribution.

6) **Functions of random variables**, transformations of variables, moments and moment generating functions.

7) **Hypothesis Testing:** Statistical Hypothesis, general concepts, testing a statistical hypothesis, types of errors in testing of hypothesis, level of significance, critical regions, use of p values for decision making, tests of significance for single mean (variance known) , tests of significance for single mean (variance unknown), confidence interval estimation.

8) **Practicals** on the all topics in course using R software.

**Reference Books:**

- Probability and Statistics for Engineers and Scientists, by R. Walpole, R.H. Myers, S.L. Myers, and K. Ye (Seventh Edition, Pearson, India).
- Statistics for Business and Economics, by Anderson, Sweeney and Williams.
- Introduction to Probability and Statistics for Engineers and Scientists, by Sheldon M. Ross (Fourth Edition).
- Mathematical Statistics, by Parimal Mukhopadhyay.
- Statistics for the Life Sciences, by M. Samules, J. Witmer and A. Schaffner (Fifth Edition, Pearson India)
- Probability and Statistics for Engineers, by Richard Gupta, C B Gupta.

## IME 17: Statistical Inference

### Course Description:

Statistical Inference is a course that focuses on statistical methods for making inferences and drawing conclusions from data. The course covers topics such as correlation and regression analysis, chi-square distribution and tests, small sample tests using t distribution and F distribution, likelihood ratio tests, non-parametric tests, and analysis of variance (ANOVA). The course provides a comprehensive understanding of these statistical techniques and their applications in data analysis and hypothesis testing.

### Course Objectives:

- To introduce correlation and regression analysis and their applications in analyzing the relationship between variables.
- To study Karl Pearson's coefficient of correlation and Spearman's rank correlation coefficient and their properties.
- To understand linear regression, lines of regression, and theorems on regression coefficients, statistical inferences about regression parameters and the estimation.
- To understand the chi-square distribution and its applications in goodness-of-fit tests and tests of independence.
- To analyze small sample tests using the t distribution, including confidence intervals, paired t-

test, and tests for correlation coefficient.

- To study the Fisher's Z-transform and the F-distribution for tests of equality of variances, likelihood ratio tests and their applications in testing hypotheses about means, variances, and parameters of the binomial distribution.
- To introduce non-parametric tests and their advantages and limitations.
- To explore non-parametric tests such as the sign test, Wilcoxon signed rank test, and Mann-Whitney test, analysis of variance (ANOVA) including one-way and two-way ANOVA, the Kruskal-Wallis one-way analysis of variance by ranks and the Friedman two-way analysis of variance by ranks.

**Course Contents:**

1. **Correlation and Regression Analysis:** - Introduction and Scatter Diagrams , Karl Pearson’s Coefficient of Correlation, Properties and Problems ,Spearmen’s Rank correlation coefficient. method of Concurrent Deviations, interpretation of r and Probable Error, Linear Regression, Lines of Regression, Theorems on Regression Coefficients, Yule’s Rule, Order of Regression coefficients, Statistical Inferences about the Regression Parameters, Variance of the Residual and the standard error of the estimate, Introduction to Multiple Correlation, Multiple linear Regression.

2. **Chi square distribution:** Introduction of Chi-square Distribution, Chi-square test for Goodness of Fit and its conditions for validity, Chi-Square test for independence of attributes, Degrees of Freedom, test for equality of several Proportions, Chi-square test for population variance, applications of Chi-square Distribution

3. **Small Sample Tests:** Critical Values and Applications of t distribution, Confidence Interval for difference of two means, Paired t-test for difference of two Means, t-test for significance of an observed sample correlation coefficient, Fisher’s Z Transform, F distributions and its applications, F-test for equality of population variances, relation between t, F and Chi-square Distributions.

4. **Likelihood Ratio Tests:** Notion of Likelihood Ratio Test (LRT), construction of LRT for mean of normal distribution (one and two sided when variance is known/unknown), construction of LRT for variance of normal distribution (one and two sided when mean is known/unknown), LRT for parameters of Binomial distribution (two sided) , LRT as a function sufficient statistics, statement of Asymptotic Distribution of -2 log lambda(x).

5. **Non Parametric Tests:** Introduction to Non Parametric tests, Advantages and Limitations, Distribution Free Statistics, Sign Test, Wilcoxon Signed Rank Test, Mann Whitney Test,

6. **Analysis of Variance (ANNOVA):** One-Way analysis of Variance, Two-Way analysis of variance, The Kruskal-Wallis One -Way analysis of variance by ranks, The Friedman Two-Way analysis of variance by ranks.

