INDIAN STATISTICAL INSTITUTE

Students' Brochure

PART II

Master of Technology (M.Tech.) in Cryptology and Security

(Effective from Academic Year 2026-27)

(See PART I for general information, rules and regulations applicable to all programmes)

The Headquarters
203 BARRACKPORE TRUNK ROAD
KOLKATA 700108

INDIAN STATISTICAL INSTITUTE

Master of Technology (M.Tech.) in Cryptology and Security

Contents

The structure of the M.Tech. in Cryptology and Security (MTCrS) programme differs from that of the other degree programmes mentioned in Part I, which lays out general information, rules and regulations applicable to all programmes. Some information/rules specific to this programme supersede the corresponding information/rules in the general brochure; see Section 3 for details.

1 CS and Non-CS Streams

Every student who is admitted to the programme is categorised as either a **CS-stream** student or a **Non-CS-stream** student. This categorisation is based on the general structure of the common admission process that is followed for the M.Tech. Computer Science (MTCS) and MTCrS programmes.[1] Typically, students qualify based on

- **EITHER** their performance in two written tests conducted by the institute,

  OR a valid GATE score;
- **AND** their performance in a subsequent interview.

One or more of the written tests generally have separate sections, comprising questions on (i) Mathematics and (ii) Computer Science.

To be categorised as a **CS-stream** student, the candidate must be a graduate in Computer Science / Information Technology / any other discipline regarded as equivalent by the admission committee[2], **AND** must have qualified based on

- **EITHER** their answers to questions from the Computer Science section(s) of the written test(s);
- **OR** a valid GATE score (above a threshold as decided by the admission committee) in Computer Science / Information Technology / any other discipline regarded as equivalent by the admission committee.

All other candidates will be categorised as **Non-CS-stream** students.

The courses in this programme are categorised as **compulsory**, **formative** and **elective**. The curricula for the two streams differ in terms of the formative courses that a student has to pass. Students are required to follow the curriculum specified for their respective stream.

[1] If the admission process followed in a particular year has a significantly different structure, the process for assigning streams to incoming candidates will be specified in the Admission Prospectus for that year.

[2] Likewise, in case of any dispute / lack of clarity about whether a candidate satisfies this eligibility criterion, the admission committee will take the final decision, using the relevant year's Admission Prospectus as a guideline.

2 Structure of the Programme

2.1 Structure

A student has to pass the courses listed below. The courses are categorized as: Compulsory, Formative and Elective. The details are as follows.

1. **Compulsory non-credit course:**

   - Introduction to Programming

   This requirement may be waived if a student passes a programming test designed by the Mentor Committee, and conducted within the first two weeks of the first semester.

2. **Seven compulsory courses** :

   i. Discrete Mathematics
   ii. Data Structures and Algorithms
   iii. Design and Analysis of Algorithms
   iv. Computing Laboratory
   v. Cryptology I
   vi. Cryptology II
   vii. Computing Systems Security

3. **Four Formative Courses** from the list of formative courses given below:

   Pool A

   i. Digital Design and Computer Architecture
   ii. Computer Networks
   iii. Database Management Systems
   iv. Operating Systems

   Pool B

   i. Abstract Algebra and Number Theory
   ii. Linear Algebra
   iii. Probability and Stochastic Processes
   iv. Statistical Methods
   v. Theory of Computation

   - A **non-CS** stream student must take **at least 3** from Pool A.
   - A **CS** stream student must take **at least 3** from Pool B.

The following restrictions are applicable for choosing the formative courses.

- **For students with a master's degree in Mathematics:**
  Linear Algebra, and Abstract Algebra and Number Theory cannot be taken.
- **For students with a master's degree in Statistics:**
  Probability and Stochastic Processes, and Statistical Methods cannot be taken.
- Students are discouraged from taking formative courses that they have successfully completed in their previous programmes. The final selection of formative courses should be done in consultation with the mentor committee.

4. **Six elective courses:** There are two types of elective courses: **CrS Elective** courses and **Free Elective** courses.

   (a) **CrS Elective:** A student must take at least four courses from the list given below:

   i. Blockchains and Cryptocurrencies
   ii. Coding and Information Theory
   iii. Computational Complexity
   iv. Computational Number Theory
   v. Computing Systems Security II
   vi. Cryptographic and Security Implementation
   vii. Hardware Security
   viii. Math Toolkits for CS
   ix. Multiparty Computations
   x. Topics in Cryptology
   xi. Topics in Privacy
   xii. Quantum Computations and Cryptography

   (b) **Free Elective:** A student can take the remaining number of electives from the list of:

   i. CrS Electives as listed in 4(a) of Section 2.1,
   ii. Formative Courses (the restrictions noted for formative course would be applicable here also) as listed in 3 of Section 2.1, and
   iii. CS Electives (the list of the courses and their corresponding syllabi can be found in the MTCS Brochure).

5. **Dissertation**: A semester long dissertation, whose details are in Section 3.4.

2.2 Suggested ideal Semester-wise layout

One who Passes the Programming Test		
	CS	non-CS
Semester 1	1. Data Structures and Algorithms 2. Discrete Mathematics 3. Cryptology I 4. Abstract Algebra and Number Theory 5. Probability and Stochastic Processes	1. Data Structures and Algorithms 2. Discrete Mathematics 3. Cryptology I 4. Digital Design and Computer Architecture 5. Statistical Methods
Semester 2	1. Design and Analysis of Algorithms 2. Cryptology II 3. Computing Laboratory 4. Theory of Computation 5. Operating Systems	1. Design and Analysis of Algorithms 2. Cryptology II 3. Computing Laboratory 4. Computer Networks 5. Operating Systems
Semester 3	1. Computing Systems Security I 2. Electives (4)	1. Computing Systems Security I 2. Electives(4)
Semester 4	1. Electives (2) 2. Dissertation	1. Electives (2) 2. Dissertation

One who Fails the Programming Test		
	CS	non-CS
Semester 1	1. Introduction to Programming 2. Data Structures and Algorithms 3. Discrete Mathematics 4. Cryptology I 5. Abstract Algebra and Number Theory 6. Probability and Stochastic Processes	1. Introduction to Programming 2. Data Structures and Algorithms 3. Discrete Mathematics 4. Cryptology I 5. Digital Design and Computer Architecture 6. Statistical Methods
Semester 2	1. Design and Analysis of Algorithms 2. Cryptology II 3. Computing Laboratory 4. Theory of Computation 5. Operating Systems	1. Design and Analysis of Algorithms 2. Cryptology II 3. Computing Laboratory 4. Computer Networks 5. Operating Systems
Semester 3	1. Computing Systems Security I 2. Electives (4)	1. Computing Systems Security I 2. Electives(4)
Semester 4	1. Electives (2) 2. Dissertation	1. Electives (2) 2. Dissertation

3 Information/rules specific to the M.Tech. CrS programme

The information given in this section supplements, and occasionally supersedes (but is never superseded by) the information given in the general brochure for all non-JRF degree programmes. In particular, the notions of promotion and repeating a year, and the corresponding sections of the general brochure, do not apply to the M Tech CrS programme.

3.1 Duration of the programme

The expected time for completion of the programme is two years. A student may take upto a maximum of three years for completion. However, after completion of the second year, a student will not be eligible for stipends, contingency grants or hostel facilities. In special circumstances, e.g., medical emergencies, exceptions to this rule may be made by the Dean, in consultation with the Teacher's Committee/Mentor Committee, on a case by case basis.

3.2 Waiver of class attendance

For a formative course, a student can bypass regular classes and claim credit by completing only the assignments and passing the examination(s). This may give the student the flexibility to attend an elective during that time, and thus complete the course requirements early.