**Reference Books:**

- Fundamentals of Statistics, by S. G. Gupta.
- Introduction to Probability and Statistics for Engineers and Scientists, by Sheldon M. Ross (Fourth Edition).
- Biostatistics, A Foundation for Analysis in Health Sciences, by Wayne W. Daniel (Eighth Edition, Wiley Publications)
- Mathematical Statistics, by Parimal Mukhopadhyay.

- Statistics for the Life Sciences, by M. Samules, J. Witmer and A. Schaffner (Fifth Edition, Pearson India)
- Probability and Statistics for Engineers, by Richard Gupta, C B Gupta.

## IME-18: Computational Geometry

**Course Description:**

Computational Geometry is a course that focuses on the study of geometric transformations, projections, curves, Bezier curves, B-splines, and various algorithms used in computational geometry. The course explores concepts and techniques used in computer-aided design, curve rendering, conics, curve parametrization, and applications of geometric algorithms in problems such as collision detection, closest pair problem, convex hull, smoothing, line segment intersection, nesting, point location, triangulation, and bounding box.

**Course Objectives:**

- To provide a review of transformations of the plane, including translations, reflections, rotations, shears, and their applications.
- To introduce homogeneous coordinates and their use in projective geometry and transformations. Further, to study projections, including parallel projection and perspective projection, and their types.
- To explore curve rendering techniques and the parametric representation of curves.
- To classify conics and understand their intersections with lines.

- To study Bezier curves of various degrees, including linear, quadratic, cubic, and general Bezier curves, and their properties.
- To introduce rational Bezier curves and their applications. Further, to explore B-splines, their properties, and their types, with applications in font design.
- To understand and analyze algorithms used in computational geometry, including the closest pair problem, collision detection, convex hull algorithms (Graham Scan, Gift Wrapping, Chan's), smoothing algorithms, line segment intersection algorithms, nesting algorithm, point location with respect to a polygon, triangulation, and bounding box algorithms.

**Course Contents:**

1. **Revision:**
   **Transformations of the Plane:** Translations, reflections, rotations, shears, concatenation of transformations, applications, Homogenous coordinates: Homogenous coordinates, points at infinity, projective plane, transformations in homogenous coordinates
   **Transformations of the Space:** Translations, scaling, reflection, rotation about coordinate axes, rotation about an arbitrary line, reflection in an arbitrary plane, applications to Computer-aided Design.

2. **Projections:** Parallel projection and its types, Perspective projection and its types.

3. **Curves:** Curve rendering, parametric Curves, arclength and reparameterization, Classification of Conics, Intersections of a Conic with a Line, parametrization of an irreducible conic, Conics in space, applications of conics.

4. **Bezier Curves:** Bezier curves of low degree, linear Bezier curves, quadratic Bezier curves, cubic Bezier curves, the general Bezier curve, properties of the Bernstein polynomials, properties of Bezier curves, The de Casteljau Algorithm and applications, Rational Bezier Curves and its properties and applications.

5. **B-splines:** Introduction to B-splines, properties of the B-spline Curve and its types, application to Font Design.

6. **Algorithms:** Closest pair problem, Collision detection, Convex hull algorithms (Graham Scan, Gift Wrapping, Chan’s), Smoothing algorithms, Line segment intersection algorithms, Nesting algorithm, Position of a point with respect to polygon, Triangulation, Bounding box algorithm.

**Reference Books:**

- Duncan Marsh, Applied Geometry for Computer Graphics and CAD (Springer, Second Edition)
- de Berg, van Kreveld, Overmars, and Schwarzkopf, Computational Geometry Algorithms and Applications, 2nd Edition, (Springer-Verlag, 2000).

## IME-19: Complex Analysis

**Course Objectives:**

The aim of this course is to study analytic functions, power series, conformal mappings, complex integrations, singularities and the applications of these concepts.

**Course Outcomes:**

- On completion of the course, student will be able to understand
- The basics of complex numbers.
- Analytic functions, power series.
- Conformal mappings and applications.
- The theory of Complex integrations.
- Residue theorem and applications to real integrals.
- The importance of Maximum modulus theorem and its consequences.
- Applications of Complex Analysis

**Course Contents:**

(1) Stereographic projection, Elementary Functions, Exponential function, mapping properties,

logarithmic function, complex exponents, branch of logarithm.

(2) Mobius Transformations, Symmetry and orientation principle, Conformal mappings.

(3) Analytic Functions: Cauchy-Riemann Equations, analyticity, harmonic functions, Power Series.

(4) Complex Integration and Cauchy's Theorem, Cauchy's integral formula, Cauchy's estimate and applications, Homotopic version of Cauchy's theorem, Open mapping theorem, Gourasat's theorem

(5) Singularities- Classification, Laurent series, Residue theorem and applications to evaluation of real integrals, Casorati-Weistrass theorem, Argument principle

(6) Maximum modulus theorem, Schwarz's lemma

(7) Applications of Complex analysis in various fields

**Reference Books:**

- J. B. Conway, Functions of one complex variables, Narosa Publishing House, 1989.
- S. Ponnusamy, H. Silverman, Complex Variables with Applications, Birkhauser, 2006.
- J. Brown and R. Churchill, Complex variables and Applications, 8th Edition, McGraw-Hill, 2009
- M. Spiegel, S. Lipschutz, J. Schiller, D. Spellman, Schaum's Outline of Complex Variables, 2nd edition, 2009.

## IME 20: Operations Research

**Course Description:**

Operations Research is a course that focuses on the application of mathematical modeling and optimization techniques to solve complex decision-making problems. The course covers linear programming (LP), simplex method, sensitivity analysis, duality, transportation and assignment models, network models, advanced linear programming, and integer linear programming. Students will learn how to formulate and solve optimization problems using various techniques and algorithms.

**Course Objectives:**

- To introduce students to the concept of mathematical modeling in operations research.
- To provide an understanding of linear programming models and their graphical solutions.
- To explore the simplex method as an algorithm for solving linear programming problems.
- To introduce sensitivity analysis and its role in evaluating the robustness of solutions.
- To understand duality in linear programming and its economic interpretation.
- To study transportation and assignment models and their solution algorithms.

- To introduce network models and solve problems related to minimal spanning trees, shortest paths, maximal flow, CPM, and PERT.
- To explore advanced topics in linear programming, including the revised simplex method and bounded variable algorithm.
- To introduce integer linear programming and illustrate its applications through case studies.
- To develop problem-solving and analytical skills through real-world applications and optimization algorithms.

**Course Contents:**

(1) **Modeling with Linear Programming:** Two variable LP model, graphical LP solutions, selected LP applications.

(2) **Simplex method and Sensitivity Analysis:** LP model in equation form, transition from graphical to algebraic solution, simplex method, artificial starting solution, special cases in the simplex method, sensitivity analysis.

(3) **Duality and Post-Optimal Analysis:** Definition of the dual problem, primal-dual relationships, economic interpretation of duality, additional simplex algorithms.

(4) **Transportation Model and its Variants:** Definition of the transportation model, non traditional transportation models, transportation algorithm, assignment model.

(5) **Network Model :** Scope and definition of network models, minimal spanning tree algorithm, shortest root problem, maximal flow model, CPM and PERT.

(6) **Advanced Linear Programming:** Simplex method fundamentals, revised simplex method, bounded variable algorithm, duality.

(7) **Integer Linear Programming:** Illustrative applications, integer programming algorithms.

**Important Note:**

- Use suitable mathematical software to solve relevant problems.

**Reference Books:**

- Hamy A.Taha, Operations Research, (Eightth Edition, Prentice Hall of India), 2008.
- J. K. Sharma, Operations Research, (Third Edition, Macmillan India Ltd.), 2008.
- P. K. Gupta and D. S. Haria, Operations Research, (Fifth Edition, S. Chand), 2014.

## IME 21: Coding Theory

**Course Description:**

Coding Theory is a course that focuses on the study of error detection, error correction, and decoding in communication channels. The course covers topics such as maximum likelihood decoding, Hamming distance, nearest neighbor/minimum distance decoding, linear codes, vector spaces over finite fields, Hamming weight, bases of linear codes, generator matrix and parity check matrix, equivalence of linear codes, encoding with a linear code, decoding of linear codes, cossets, syndrome decoding, cyclic codes, generator polynomials, generator and parity check matrices, decoding of cyclic codes, burst-error-correcting codes, BCH codes, and parameters of BCH codes. The course provides a solid foundation in coding theory and its applications in error detection and correction in various communication systems.

**Course Objectives:**

- To introduce the concepts of error detection, error correction, and decoding in communication channels.

- To study the concept of Hamming distance and its role in error detection and correction, explore nearest neighbor/minimum distance decoding and its application in error correction. Further, to understand linear codes as vector spaces over finite fields, Hamming weight and its significance in linear codes.
- To study bases of linear codes, including generator matrix and parity check matrix representations.
- To explore encoding with a linear code and the process of transmitting information using linear codes. Further, to study the decoding techniques for linear codes, including cossets and nearest neighbor decoding.
- To understand syndrome decoding and its application in error correction for linear codes.
- To introduce cyclic codes and their properties, including generator polynomials and generator/parity check matrices.
- To study the decoding process for cyclic codes and their application in burst-error correction.
- To explore special cyclic codes, specifically BCH codes, and their parameters.