- If a student opts for waiver of class attendance, (s)he is required to seek permission from the Mentor Committee.
- The teacher of the course and the Dean's Office need to be informed before the mid-semester week.
- The usual attendance requirement for the student in such cases would be completely waived for the specific course.
- Under this option, the student has to obtain at least 60% to pass.

3.3 Registering for a course

Within 2 weeks of the start of a semester, a student will have to inform the Dean's Office through the Mentor Committee about the courses that (s)he is registering for. A student cannot opt out of a course after these 2 weeks. The scores obtained in all the courses thus registered will be recorded in the marksheet. Even after passing a course, a student may, with the approval of the Mentor Committee, opt to take the course again in subsequent semester(s). In such cases, the highest score obtained in the course across all attempts will be considered when computing the student's final result.

3.4 Dissertation

A student is required to work for a dissertation on a topic assigned/approved by the CrS Dissertation Committee under a supervisor.

The work for a dissertation should be substantial and related to some important problem in an area of Cryptology and/or Information Security. It should have notable theoretical and/or practical significance. A critical review of recent advances in an area of Cryptology and/or Information Security, with some contributions by the student, is also acceptable as a dissertation.

Duration: The work for the dissertation should cover one semester, which will typically be the fourth semester. A student who opts for finishing the degree programme in more than two years but within three years, as per guidelines in the brochure, can commence her/his dissertation at the beginning of the fifth or the sixth semester. The semester in which a student starts his/her dissertation will be called as the dissertation semester.

The assignment of dissertation topics and the evaluation of students' work will be coordinated by a CrS Dissertation Committee that is to be formed by the Dean's Office in consultation with the Mentor Committee before the students of that batch are promoted to their second year.

Supervisor: The dissertation shall be carried out under the guidance of a supervisor. Typically, the supervisor will be a full-time faculty member of ISI; however, joint supervision may also be permitted. In cases of joint supervision, at least one of the supervisors must be an ISI faculty member.

In case of joint supervision, a student may choose to have a primary supervisor from outside the Institute. In such cases, provided the student has completed all course requirements, the student may be permitted to spend the dissertation semester at the institution of the primary supervisor. A student may also have a primary supervisor external to ISI while conducting the dissertation work at ISI.

The primary supervisor must hold at least a PhD degree and be affiliated with a reputed academic institution or industry. All cases of joint supervision must be approved by the

CrS Dissertation Committee within the first four weeks of the dissertation semester. The selection of the supervisor for the dissertation of each student should be done within the first four weeks of the dissertation semester.

Evaluation: An interim progress report of the dissertation work should be submitted at the middle of the semester in which it was commenced (typically by the first week of April in the fourth semester). This report will be evaluated by a committee formed by the CrS Dissertation Committee. The committee will consist of relevant ISI faculty and the supervisor(s).

- The student's interim report would be judged either satisfactory or unsatisfactory.
- If the report is found to be unsatisfactory, the CrS Dissertation Committee will, in consultation with the supervisor(s), allow the student extra time (not exceeding a month) to make up. If the progress of the student is deemed to be satisfactory at this second evaluation, (s)he will be allowed to continue with the dissertation. Otherwise, the student will have to start the dissertation afresh in the beginning of the following semester (typically the fifth semester), possibly with a different supervisor.

There will be no back paper for the dissertation.

The final dissertation should be submitted at the end of the semester in which it was commenced as indicated in the academic calendar (typically by the middle of **June** of the standard year of completion of the batch). The dissertation will be evaluated out of **300 marks** by a committee formed by the CrS Dissertation Committee. The committee will consist of relevant ISI faculty, the supervisor(s) and one or more external expert(s). The student has to defend his/her dissertation in an open seminar.

If the final dissertation is not submitted within the stipulated deadline, or if the evaluation committee, as mentioned earlier, evaluates the student to have failed in the dissertation, then the dissertation will be evaluated at the end of the next semester (typically the fifth semester). Thus, if a student's interim progress report is regarded as unsatisfactory, or if s/he fails the final dissertation evaluation, her/his dissertation will typically be re-evaluated at the end of the fifth semester.

If a student undertakes her/his dissertation during the fifth or sixth semesters, and fails in the interim or the final evaluation organized by the CrS Dissertation Committee, then the Dean and the Mentor Committee will decide on a case-by-case basis whether the student is not to be awarded the degree at all, or to be given one last chance to finish the dissertation if the student has strong reasons for the failure. In these exceptional cases, while continuing to work on the dissertation for another semester, the student will be enrolled as an ISI student (full-time or part-time) but will not be eligible for stipend and hostel facilities. The allowable exceptions are only for valid medical grounds, to be decided upon by the Dean, in consultation with the concerned mentor committee.

3.5 Internship/industrial training

There would be a mandatory 12 weeks gap between the first and the second year in the academic calendar. During this period, students may pursue internship/industrial training outside the institute in research institutes or public/private sector organizations in India or abroad. However, internship is not mandatory.

A student who undergoes internship/industrial training somewhere in India during the training period may receive his/her usual monthly stipend/remuneration/emoluments either from the Institute (ISI) or from the host organization at his/her own discretion with appropriate permission from the Dean. The students who are placed outside Kolkata for training will be reimbursed to and fro train fare from Kolkata to the place of internship as per ISI rules unless the host organization provides travel assistance.

Training may also be arranged at a research/R&D organization abroad. In case a student undergoes practical training at such a place abroad, the Institute (ISI) will not provide any financial support including the monthly stipend for that period.

3.6 Final Result

Upon passing the minimum number of courses and dissertation, the final percentage obtained by a student is to be computed out of 2000 marks, and is the sum of (a) the best seventeen scores obtained in the courses passed as per requirements[3], and (b) the dissertation marks.

[3] A student is required to pass 7 compulsory courses, 4 formative courses, and 6 elective courses (of which 4 are required to be CrS electives). However, (s)he may, with permission from the Mentor Committee, repeat one or more of the compulsory courses, take more than 4 formative courses, or more than 6 elective courses. In such cases, the final score will be computed using the best score obtained in each compulsory course, the best 4 scores obtained in the formative courses, and the best 6 scores obtained in the elective courses (this must include 4 CrS electives).

3.7 Stipend

On admission, each student will receive the institute specified stipend, subject to the rules described in the general brochure for all non-JRF degree courses, with the following modification in respect of 'Performance in coursework' criterion. A student is eligible for a full stipend

- in the first semester, only if (s)he registers for at least four courses;
- in the second semester, only if (s)he
  - registers for a total of at least nine courses in the first two semesters, and
  - obtains at least 45% in each of at least four courses in the first semester with an average score of 60% in the best four courses;
- in the third semester, only if (s)he
  - registers for a total of at least fourteen courses in the first three semesters, and
  - obtains at least 45% in each of at least nine courses in the first two semesters with an average score of 60% in the best nine courses;
- in the fourth semester, only if (s)he obtains at least 45% in each of at least fourteen courses in the first three semesters with an average score of 60% in the best fourteen courses.

Further, a student who is ineligible for a full stipend in the second, third or fourth semester, may be eligible for a half-stipend in that semester (i.e., second, third or fourth semester) if (s)he fulfils the registration requirements, and gets at least 45% score in each of the minimum number of individual courses as per the schedule listed above.

3.8 Mentor Committee

A group of students entering the M.Tech. (CrS) programme in a particular year is treated as a batch. The Mentor Committee for each batch of students is announced at the time of their admission, and remains valid until all the students of that batch leave the programme. This Committee advises the students on their choice of courses (including opting for a course more than once even after passing it), and as a matter of fact, on any other problem. Students should approach the Mentor Committee if they face any problems.

A student's choice of courses in a semester needs to be approved by the Mentor Committee. Students are required to submit the final course choice list to the Dean's office, signed by the members of the Mentor Committee for his/her batch, within 2 weeks from the start of the semester.