**Course Contents:**

1) **Error detection:** correction and decoding: Communication channels, Maximum likelihood decoding, Hamming distance, Nearest neighbor / minimum distance decoding, Distance of a code.
2) **Linear codes:** Vector spaces over finite fields, Linear codes, Hamming weight, Bases of linear codes, Generator matrix and parity check matrix, Equivalence of linear codes, Encoding with a linear code, Decoding of linear codes, Cossets, Nearest neighbor decoding for linear codes, Syndrome decoding.
3) **Cyclic codes:** Definitions, Generator polynomials, Generator and parity check matrices, Decoding of cyclic codes, Burst-error-correcting codes.
4) **Some special cyclic codes:** BCH codes, Definitions, Parameters of BCH code

**Reference Books:**

- San Ling and Chaoing xing, Coding Theory- A First Course
- Raymod Hill, A First Course in Coding Theory (Oxford)
- Lid and Pilz, Applied Abstract Algebra Second Edition

## IME-22: Rings and Fields

**Course Description:**

The course Rings and Fields provides students with a comprehensive understanding of fundamentals of Rings and Fields. It comprises of the study of basic properties of Rings, Integral Domains, ED, PID and UFD. Further it introduces Finite fields and their applications to Geometry. It also explains the insolvability of quintic polynomials using algebra.

**Course Objectives:**

- To understand the basics of Rings and Fields.
- To know Field Extensions and their degree over subfield.
- To learn the relation of field extensions and polynomials
- To understand Ruler and Compass Construction and its application to solve

- celebrated problems in Geometry.
- To explore Finite fields and Galois Theory
- To study Solvability of a polynomial by radicals.

**Course Contents:**

1) **Rings:** Definition and examples, ideals, ring homomorphisms, fundamental theorem of ring homomorphisms, integral domains, fields, polynomial ring, Euclidean domains and division algorithm for polynomials, unique factorization in the ring of integers and polynomial ring K[x], criterion for irreducible polynomials, Eisenstein criterion
2) 
3) **Fields:** Examples of finite and infinite fields, field extensions, degree of extension, construction of finite fields, primitive elements, cyclotomic polynomials, irreducible Polynomials over finite fields, Factorization of polynomials over finite fields, Chinese Remainder theorem for polynomials, Berlekamps algorithm. Application: Any one application to either Coding theory or RSA cryptosystem.

**References Books:**

- Rudolf Lidl and Gunter Pilz, Applied Abstract Algebra (Springer, Second Edition)
- G. Mullen and C. Mummert, Finite Fields and Applications (AMS, Indian Edition)
- John M. Howie, Fields and Galois Theory, (Springer)

## IME-23: Applied Linear Algebra

**Course Description:**

The course on Applied Linear Algebra is aimed at studying the applications of Linear Algebra. The course introduces students to basic tools and techniques required for applications such as Method of Least Squares and the Discrete Fourier Transform. An important application is Singular Value Decomposition / Principal Component Analysis. Further, the course discusses some of the computational issues in numerical linear algebra such as condition number of a matrix and computation of eigenvalues.

**Course Objectives:**

- To understand the role and significance of Linear Algebra in practice
- To learn the tools and techniques of Linear Algebra with a view towards applications
- To learn some concrete applications of Linear Algebra such as PCA/SVD
- To understand and appreciate the difficulties of computations in Linear Algebra.

**Course Contents:**

1) **Basics :** Projections and orthogonalization, Least squares method, the Fast Fourier transform, applications of determinants, diagonalization and powers of matrices, differential equations and matrix exponential

2) **Positive Definite Matrices :** Minima, maxima and saddle points, tests for positive definiteness, semidefinite and indefinite matrices, Minimum principles and Rayleigh Quotient, Finite element method

3) **Computations with Matrices :** Norm and condition number of a matrix, computation of eigenvalues, iterative methods for Ax=b

4) **Decomposition methods :** Singular Value Decomposition(SVD), Applications, Pseudoinverse using SVD

**References Books:**

- Gilbert Strang, Linear Algebra and its Applications, (Brooks/Cole, Third Edition)
- Peter Lax, Linear Algebra ( John Wiley & Sons Inc)
- Gilbert Strang, Introduction to Applied Mathematics ( Wellesley Cambridge Press)

## IME-24: Financial Mathematics

**Course Description:**

Financial Mathematics is a course that focuses on the study of options and markets, option valuation models, and various numerical methods used in financial mathematics. The course covers topics such as the introduction to options and markets, Black-Scholes model, option values, payoffs and strategies, put-call parity, the Black-Scholes equation, exact formulas for European options, American options, binomial methods, Monte Carlo simulation, finite difference methods, and a lab component involving the implementation of option pricing algorithms and evaluations for Indian companies. The course provides a comprehensive understanding of option pricing and the application of numerical methods in financial mathematics.

**Course Objectives:**

- To introduce the types of options and their role in financial markets.
- To study the Black-Scholes model and its application in option valuation.

- To analyze option values, payoffs, strategies in different market scenarios, put-call parity and its significance in option pricing.
- To understand the Black-Scholes equation and its solution for European options, American options and the free boundary problem associated with them.
- To analyze option valuation using binomial methods, considering dividend-paying stocks and general formulations.
- To explore option valuation through Monte Carlo simulation.
- To study finite difference methods, including explicit and implicit methods, for option valuation.

**Course Contents:**

1) **Introduction to options and markets:** types of options, interest rates and present values.
2) **Black Sholes model:** arbitrage, option values, pay offs and strategies, put call parity, Black Scholes equation, similarity solution and exact formulae for European options, American option, call and put options, free boundary problem.
3) **Binomial methods:** option valuation, dividend paying stock, general formulation and implementation.
4) **Monte Carlo simulation:** valuation by simulation
5) **Finite difference methods:** explicit and implicit methods with stability and conversions analysis methods for American options- constrained matrix problem, projected SOR, time stepping algorithms with convergence and numerical examples.
6) **Lab component:** implementation of the option pricing algorithms and evaluations for Indian companies.

**Reference Books:**

- D.G.Luenberger, Investment Science, Oxford University Press,1998.
- J.C.Hull , Options, Futures and Other Derivatives, 4th ed., Prentice- Hall ,New York,2000.
- J.C.Cox and M.Rubinstein, Option Market, Englewood Cliffs,N.J.: Prentice- Hall,1985.
- C.P. Jones. Investments, Analysis and Measurement, 5th ed.,John Wiley and Sons,1996.

## IME-25: Integral Transforms

**Course Description:**

The aim of this course is to describe the ideas of Fourier and Laplace Transforms and indicate their applications in the fields such as the application of PDE, Digital Signal Processing, Image Processing, Theory of wave equations, Differential Equations and many others. To use the Fourier transform for solving boundary value problems appearing in scientific & engineering problems. Moreover, this course also aims to study different types of integral transforms such as Laplace transforms and Wavelet Transform along with its applications.

**Course Objectives:**

- Understand different kinds of integral transforms.
- Understand Fourier transform and its properties and will be able to solve the examples based on it.

- Have a deep understanding of Laplace Transformation and its real-life application.
- Solve initial value problem and boundary value problem using Laplace Transform.
- Derive Fourier series representation of Periodic functions
- Have an understanding about wavelet transform and its applications.

**Course Contents:**

1) **Laplace Transform:** Properties of Laplace Transform, Laplace Transform of the derivatives of a function, Inverse Laplace transform, Properties of inverse Laplace transform, Inverse Laplace transform of derivatives, Convolution theorem, Heavisides expansion theorem. Application of Laplace Transform, Solution of ODEs and PDEs.
2) **Fourier Transforms:** Fourier integral theorem, Fourier transform Pairs, Properties of Fourier transform, Fourier cosine transform, Inverse Fourier Transform, Inverse Fourier sine Transform, Inverse Fourier cosine Transform, Properties of Fourier Transforms, Modulation theorem, Convolution theorem, Fourier Transform of the derivatives of functions, Parsevals identity, Application of Fourier Transforms to the solution of initial & boundary value problems.
3) **Finite Fourier Transform:** Finite Fourier sine and cosine transform, inversion formula for sine and cosine transforms, multiple finite Fourier transforms, operational properties of finite Fourier sine and cosine transforms, convolution theorem, applications.
4) **Wavelet Transform:** Continuous Wavelet Transform, Time-Frequency Space Analysis, Short-Time Fourier Transform, Wigner Distribution and Ambiguity Functions, Properties of the Wavelets, Admissible Condition, Regularity, Multiresolution Wavelet Analysis, Linear Transform Property, Examples of the Wavelets, Discrete Wavelet Transforms, Time-Scale Space Lattices, Wavelet Frame.

**References Books:**

- Larry C. Andrews and Bhimsen K. Shivamoggi, Integral Transforms for Engineers, Prentice Hall, 2003.
- N. Srivastava and Mohammad Ahmad, Integral Transforms and Fourier Series, Narosa Publishing House, 2015.