The Chairperson of the Mentor Committee acts as the Class Teacher for all students in a particular batch. All communications on academic matters with other authorities or statutory bodies of the institute by a student should be routed through the Class Teacher.

3.9 Teachers' Committee

The Teachers' Committee in a particular semester consists of all the teachers of the courses (including regular courses and lab courses) opted for by all students (across all years) of the MTCrS programme in that semester. The Chairperson and Convener of the existing Mentor Committees will be invited members of the Teachers' Committee. The Dean of Studies is the Chair of the Teachers' Committee. Every decision related to the academic performance of a student is taken after deliberation in the relevant Teachers' Committee. A decision can be taken by any higher authority/statutory body only after the recommendations of the relevant Teachers' Committee have been taken into cognizance.

3.10 CrS Dissertation Committee

This committee, formed for a batch at the start of its second year, is responsible for the following tasks:

- Create a list of dissertation topics from the relevant faculty members and circulate the list among the students of that batch. However, the students are free to choose a topic outside the list, provided a faculty member agrees to supervise such a project.
- Help a student with finding a Dissertation topic and supervisor, if needed.
- Approve cases of joint supervision.
- Conduct evaluation of the dissertation at appropriate times.

3.11 Prizes

At the end of the four semesters of a particular batch, the Mentor Committee may nominate a few students for their outstanding academic performances during this period, for awards.

There will be no consideration of prize for semester-specific performance.

4 Detailed Syllabi of the Courses

The courses are categorised as compulsory, formative and elective. Each course has a detailed syllabus, a few prerequisites, the number of hours of class needed, marks distribution, and names of reference books.

Prerequisites. Many courses have a few suggested prerequisite(s). The prerequisite courses have to be passed either at ISI or in the undergraduate / postgraduate degrees obtained before joining ISI. The students need to confirm with the concerned teacher if her/his prerequisite courses satisfy the demands of the particular course being taught at ISI. It is the responsibility of the students to verify from the concerned teacher that they satisfy the prerequisites.

Marks distribution. The marks distribution given against each course is suggestive for the teacher. A teacher of a course will announce within two weeks of the start of the course the explicit marks distribution for the course. The marks distribution for each course has two components.

Examination: This includes mid-semester and end-semester examinations. Except for the Computing Laboratory course that has no mid-semester and end semester examinations, all other courses have at least 50% weightage for the end-semester examination.

Lab/assignment: This component includes the internal assessment marks and comprises programming assignments, written assignments, class tests, quizzes, etc.

4.1 Compulsory and Formative Courses

Abstract Algebra and Number Theory

(a) Groups: Elementary properties, subgroups and cosets, Lagrange's theorem, cyclic groups, normal subgroups and quotient groups, homomorphism and isomorphism, permutation groups, Sylow's theorem and applications. Applications to number theory. **(4 weeks)**

Rings and fields: Introduction to rings, subrings, ideals and quotient rings, homomorphism and isomorphism, integral domains, principal ideal domain(PID), Euclidean domain(ED), Euclidean algorithm for gcd, extended Euclidean algorithm, modular inverse of an integer, Chinese Remainder Theorem(CRT), Euler's ϕ-function, quadratic residues, ring of polynomials, irreducible polynomials, factorizations, fields, field of fractions, finite fields. **(7 weeks)**

Selected topics in number theory: For example, arithmetic functions, Diophantine equations, continued fractions. **(3 weeks)**

(b) **Prerequisites:** None

(c) **References:**

1. I. Niven, H. S. Zuckerman, and H. L. Montgomery, An Introduction to the Theory of Numbers. Wiley, 1991.
2. J. B. Fraleigh, First Course in Abstract Algebra. Narosa/Addison-Wesley, New Delhi/Reading, 1982/1978.
3. I. N. Herstein, Topics in Algebra. John Wiley & Sons., 1987.
4. M. Artin, Algebra. Pearson, 2010.
5. D. M. Burton, Elementary Number Theory. Allyn and Bacon, 1976.

Data Structures and Algorithms

(a) Introduction: **[2 weeks]** Initial ideas of algorithms and their resource usage in terms of space and time complexity; RAM model, initial ideas of memory model and memory hierarchy; asymptotic notations; ideas of worst case, average case and amortized case analysis;

Data Structure basics: **[4 weeks]** Idea of Abstract Data Types and its concrete implementation; Basic data structures – Lists, Array implementation of lists; Linked lists, Doubly linked lists. Initial ideas of Queue and Stack and their implementation. Binary trees and their traversal.

Algorithms and Data Structures for searching and sorting: **[2 weeks]** Binary search and binary search trees; finding maximum and minimum, k largest elements in order; basic sorting and searching algorithms; heaps and heapsort, priority queue.

Basic Graph Algorithms and associated data structures: **[3 weeks]** Representation of graphs; traversal algorithms (BFS, DFS) and their applications using Queue and Stack data structure; Kruskal's and Prim's Algorithms and the use of data structure.

Advanced data structures and their applications: **[3 weeks]** Union Find Data structures, tree representation of a set, path compression techniques and analysis; Height-balanced binary search trees; Range search using K-d tree; B-tree and B+ tree.

Programming practices: Apart from theoretical analysis of data structures, programming implementations should be done as assignments. The Introduction to programming and Computing Laboratory courses acts as a supplement to this course.

(b) **Prerequisites:** None

(c) **References:**

1. T. A. Standish, Data Structures, Algorithms and Software Principles in C. Addison-Wesley, Reading, Mass., 1995.
2. L. Nyhoff, C++: An Introduction to Data Structures. Prentice Hall, Englewood Cliffs, 1998.
3. A. M. Tenenbaum, Y. Langsam and M. J. Augenstein, Data Structures Using C. Pearson, 1998.
4. D. E. Knuth, The Art of Computer Programming. Vol. 1, 3rd. ed. Narosa/Addison-Wesley, New Delhi/London, 1997.
5. T. A. Standish, Data Structure Techniques. Addison-Wesley, Reading, Mass., 1980.
6. E. Horowitz and S. Sahni, Fundamentals of Data Structures. Galgotia Booksource, New Delhi, 1977.
7. R. L. Kruse, Data Structures and Program Design in C. Prentice Hall of India, New Delhi, 1996.
8. A. Aho, J. Hopcroft, and J. Ullman, Data Structures and Algorithms. Addison-Wesley, Reading, Mass., 1983.
9. B. Salzberg, File Structures: An Analytical Approach. Prentice Hall, New Jersey, 1988.
10. T. Harbron, File System Structure and Algorithms. Prentice Hall, New Jersey, 1987.
11. P. E. Livadas, File Structure: Theory and Practice. Prentice Hall, New Jersey, 1990.

12. T. Coreman, C. Leiserson, R. Rivest and C. Stein, Introduction to Algorithms. PHI Learning Pvt. Ltd., New Delhi, 2009.
13. S. Sahani, Data Structure, Algorithms and Applications in JAVA. Universities Press (India) Pvt. Ltd., New York, 2005.
14. D. Wood, Data Structure, Algorithms and Performance. Addison-Wesley, Reading, Mass., 1993.
15. M. T. Goodrich, R. Tamassia and David Mount, Data Structures and Algorithms in C++. 2nd ed., Wiley, 2011.
16. M. A. Weiss, Data Structures and Algorithms in C. 2nd ed., Pearson, 2020.
17. Pat Morin, Open Data Structures. Athabasca University Press, 2020.

Computer Networks

(a) Introduction: Characteristics of computer networks, Network hardware, Network software, Layered network structures and Reference models.

Data communication fundamentals: Analog and digital transmissions, Different transmission media, Different transmission impairments, Different modulation techniques, Channel capacity, Basic concept of spread spectrum (DSSS and FHSS), Asynchronous and synchronous transmission, Multiplexing.

Communication networks: Introduction to LANs, MANs, and WANs; Switching techniques (Circuit-switching and Packet-switching), Network topologies, Ethernet and its performance, Repeaters, Hub, bridges, Switches and Routers.

Data link layer: Basic functions and services, Framing techniques, Error detection and correction, Flow control (Stop-and-wait and Sliding window), Performance analysis of stop-and-wait and sliding window protocols, MAC Protocols (ALOHA, CSMA, CSMA/CD, Collision free protocols, Limited contention protocol), Performance analysis of different MAC protocols, Wireless LAN (MACA, CSMA/CA).

Network Layer: Design issues, Internet protocols (IPV4, IPV6, ICMP, ARP, RARP, DHCP), IP addressing (Classful and Classless), Introduction to subnetting, NAT, Routing (Flooding, Distance vector, Link state), Congestion control (choke packets, leaky bucket, token bucket).

Transport Layer: Design issues and services, Connection establishment and release, TCP, UDP, TCP congestion control, TCP timer management, RPC, Introduction to ports, sockets and socket programming.

Application Layer: WWW, Emails, DSN.

Network Labs: Interprocess communications and socket programming: Implementation and realization of simple echo client-server over TCP and UDP, proxy web server, FTP, TELNET, Chat programs, DNS and HTTP. Implementation of client-server applications using remote procedure call. Create sockets for handling multiple connections and concurrent server. Simulating PING and TRACEROUTE commands.

(b) **Prerequisites:** None

(c) **References:**

1. L. L. Peterson and B. S. Davie, Computer Networks: A Systems Approach. Morgan Kaufmann Publishers, 2007.
2. A. S. Tanenbaum, Computer Networks. 5th ed., Prentice Hall, 2010.
3. W. Stallings, Data and Computer Communcations. 10th ed., Prentice Hall, 2017.
4. D. P. Bertsekas and R. G. Gallagher, Data Networks. 2nd ed., Prentice Hall, 1992.
5. B. A. Forouzan, Data Communications and Networking. McGraw-Hill.
6. W. R. Stevens, Unix Network Programming. PHI, 2009.
7. W. R. Stevens, TCP/IP Illustrated. Volume 1: The Protocols, Addison-Wesley Professional.

Digital Design and Computer Architecture

(a) Information representation: Number systems – binary, octal and hexadecimal numbers, signed number representation, fixed and floating point representations, and arithmetic.

Boolean algebra: Postulates and fundamental theorems, Representation of Boolean functions, canonical forms, fundamental Boolean operations, Karnaugh Maps, Minimizing Boolean functions.

Combinational Logic Design: Logic gates, Adder, Subtractor, Comparator, decoder, Multiplexer, PLA.

Sequential Logic Design: Latches and Flip-Flops, Finite state models for sequential machines, registers, shift register, ripple counters, synchronous counters; state diagrams, characteristics and excitation tables of various memory elements, state minimization for synchronous and asynchronous sequential circuits.

ALU Design: Addition of numbers – carry look-ahead and pre-carry vector approaches, carry propagation-free addition. Multiplication - using ripple carry adders, carry save adders, redundant number system arithmetic, Booth's algorithm. Division - restoring and non-restoring techniques, using repeated multiplication. Floating-point arithmetic, multiplication and addition algorithms. ALU design, instruction formats, addressing modes.

Processor Design: ISA and Microarchitecture design, hardware control unit design, Pipelining, Out of order execution.

Memory Organization: Random and serial access memories, static and dynamic RAMs, ROM. Cache Memory fundamentals.

I/O Organization: Different techniques of addressing I/O devices, data transfer techniques, programmed interrupt, DMA.

(b) **Prerequisites:** None

(c) **References:**

1. Z. Kohavi, Switching and Finite Automata Theory. 2nd ed., McGraw Hill, New York, 1978.
2. E. J. Mcclusky, Logic Design Principles. Prentice Hall International, New York, 1986.
3. John L. Hennessy and David A. Patterson, Computer Architecture: A Quantitative Approach. 2012.
4. John L. Hennessy and David A. Patterson, Computer Organization and Design: The Hardware/Software Interface. 2021.
5. J. P. Hayes, Computer Architecture and Organization. McGraw Hill, New York, 1988.
6. P. P. Choudhury, Computer Organization and Design. Prentice Hall of India, New Delhi, 1994.
7. M. M. Mano, Computer System Architecture. 3rd ed., Prentice Hall of India, New Delhi, 1993.
8. W. Stallings, Computer Organization and Architecture: Principles of Structure and Function. 2nd ed., Macmillan, New York, 1990.

Computing Laboratory

(a) This laboratory course is meant to complement the Data Structures and Algorithms and Design and Analysis of Algorithms courses. The material, assignments and examinations are to be designed based on the coverage in those courses. The initial programming language can be `C` or can be decided by the instructor based on the background of the students. For this course, conventional mid-semester and end-semester examinations may be replaced by programming tests.

The laboratory sessions should include but are not limited to:

Problem solving techniques: recursion, divide-and-conquer, branch-and-bound, dynamic programming.

Function pointers, generic data structures, implementation of generic stacks and queues.

Trees: binary trees, recursive and non-recursive traversal of trees; heaps; binary search trees, implementation of balanced search trees (AVL / Red-Black tree).

Tries.

Disjoint set data structures (union-find).

Graphs: directed and undirected graphs, adjacency matrix and list representations, traversal algorithms and their applications; connected components, strongly connected components; shortest path algorithms; minimum spanning tree algorithms.

Introduction to other programming languages: C++ or Python.

Object oriented programming: introduction, classes and methods, polymorphism, inheritance.

Project management: multi-file programs, build systems (GNU make, CMake, Meson or similar), version management using Git.

Program debugging and testing: elementary testing methods, developing the test plan, developing a test, concept of writing automation scripts using bash / tcsh (or similar).

(b) **Prerequisites:** Introduction to Programming, Data Structures and Algorithms, Design and Analysis of Algorithms (may be taken concurrently)

(c) **References:**

1. T. A. Standish, Data Structures, Algorithms and Software Principles in C. Addison-Wesley, Reading, Mass., 1995.

2. L. Nyhoff, C++: An Introduction to Data Structures. Prentice Hall, Englewood Cliffs, 1998.
3. A. M. Tenenbaum, Y. Langsam and M. J. Augenstein, Data Structures Using C. Pearson, 1998.
4. D. E. Knuth, The Art of Computer Programming. Vol. 1, 3rd. ed. Narosa/Addison-Wesley, New Delhi/London, 1997.
5. T. A. Standish, Data Structure Techniques. Addison-Wesley, Reading, Mass., 1980.
6. E. Horowitz and S. Sahni, Fundamentals of Data Structures. Galgotia Booksource, New Delhi, 1977.
7. R. L. Kruse, Data Structures and Program Design in C. Prentice Hall of India, New Delhi, 1996.
8. A. Aho, J. Hopcroft, and J. Ullman, Data Structures and Algorithms. Addison-Wesley, Reading, Mass., 1983.
9. B. Salzberg, File Structures: An Analytical Approach. Prentice Hall, New Jersey, 1988.
10. T. Harbron, File System Structure and Algorithms. Prentice Hall, New Jersey, 1987.
11. P. E. Livadas, File Structure: Theory and Practice. Prentice Hall, New Jersey, 1990.
12. T. Cormen, C. Leiserson, R. Rivest and C. Stein, Introduction to Algorithms. PHI Learning Pvt. Ltd., New Delhi, 2009.
13. S. Sahni, Data Structure, Algorithms and Applications in JAVA. Universities Press (India) Pvt. Ltd., New York, 2005.
14. D. Wood, Data Structure, Algorithms and Performance. Addison-Wesley, Reading, Mass., 1993.
15. M. T. Goodrich, R. Tamassia and David Mount, Data Structures and Algorithms in C++. 2nd ed., Wiley, 2011.
16. B. W. Kernighan and D. M. Ritchie, The C Programming Language. Prentice Hall of India, 1994.
17. B. Gottfried: Programming in C, Schaum Outline Series, New Delhi, 1996.
18. B. W. Kernighan and R. Pike, The Unix Programming Environment. Prentice Hall of India, 1996.
19. R. G. Dromey, How to Solve it by Computers. Pearson, 2008.