## IME-26: Differential Geometry

**Course Description:**

Differential Geometry is an advanced course that explores the mathematical study of curves and surfaces in multidimensional spaces. This course provides a rigorous introduction to the fundamental concepts and techniques of differential geometry, focusing on the geometry of curves and surfaces. It covers topics such as vector fields, tangent spaces, orientation, curvature, geodesics, and surface area. Additionally, it introduces important tools and methods for understanding the intrinsic geometry of curves and surfaces, including the Gauss map, Weingarten map, and exponential map. Applications of differential geometry to various fields, such as physics and computer graphics, will also be discussed.

**Course Objectives:**

- Understand the basic concepts and principles of differential geometry, specifically in the context of curves and surfaces.

- Develop proficiency in working with vector fields and tangent spaces, and their applications in differential geometry.
- Analyze the geometry of surfaces, including properties such as orientation, curvature, and surface area.
- Study the behavior of curves on surfaces, including geodesics and parallel transport.
- Explore the Gauss map and its role in understanding the local and global properties of surfaces, Weingarten map and its relationship to curvature, providing insights into the shape of surfaces.
- Apply the concepts of differential geometry to calculate arc length, line integrals, surface area, and volume of parametrized surfaces.
- Utilize the exponential map to understand the behavior of curves and surfaces in relation to their local geometry.
- Examine surfaces with boundaries and explore their unique geometric properties.

**Course Contents:**

Graphs and level sets, vector fields, tangent spaces, surfaces, vector fields on surfaces, orientation, gauss map, geodesics, parallel transport,Weingarten map, curvature, arc length and line integrals, curvature of surfaces, parametrized surfaces, surface area and volume, exponential map, surfaces with boundary.

**Reference Books:**

- John A. Thorpe, Elementary topics in differential Geometry, Springer, (2004)
- B Oneill : Elementary differential Geometry, (Academic - New York)

## IME-27: Numerical Analysis

**Course Description:**

The course "Numerical Analysis" focuses on various numerical methods used for solving mathematical problems and approximating solutions. It covers topics such as solving algebraic and transcendental equations, interpolation, numerical differentiation and integration, numerical linear algebra, and numerical solutions of ordinary differential equations.

**Course Objectives:**

- To understand and apply different numerical methods for solving algebraic and transcendental equations.
- To learn interpolation techniques for approximating unknown values within a set of given data points.

- To explore numerical differentiation and integration methods for approximating derivatives and integrals.
- To study numerical techniques for solving systems of linear equations and understand their computational aspects.
- To learn numerical methods for approximating solutions to ordinary differential equations.

**Course Contents:**

1) **Iterative solutions of nonlinear equation:** bisection method. Fixed-point iteration, Newton's method, secant method, acceleration of convergence, Newton's method for two nonlinear equations, polynomial equation methods.

2) **Polynomial interpolation:** interpolation polynomial, divided difference interpolation, Aitken's formula, finite difference formulas, Hermite's interpolation, double interpolation.

3) **Linear systems of Equations:** Gauss Elimination, Gauss-Jordan method, LU decomposition, iterative methods, and Gauss- Seidel iteration.

4) **Numerical Calculus:** Numerical differentiation, Errors in numerical differentiation, Numerical Integration, Trapezoidal rule, Simpson's 1/3 - rule, Simpson's 3/8 rule, error estimates for Trapezoidal rule and Simpson's rule.

5) **Numerical Solution of Ordinary differential Equations :** Solution by Taylor series, Picard Method of successive approximations, Euler's Method, Modified Euler Method, Runge-Kutta Methods, Predicator-Corrector Methods.

6) **Eigenvalue Problem:** Power method, Jacobi method, Householder method.

**NB :** Students are also required to use suitable programming language to solve relevant problems.

**Reference Books:**

- M. K. Jain, S. R. K. Iyengar and R. K. Jain, Numerical methods for scientific and engineering computations, (Sixth Edition, New Age International Publishers), 2015.
- S. S. Sastry, Introduction Methods of Numerical Analysis (4th Edition) (Prentice-Hall).
- K .E. Atkinson: An Introduction to Numerical Analysis.
- J. I. Buchaman and P. R. Turner, Numerical Methods and Analysis

## IME-28: Partial Differential Equations

**Course Objectives:**

The aim of this course is to study the concept of different types of partial differential equations (PDEs) such as first-order differential equations, second-order differential equations, Laplace equations, and heat equations. and wave equations and solve them using well-known methods such as the method of characteristics and separation of the variable method.

**Course Outcomes:**

On completion of the course, students will be able to

- Understand the basics of partial differential equations.

- Apply the method of characteristics to solve types of PDEs.
- Apply the method of variables separation.
- Solve particular examples of PDEs.

**Course Contents:**

1) **First and second order linear equations:** terminologies, superposition principle, linear dependence, First order linear equations, initial value problem, classification of second order equations, well posedness

2) **Heat equation:** Derivation of heat equation, initial boundary value problems, homogeneous boundary conditions, non-homogeneous boundary conditions, Robin boundary conditions, Infinite domain problems, maximum principle, energy method, uniqueness of solutions.