Cryptology I

(a) Symmetric-Key encryption:

   i. Classical ciphers (like Shift, Substitute, Affine, Vigenère, Hill) and their cryptanalysis;

   ii. Information Theoretic Security, and One-Time Pad;

   iii. Pseudorandom Generators (PRG) and Stream ciphers (like CSS, RC4, Trivium, Salsa/ChaCha); Cryptanalysis of Stream Ciphers;

   iv. Pseudorandom Function (PRF) and Pseudorandom Permutation (PRP); Block ciphers (like DES, AES); Cryptanalysis of Block ciphers;

   v. Mode of Operations;

   vi. Notion of CPA and CCA security with examples;

   Cryptographic hash functions: Preimage, Second-Preimage and Collision resistance, Merkle-Damgard Transform; Universal and Almost-Universal hash functions;

   Symmetric-Key authentication: Message Authentication Code (MAC) (like CBC-MAC, NMAC, HMAC), and their cryptanalysis;

   Modern modes of operations: Authenticated Encryption;

   Key Exchange Protocol: The Diffie-Hellman key exchange and its cryptanalysis;

   Introduction to Public-key encryption (like The RSA, ElGamal, Rabin, Paillier encryption schemes) and their cryptanalysis;

   Digital Signatures (like RSA, ElGamal, DSA) and their cryptanalysis;

   Elliptic curve based Cryptography: The group of points on an elliptic curve; Elliptic curves over finite fields; The curve P256; The Montgomery and Edwards curves: curve 25519; ECDH and ECDSA;

(b) **Prerequisites:** None

(c) **References:**

   1. J. Katz and Y. Lindell, Introduction to Modern Cryptography. Chapman & Hall/CRC, 2007.

   2. D. R. Stinson, Cryptography Theory and Practice. 3rd ed., Chapman & Hall/CRC, 2006.

   3. D. Boneh, V. Shoup, A Graduate Course in Applied Cryptography, 2023. (Source: http://toc.cryptobook.us).

4. B. S. Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C. 2nd ed., John Wiley & Sons, New York, 1995.
5. A. Menezes, P. C. Van Oorschot, and S. A. Vanstone, Handbook of Applied Cryptography. CRC Press, Boca Raton, 1996.

Cryptology II

(a) Theoretical construction of pseudorandom objects: One way functions, pseudorandom generators, pseudorandom functions and pseudorandom permutations.

Public-Key Encryption and Commitments: El-Gamal and its variants, additively homomorphic encryptions; CCA-security - Fujisaki Okamoto; Cramer-Shoup; Commitment Schemes; Pedersen, Hash commitments;

Zero-Knowledge Proofs: Knowledge Extraction; Zero-knowledge Simulation; Sigma Protocols (like Schnorr Chaum-Pederson; Okamoto's); General Sigma Protocols for any algebraic relations; Existential Soundness; OR proofs; Proof of Equal Plaintext; Proof of 1-bit messages;

Secure Multiparty Computations: Idea of Simulation-based definition; Oblivious Transfer; Yao's Garbled Circuits; GMW- dishonest majority; BGW - honest majority; Beaver's protocol in the pre-processing model - maliciously secure; MPC in the head - ZK from MPC;

Bilinear pairings: Bilinear pairings; Signature schemes from bilinear pairings; Identity-based encryption; Broadcast encryption.

Lattice Based Cryptology: Integer lattices; hard problems on lattices: SIS, LWE and ring LWE problems; Trapdoor sampling from lattices; signatures and encryption schemes from lattices;

Some other relevant topics may be covered – however it will depend on the instructor;

(b) **Prerequisites:** Cryptology I

(e) **References:**

1. O. Goldreich, Foundations of Cryptography Vol 1 and 2, Cambridge University Press.
2. D. Boneh and V Shoup, A Graduate Course in Applied Cryptography. (Source: `http://toc.cryptobook.us`)

3. S. D. Galbraith, Mathematics of Public Key Cryptography. Cambridge University Press, 2012.
4. R. Pass and A. Shelat, A Course in Cryptography, Lecture notes. (Source: https://www.cs.cornell.edu/courses/cs4830/2010fa/lecnotes.pdf)
5. D. Micciancio, S. Goldwasser, Complexity of Lattice Problems: A Cryptographic Perspective. Kluwer, 2002.
6. L. C. Washington, Elliptic Curves: Number Theory and Cryptography. 2nd ed., CRC Press 2008.
7. S. Chatterjee and P. Sarkar, Identity-Based Encryption. Springer, 2011.

Computing Systems Security

(a) Computer Security : Linux Security Basics, Set-UID Privileged Programs, Shellshock Attack, Buffer-Overflow Attacks, Return-to-Libc Attacks, Race Condition, Dirty COW Attack, Meltdown and Spectre Attacks, Format String Attacks

Internet Security: Network Basics, Packet Sniffing and Spoofing, MAC Layer and Attacks, Network Layer: IP, ICMP and Attacks, Transport Layer: UDP and Attacks, Tansport Layer: TCP and Attacks, DNS and Attacks, Virtual Private Network (VPN), Firewall, BGP Routing and Attacks, Heartbleed Attack

Web Security: Web Security Basics, Cross-Site Request Forgery Attack, Cross-Site Scripting Attack, SQL Injection Attack, Clickjacking Attack

(b) **Prerequisites:** Operating Systems, Computer Networks, Discrete Mathematics

(c) **References:**

1. Computer & Internet Security: A Hands-on Approach, Wenliang Du.
2. Computer Security: A Hands-on Approach, Wenliang Du.
3. Internet Security: A Hands-on Approach, Wenliang Du.
4. C. P. Pfleeger, S. L. Pfleeger, and J. Margulies, Security in Computing. 5th ed., Prentice Hall, 2015.
5. D. Wheeler, Secure Programming HOWTO. (Source: https://www.dwheeler.com/secure-programs/)

6. M. Zalewski, Browser Security Handbook. Michael Zalewski, Google. (Source: https://code.google.com/archive/p/browsersec/wikis/Main.wiki)
7. B. S. Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C. 2nd ed., John Wiley & Sons, New York, 1995.
8. A. Menezes, P. C. Van Oorschot, and S. A. Vanstone, Handbook of Applied Cryptography. CRC Press, Boca Raton, 1996.

Database Management Systems

(a) Introduction: Purpose of database systems, data abstraction and modelling, instances and schemes, database manager, database users and their interactions, data definition and manipulation language, data dictionary, overall system structure.

Relational model: Structure of a relational database, operation on relations, relational algebra, tuple and domain relational calculus, salient feature of a query language.

SQL: domain types, construction, alteration and deletion of tables, query structure and examples, natural joins and other set operations, aggregations, nested subqueries, inserting, modifying and deleting data, advanced joins, views, transactions, integrity constraints, cascading actions, autorization and roles. Hands on and practical assignments.

Entity - relationship model: Entities and entity sets, relationships and relationship sets, mapping constraints, E - R diagram, primary keys, strong and weak entities, reducing E - R diagrams to tables.

Introduction to hierarchical and network model: Data description and tree structure diagram for hierarchical model, retrieval and update facilities, limitations; Database task group (DDBTG) model, record and set constructs retrieval and update facilities, limitations.

Databases in application development: cursors, database APIs, JDBC and ODBC, JDBC drivers, Connections, Statements, ResultSets, Exceptions and Warnings. Practical case studies.

Normalization: Anomalies in RDBMS, importance of normalization, functional, multi-valued and join dependencies, closures of functional dependencies and attribute sets, 1NF, 2NF, 3NF and BCNF; (Optionally) 4NF, 5NF and 6NF; Discussion on tradeoff between performance and normalization.

Database tuning: Index selection and clustering, tuning of conceptual schema, denormalization, tuning queries and views;

Query optimization: Importance of query processing, equivalence of queries, join ordering, cost estimation, cost estimation for complex queries and joins, optimizing nested subqueries, I/O cost models, external sort.

Crash recovery: Failure classification, transactions, log maintenance, check point implementation, shadow paging, example of an actual implementation. Concurrency Control in RDBMS: Testing for serializability, lock based and time - stamp based protocols; Deadlock detection and Recovery.

NoSQL: Introduction to NoSQL databases, ACID vs BASE requirements, practical exercises with one NoSQL system (e.g., MongoDB or Neo4j).

MapReduce and Hadoop: Basics of MapReduce, Basics of Hadoop, Matrix-vector multiplication using MapReduce, relational algebra using MapReduce, matrix multiplication using MapReduce, combiners, cost of MapReduce algorithms, basics of Spark, practical exercises using Spark.

(b) **Prerequisites:** None

(c) **References:**

1. A. Silberschatz, H. F. Korth and S. Sudarshan, Database System Concepts. Tata McGraw-Hill, 6th ed., 2011. (Source: http://www.db-book.com)
2. R. Ramakrishnan and J. Gehrke, Database Management Systems, McGraw-Hil, 3rd ed., 2007. (Source: http://pages.cs.wisc.edu/~dbbook)
3. J. Leskovec, A. Rajaraman, J. Ullman, Mining of Massive Datasets. (Source: http://www.mmds.org)
4. C. J. Date, An Introduction to Database Systems, Pearson Education, Inc., 8th ed., 2006.
5. R. Elmasri and S. B. Navathe, Fundamentals of Database Systems. Pearson Education, Inc., 4th ed., 2004.
6. G. Harrison and S. Feuerstein, MySQL stored procedure programming. O'Reilly Media, Inc., 2006.
7. P. J. Sadalage and M. Fowler, NoSQL distilled: a brief guide to the emerging world of polyglot persistence. Pearson Education, Inc., 1st ed., 2013.

Design and Analysis of Algorithms

(a) Paradigms of Algorithm Design and sorting and selection: **[2 weeks]** Initial ideas of algorithmic paradigms like divide-and-conquer, greedy, induction, dynamic programming, etc. Comparisons between various sorting algorithms, Quick sort (Average case complexity), lower bounds on comparison based sorting algorithm, non-comparative (linear time) sorting algorithms;

Recursive Algorithms: **[2 weeks]** Karatsuba Algorithm; Median Finding Algorithm; Strassen's Algorithm; Dynamic Programming Algorithms – longest common subsequence, matrix chain multiplication, etc.

Hashing: **[1 weeks]** Hash tables, hash functions, open addressing, perfect hashing.

Graph Algorithms: [2 weeks] Shortest Path Algorithms; Applications of depth first search – cycle detection in graph, topological sort, articulation points in graphs, strongly connected components; Network Flow – Max-Flow Min-Cut theorem, Max-flow algorithms like Ford-Fulkerson;

Algorithms from different domains: [2 weeks] FFT, Number Theoretic Algorithms, Geometric Algorithms – convex hull, line segment intersection.

Randomized Algorithms: [1 week] Randomized quicksort, Min-cut, Universal Hashing.

"Hard" Problems: **[3 weeks]** P vs. NP, NP-completeness, statement of Cook's theorem, some important NP-complete problems, problems beyond NP, ideas on attacking NP-complete problems, using reductions to solve hard problems.

Approximation Algorithms: [1 week] Constant multiplicative factor approximation algorithms – bin packing, vertex cover, Euclidean traveling salesman problem, etc. notions of hardness of approximation.

(b) **Prerequisites:** None

(c) **References:**

1. A. Aho, J. Hopcroft and J. Ullman, The Design and Analysis of Computer Algorithms. A. W. L, International Student Edition, Singapore, 1998.
2. S. Baase, Computer Algorithms: Introduction to Design and Analysis. 2nd ed., Addison- Wesley, California, 1988.
3. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, Introduction to Algorithms. third edition, MIT Press, 2009.

4. E. Horowitz and S. Sahni, Fundamental of Computer Algorithms. Galgotia Pub. /Pitman, New Delhi/London, 1987/1978.
5. K. Mehlhorn, Data Structures and Algorithms. Vol. 1 and Vol. 2, Springer-Verlag, Berlin, 1984.
6. A. Borodin and I. Munro, The Computational Complexity of Algebraic and Numeric Problems. American Elsevier, New York, 1975.
7. D. E. Knuth, The Art of Computer Programming. Vol. 1, 2nd ed., Narosa/Addison-Wesley, New Delhi/London, 1973; Vol. 2: 2nd ed., Addison-Wesley, London, 1981; Vol. 3: Addison-Wesley, London, 1973.
8. S. Winograd, The Arithmetic Complexity of Computation. SIAM, New York, 1980.
9. S. Dasgupta, C. Papadimitriou, U. Vazirani, Algorithms. McGraw Hill India, Chennai.
10. J. Kleinberg and E. Tardos, Algorithm Design. Pearson Education.
11. M. Mitzenmacher and E. Upfal, Probability and Computing: Randomized Algorithms and Probabilistic Analysis. Cambridge University Press.

Discrete Mathematics

(a) Combinatorics: **[6 weeks]** Introduction to posets, relations, asymptotic notations, proof techniques, induction on sets and graphs, multinomial theorem, principle of inclusion exclusion; pigeonhole principle; Counting using (in)distinguishable balls and (in)distinguishable bins, functions and counting, Stirling number; Recurrences and ways to solve them, solving recurrences using asymptotic notations. Generating functions. Introduction to Probabilistic methods.

Graph Theory: **[6 weeks]** Graphs and digraphs, isomorphism, connectedness and reachability, representation of graphs, Eulerian paths and circuits in graphs and digraphs, Hamiltonian paths and circuits in graphs and tournaments, trees; Minimum spanning tree, rooted trees and binary trees, planar graphs, Euler's formula, statement of Kuratowski's theorem, dual of a planar graph, independence number and clique number, chromatic number, statement of Four-color theorem, dominating sets and covering sets, matching, network flows

Basic Logic: **[2 weeks]** Introduction to propositional logic – propositions and connectives, syntax; semantics – truth assignments and truth tables, validity and satisfiability, tautology; Adequate set of connectives; Introduction to quantifers - first order logic over simple relational vocabulary.

(b) **Prerequisites:** None

(c) **References:**

1. C. L. Liu, Elements of Discrete Mathematics. McGraw Hill, 1985.
2. D. B. West, Introduction to Graph Theory. Pearson, 2000.
3. R. Diestel, Graph Theory. Springer, 2010.
4. F. Harary, Graph Theory. Narosa Publishing House, 2001.
5. E. Mendelsohn, Introduction to Mathematical Logic. Van-Nostrand, 1979.
6. J. Matousek and J. Nesetril, Invitation to Discrete Mathematics Paperback. Oxford University Press, 2009.
7. D. West, Combinatorial Mathematics. Cambridge University Press, 2021.
8. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms. MIT Press, 1990.

Linear Algebra

(a) <u>Matrices and System of Equations:</u> System of Linear Equations, Row Echelon Form, Matrix Arithmetic, Matrix Algebra, Elementary Matrices, Partitioned Matrices, Determinant and its properties.