3) **Wave equation:** Derivation of wave equation, Initial value problems, wave reflection problems, Initial boundary value problems, Energy method

4) **Laplace equation:** boundary value problems, separation of variables, Fundamental solution, Greens identity, Greens function, Properties of harmonic function, Well posedness issues

5) **First order quasilinear equations**, scalar conservation law, Rankine-Hugoniot condition, weak solutions, entropy condition, traffic flow problem, First order nonlinear equations, systems of first order equations

6) **Fourier series and Eigenvalue problems:** Fourier convergence theorems, Derivations of Fourier series, Sturm-Liouville Problems.

**Reference Books:**

- H. Hattori, Partial Differential Equations, Methods, Applications and Theories, World Scientific publications, 2014.
- T. Amaranath, An Elementary Course in Partial Differential Equations, Narosa, 2003.
- J. Brown and R.V. Churchill, Fourier Series and Boundary Value Problems, (McGraw-Hill)

## IME-29: Cryptography

**Course Description:**

Cryptography is a course that focuses on the study of cryptographic techniques and systems. The course covers topics such as divisibility and the Euclidean algorithm, congruences, factorizations, finite fields, quadratic residues, simple cryptosystems, integer factorization, discrete logarithm, public key cryptography, hash functions, RSA (Rivest-Shamir-Adleman), Diffie-Hellman key exchange system, the ElGamal cryptosystem, digital signatures, primality and factoring, primality tests, pseudo primes, Miller-Rabin primality test, elliptic curve cryptography, and elliptic curve cryptosystems. The course provides a comprehensive understanding of the mathematical foundations of cryptography and various cryptographic algorithms.

**Course Objectives:**

- To introduce the concept of divisibility and the Euclidean algorithm,
- congruences and their applications in cryptographic algorithms.
- To study factorizations of numbers and their relevance to cryptographic systems. Further, to explore finite fields and quadratic residues in the context of cryptography.
- To study integer factorization and the challenges associated with breaking cryptographic systems.
- To understand the discrete logarithm problem and its significance in cryptography.
- To analyze the RSA algorithm, which is widely used in public key cryptography.
- To understand the Diffie-Hellman key exchange system and its role in secure communication.
- To explore the ElGamal cryptosystem and its mathematical foundations.
- To study digital signatures and their applications in verifying the authenticity of digital documents.
- To understand primality and factoring of numbers and their relevance to cryptography.
- To explore primality tests, including pseudo primes and the Miller-Rabin primality test.

**Course Contents:**

1) **Introduction to cryptography:** Cryptography in Modern world. Substitution cipher, Ceaser cipher as a special case of substitution cipher, Monoalphabetic ciphers, Transposition Cipher, Polyalphabetic substitution ciphers, Vigenere Cipher, Introduction to polygraphic substitution ciphers, cryptanalysis of substitution cipher

2) **Symmetric key cryptography:** Introduction and overview, Stream Cipher, one-time Pad, Block cipher, Modes of operation Electronic code book, cipher block chaining, Cipher feedback, Algorithms: Data Encryption Standard, Advanced Encryption Standard, IDEA (International Data Encryption Algorithm), Attacks against DES, AES, IDEA

3) **Public key Cryptography:** Introduction and Overview, The RSA algorithm, Generation of keys, Exchange messages, Diffie Hellman Key Agreement protocol, EIGamal Encryption, Algorithms: Discrete Logarithm, MD5, Attacks against RSA, Discrete Logarithm.

4) **Hashing:** Motivation and applications, cryptographically secure hashing, message authentication codes (MAC), HMAC, Network security, Secure Socket layer (SSL), Definition of secrete sharing, visual secret schemes, Shamirs sharing scheme. Applications of Cryptography: - Digital Signature, Kerberos, Pretty Good privacy Internet protocol security Note

5) **Hands on (Optional) :** Applications of cryptography such as: Digital Signature, Kerberos, Pretty Good privacy, Internet protocol security. All the topics based on Ciphers can be implemented using C, C++ or python as Programming Exercises or Assignments.

**Reference Books:**

- Neal Koblitz, A Course in Number Theory and Cryptography (Springer, Second Edition)
- Robert Edward Lewand: Cryptological Mathematics (Mathematical Association of America).