<u>Vector Spaces:</u> Definition and Examples, Subspaces, Linear Independence, Basis and Dimension, Change of Basis, Row Space, Column Space, Null space

<u>Inner product spaces:</u> The Euclidean dot product, Orthogonal Subspaces, Least Squares Problems, Orthonormal Sets, The Gram-Schmidt Orthogonalization Process, Orthogonal Polynomials

<u>Linear Transformations:</u> Definition and Examples, Matrix Representations of Linear Transformations, Similarity

<u>Eigenvalues and Eigenvectors:</u> System of Linear Differential Equations, Diagonalization, Hermitian Matrices, Singular Value Decomposition.

<u>Quadratic Forms:</u> Classifiction and characterisations, Optimisation of quadratic forms.

<u>Algorithms:</u> Gaussian Elimination with different Pivoting Strategies; Matrix Norms and Condition Numbers; Orthogonal Transformations; The Eigenvalue Problem; Least Squares Problems.

(b) **Prerequisites:** None

(c) **References:**

1. G. Strang, Introduction to Linear Algebra. 5th ed., Wellesley - Cambridge Press, 2016.
2. S. Axler, Linear Algebra Done Right. 4th ed., Springer, 2025.
3. G. H. Golub and C. F. Van Loan, Matrix Computations. 4th Edition, The Johns Hopkins University Press, Baltimore, 2013.

Introduction to Programming

(a) Introduction to UNIX-like systems.

Basics of C programming: variables, operators, expressions, control flow (sequential, conditional, repetitive, using factorial, Fibonacci, GCD, etc. as examples), basic I/O, arrays, strings, pointers, functions, parameter passing (call by value, call by reference), structures, unions, dynamic memory allocation, multi-dimensional arrays, recursion, preprocessor directives, file I/O.

Linked lists, stacks, queues, circular queues: implementation of basic operations (insertion, deletion, reversal, etc.) and applications (prefix, infix, postfix expressions, polynomial addition and multiplication, sparse matrix multiplication and addition, etc.).

Sorting: insertion, selection, bubble sort, merge sort, quick sort; binary search.

Hashing: hash table implementation, searching, insertion and deletion.

Writing and using multi-file programs and libraries.

Efficient programming.

Memory management in programs: call stack, heap management.

Debugging tools: GDB, Valgrind.

(b) **Prerequisite(s):** None

(c) **References:**

(a) B. W. Kernighan and D. M. Ritchie, The C Programming Language. Prentice Hall, 1980.

(b) B. Gottfried, Programming in C. Schaum Outline Series, 1996.

(c) B. Stroustrup, The C++ Programming Language. 2nd ed., Addison-Wesley, 1995.

(d) C. S. Horstmann, Core Java Volume I – Fundamentals. 11th ed., Prentice Hall, 2018.

(e) J. Bloch, Effective Java. 3rd ed., Addison-Wesley, 2018.

(f) B. W. Kernighan and R. Pike, The Practice of Programming, Addison-Wesley.

(g) B. W. Kernighan and R. Pike, The Unix Programming Environment. Prentice Hall.

Operating Systems

(a) *Introduction:* Basic architectural concepts, interrupt handling, concepts of batch-processing, multiprogramming, time-sharing, real-time operations; Resource Manager view, process view and hierarchical view of an OS.
Memory management: Partitioning, paging, concepts of virtual memory, demand-paging – page replacement algorithms, working set theory, load control, segmentation, segmentation and demand-paging, Cache memory management.
Process management: CPU scheduling – short-term, medium term and long term scheduling, non-preemptive and preemptive algorithms, performance analysis of multiprogramming, multiprocessing and interactive systems; Concurrent processes, precedence graphs, critical section problem - 2-process and n-process software and hardware solutions, semaphores; Classical process co-ordination problems, Producer-consumer problem, Reader-writer problem, Dining philosophers problem, Barber's shop problem, Interprocess communication.
Concurrent Programming: Critical region, conditional critical region, monitors, concurrent languages (eq. concurrent Pascal), communicating sequential process (CSP); Deadlocks: prevention, avoidance, detection and recovery.
Device Management: Scheduling algorithms – FCFS, shortest-seek-time-first, SCAN, C-SCAN, LOOK, C-LOOK algorithms, spooling, spool management algorithm.
Information Management: File concept, file support, directory structures, symbolic file directory, basic file directory, logical file system, physical file system, access methods, file protection, file allocation strategies.
Protection: Goals, policies and mechanisms, domain of protection, access matrix and its implementation, access lists, capability lists, Lock/Key mechanisms, passwords, dynamic protection scheme, security concepts and public and private keys, RSA encryption and decryption algorithms.

A case study: UNIX OS file system, shell, filters, shell programming, programming with the standard I/O, UNIX system calls.

(b) **Prerequisites:** None

(c) **References:**

1 A. Silberschatz and P. B. Galvin, Operating Systems Concepts. 5th ed., John Wiley and Sons, New York, 1998.

2 J. L. Peterson and A. Silberschatz, Operating Systems Concepts. Addison-Wesley, Reading, Mass., 1987.

3 P. B. Hansen, Operating System Principles, Prentice Hall, Englewood Cliffs, 1980.

4 A. S. Tannenbaum, Modern Operating Systems. Prentice Hall, Englewood Cliffs, 1992.

5 S. E. Madnick and J. J. Donovan, Operating Systems. McGraw Hill, New York, 1974.

Probability and Stochastic Processes

(a) Introduction: Sample space, probabilistic models and axioms, conditional probability, Total probability theorem and Bayes' rule, independence.

Discrete random variables: basics, probability mass functions, functions of random variables, expectation, variance, joint probability mass functions, conditioning and independence, notions of Bernoulli, Binomial, Poisson, Geometric, etc., covariance and correlation, conditional expectation and variance, some introduction to probabilistic methods.

Continuous random variables: basics, probability density functions, cumulative distribution function, normal random variable.

Moments and deviations: Markov's inequality, Chebyshev's inequality, Chernoff bounds

Limit theorems: Weak law of large numbers, central limit theorem, strong law of large numbers (proofs under some restrictive setups, if possible.)

Stochastic processes: Bernoulli and Poisson processes, branching processes. Markov chains, classification of states, ideas of stationary distributions. Introduction to Martingales and stopping times.

Applications to computer science: Balls and bins, birthday paradox, hashing, sorting, random walks, etc.

(b) **Prerequisites:** None

(c) **References:**

1. D. P. Bertsekas and J. N. Tsitsiklis, Introduction to Probability. 2nd ed., Athena Scientific, 2008.
2. W. Feller, An Introduction to Probability and its Applications. Volume I (3rd Edition), Wiley, 2008
3. W. Feller, An Introduction to Probability and its Applications. Volume II (2nd Edition), John Wiley & Sons, Inc, 1971.
4. S. Ross, A First Course in Probability, 11th ed., Academic Press, 2014.
5. N. Alon and J. Spencer, The Probabilistic Method. 4th ed., Wiley-Blackwell, 2016.
6. M. Mitzenmacher and E. Upfal, Probability and Computing. 2nd ed., Cambridge University Press, 2017.

Statistical Methods

(a) Review of Descriptive statistics: Representation of Data (collection, tabulation and diagrammatic representation of different types of univariate and bivariate data); Measures of Central Tendency (Mean, Median, Quartiles and Percentiles, Trimmed Mean, Mode); Measures of Dispersion (Range, Variance and Standard Deviation, Mean Deviation, Quartile Deviation), Measures of Shape (Skewness, Kurtosis), Contingency Table

Correlation: Pearson's product-moment correlation coefficient, Spearman's rank correlation, Kendall's tau statistic.

Regression: Simple and multiple linear regression, Method of least squares.

Estimation: Method of moments, Method of maximum likelihood estimation.

Hypothesis Testing: Concept of null and alternative hypotheses, Acceptance and critical regions, Probabilities of Type I and Type II errors, Level and Power of a test, Most powerful tests, Tests for mean and variance of a normal distribution, Randomized test, Tests for parameters of Binomial and Poisson distributions. Concept of

P-value, Z-test, paired and unpaired t-test, Likelihood ratio tests, χ^2 test for categorical data.