- D. R. Stinson: CRYPTOGRAPHY, Theory and practice, CRC Press, 1995
- Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman: An introduction to Mathematical Cryptography, Springer
- Adam J. Elbirt: (CRC press): Understanding and Applying cryptography and Data security.
- Bruice Schneier: Applied Cryptography (Wiley India Edition)
- Atul Kahate: Cryptography and Network security (Tata McGraw Hill)

## IME-30: Theory of Computer Science

**Course Description:**

Theory of Computer Science is a course that focuses on the theoretical foundations of computer science. The course covers various topics such as formal languages, automata theory, context-free languages, pushdown automata, Turing machines, recursive and recursively enumerable languages, and undecidable problems. The course introduces fundamental concepts and techniques in the theory of computation and provides students with a solid understanding of the theoretical aspects of computer science.

**Course Objectives:**

- To introduce the basic concepts and terminology of formal languages and automata.
- To study regular languages, regular expressions, and finite automata.

- To understand context-free languages, context-free grammars, and pushdown automata, the relationships between different models of computation, such as finite automata, regular grammars, and pushdown automata.
- To introduce Turing machines and their computational power.
- To study the concepts of recursive and recursively enumerable languages.
- To understand the limitations of computation and the concept of undecidability.
- To analyze the halting problem and other undecidable problems using Turing machines, RAM model of computation.
- To solve simple arithmetic problems using Turing machines.

**Course Contents:**

1) Preliminaries: Symbol, Alphabet, String, Prefix and Suffix of Strings, Sets, Operations on sets, Finite and infinite sets, Russells Paradox, Formal Language Relation, Equivalence Relation,(reflexive, transitive and symmetric closures) Principle of Induction

2) Regular Languages: Regular Expression: Definition, Examples, and Identities Finite Automata: Concept DFA: Definition and examples NFA: Definition, examples, Language accepted By FA, NFA with e- moves, Regular Expression to FA: Method and Problems NFA with e- moves to NFA, NFA to DFA: Method Problems: Minimization of DFA: Problem using Table Method, Subset Construction for NFA with e-moves to DFA conversion, Application of FA: Pumping Lemma and Examples Closure Properties: Union, Intersection, -Concatenation, Complement, and Kleene Closure

3) Context Free Languages -Chomsky Hierarchy -CFG: Definition and examples, Ambiguous Grammar: Concept and Examples Simplification of CFG: Removing Useless Symbols, removing unit productions and removing Nullable symbols: Methods and Problems, Normal Forms: CNF and GNF: Method and Problems - Regular Grammar: Definition, Equivalence of FA and Regular Grammar

4) Push Down Automaton: Basic Concept, Definition (DPDA and PDA) Construction of PDA using empty stack and final State method: Examples using stack method Equivalence between acceptance by final state and Empty stack method and examples Equivalence between PDA and CFG (in GNF): Method and examples Properties of Context-Free Languages Pumping Lemma for CFL: methods and problems, Closure Properties of CFLs(Union, Concatenation, and Kleene Closure: Method and Examples)

5) Turing Machine -Recursive and recursively enumerable language -Introduction to LBA (Basic Model) and CSG. -Definition of TM, -Design of TM for language recognition, Types of Turing Machine (Multitape TM, Non-Deterministic TM, Universal TM, Restricted TM) Undecidable Problem, Halting Problem of TM Simple Arithmetic Problems on Unary Numbers using TM, RAM model of computation.

   Important Note: The LEX tool on Linux is to be used to address the understanding of the Language and grammar aspects in this course. Few laboratory sessions are expected to be covered. Some marks are to be reserved in Continuous Evaluation/Assessment for the laboratory assignments/work.

**References Books:**

- Introduction to Automata Theory, Languages, and Computation (2ndEdition Pearson Education) By John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman
- An Introduction to Formal Languages and Automata, Peter Linz, Jones & Barlett Student Edition, ISBN: 9789380853284
- Fundamentals of Theory of Computation, Principals and Practice, Greenlaw, Hoover, Elsevier, ISBN:9781558604742
- Introduction to Computer Theory By - Daniel I.A. Cohen (JohnWiley & Sons (ASIA) Pvt Ltd. 2ndEdition)
- An Introduction to the Theory of Computer Science Languages & Machine (3rdEdition Pearson education) By Thomas A. Sudkamp
- Introduction to Languages and the theory of Computation By John C.Martin (Tata Mc-Graw Hill Edition,2ndEdition)
- Theory of Computer Science (Automata Languages And Computation By K.L.P. Mishra & N. Chandrasekaran (Prentice Hall India 2nd Edition)

| | | | |
|---|---|---|---|
| IME-31 | Emerging Technologies | 4 | 2T + 2P |
| IME-32 | Mooc/NPTEL/Swayam/Any other courses approved by the Departmental Committee of the Department | 4 | 2T + 2P |

The departmental committee will decide on the syllabus for the following subjects as per the need and demands of the current industry/research institutes' requirements.