Interval Estimation: Interval estimation and its connection with hypothesis testing. Interval estimation for parameters of normal, binomial and Poisson distributions.

ANOVA: Fixed effect model for one-way classified data, two-way classified data with one observation per cell (if time permits).

Bayesian Estimation and Bayesian Statistical Testing

(b) **Prerequisite:** None.

(c) **References:**

1. C. E. Croxton and D. J. Cowden, Applied General Statistics, Pitman Publishing, 3rd ed., 1968.
2. M. J. Schervish, Theory of Statistics. Springer Series in Statistics, 1995.
3. D. Montgomery and G. C. Runge, Applied Statistics and Probability for Engineers, Wiley, 1994.
4. A. M. Goon, M. Gupta and B. Dasgupta, Fundamentals of Statistics. Vol I, II, III and IV, World Press.

Theory of Computation

(a) Automata and Languages: Finite automata, regular languages, regular expressions, deterministic and non-deterministic finite automata, minimization of finite automata, closure properties, Kleene's Theorem, Pumping Lemma and its application, Myhill-Nerode theorem and its uses; Context-free grammars, context-free languages, Chomsky normal form, closure properties, Pumping Lemma for context-free languages, push down automata.

Computability: Computable functions, primitive and recursive functions, universality, halting problem, recursive and recursively enumerable sets, parameter theorem, diagonalisation, reducibility, Rice's Theorem and its applications. Turing machines and variants; Equivalence of different models of computation and Church-Turing thesis.

Introduction to Complexity: Discussions on time and space complexities; P and NP, NP-completeness, Cook's Theorem, other NP-Complete problems; PSPACE; polynomial hierarchy.

(b) **Prerequisites:** Discrete Mathematics

(c) **References:**

1. N. J. Cutland, Computability: An Introduction to Recursive Function Theory. Cambridge University Press, London, 1980.
2. M. D. Davis, R. Sigal and E. J. Weyuker, Complexity, Computability and Languages. Academic Press, New York, 1994.
3. J. E. Hopcroft, J. D. Ullman and R. Motwani, Introduction to Automata Theory, Languages and Computation. Addison-Wesley, California, 2001.
4. H. R. Lewis and C. H. Papadimitriou, Elements of the Theory of Computation. Prentice Hall, Englewood Cliffs, 1981.
5. M. Sipser, Introduction to the Theory of Computation. PWS Pub. Co., New York, 1999.
6. M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to The Theory of NP- Completeness. Freeman, New York, 1979.

4.2 CrS Elective courses

Blockchains and Cryptocurrencies

(a) This course will provide an introduction to blockchains and cryptocurrencies, including Bitcoin and contracts. Basic tools will be introduced first, followed by the implementation of practical blockchain protocols.

i. Motivation and overview of blockchains.

ii. Building blocks: Hash functions, signature schemes, zero-knowledge proofs, consensus algorithms.

iii. Bitcoin: Transactions, blocks, mining, scripting, attacks on mining.

iv. Proof of work, proof of stake, proof of burn, proof of storage.

v. Smart contracts

vi. Privacy issues: Anonymity, mixing techniques, privacy with ZK-Snarks.

vii. Permissioned blockchains: Distributed consensus, sharing algorithms, privacy issues.

viii. Scaling issues: Lightning networks, Payment networks.

ix. Platforms and ledgers: Ethereum, Ripple, Hyperledger, Algorand, etc.

x. Altcoins and their analysis.

xi. Applications: Banking and finance, Data Sharing, Supply-chain, etc.

(b) **Prerequisites:** None

(c) **References:**

1. A. Narayanan, J. Bonneau, E. Felten, A. Miller, S Goldfeder, J. Clark: Bitcoin and Cryptocurrency Technologies, Princeton University Press. 2017.

2. A. M. Antonopoulos: Mastering Bitcoin: Programming the Open Blockchain, O'Reilly, 2017.

3. Web Resources.

Computational Complexity

(a) Introduction: Review of machine models, Turing machines and its variants, reduction between problems and completeness, time and space complexity classes

Structural Results: Time and space hierarchy theorems, polynomial hierarchy, Ladner's theorem, relativization, Savitch's theorem

Circuit Complexity: Circuits and non-uniform models of computation, parallel computation and NC, P-completeness, circuit lower bounds, AC^0 and parity not in AC^0, Hastad's Switching Lemma, introduction to natural proof barrier

Random Computation: Probabilistic computation and complexity classes and their relations with other complexity classes, BPP=P?

Interactive proofs: Introduction to Arthur-Merlin Games, IP=PSPACE, multiprover interactive proofs, introduction to PCP theorem

Complexity of counting: Complexity of optimization problems and counting classes, Toda's theorem, inapproximability, application of PCP theorem to inapproximability and introduction to unique games conjecture

Cryptography: Public-key cryptosystems, one-way functions, trapdoor-functions, application to derandomization

(b) **Prerequisites:** Discrete Mathematics, Design and Analysis of Algorithms

(c) **References:**

1. J. Balcazar, J. Diaz, and J. Gabarro, Structural Complexity – I and II. Springer-Verlag, Berlin, 1988/1990.
2. D. P. Bovet and P. Crescenzi, Introduction to the Theory of Complexity. Prentice Hall, Englewood Cliffs, 1994.
3. M. Sipser, Introduction to Theory of Computation, PWS Publishing, New York, 1999.
4. C. H. Papadimitriou, Computational Complexity. Addison-Wesley, 1994.
5. J. E. Hopcroft and J. D. Ullman, Introduction to Automata Theory. Languages and Computation, Addison-Wesley, 1979.
6. O. Goldreich, Lecture Notes on Computational Complexity. Tel Aviv University, 1999.
7. S. Arora and B. Barak, Computational Complexity: A Modern Approach. Cambridge University Press, 2009.

Computational Number Theory

(a) i. Divisibility, primality, GCD, factorization.

ii. Congruences: Basic properties; solving linear congruences, Chinese remainder theorem. Euler's ϕ function, quadratic residues.

iii. Computing with integers: Asymptotic notations, Integer arithmetic, computing in $\mathbb{Z}_n$.

iv. Euclid's Algorithm: Division Algorithm, extended Euclidean Algorithm, Computing modular inverses.

v. Distribution of primes: Chebyshev's Theorem and the distribution of primes, Bertrand's postulate, Prime Number Theorem.

vi. Probabilistic Algorithms: Basic definitions, generating a random prime, generating a random factored number. Primality testing - Miller-Rabin; generating random primes using Miller-Rabin, Factoring and computing Euler's ϕ function.

vii. Finding generators and discrete log in $\mathbb{Z}_p^*$. Diffie-Hellman key exchange. Testing quadratic residuosity. Computing modular square root. Quadratic residuosity assumption.

viii. Subexponential-time discrete log and factoring.

ix. Algorithms for finite fields.

x. Special topics.

(b) **Prerequisites:** None

(c) **References:**

1. Victor Shoup: A Computational Introduction to Number Theory and Algebra, Cambridge University Press.
2. Abhijit Das: Computational Number Theory, CRC Press.

Cryptographic and Security Implementations

(a) This course would be project-based; by the end of the course, each student should have done two non-trivial implementation projects in the following broad areas.

  (a) Cryptographic Implementations:

    i. Instruction set architecture of modern processors with emphasis on instructions dedicated for cryptographic use.
    ii. Programming in micro controllers.
    iii. Software side channel attacks and countermeasures.
    iv. Principles of hardware design.
    v. Introduction to hardware description languages.
    vi. Hardware side channel attacks, fault attacks and countermeasures.
    vii. Crypto libraries.

  (b) Security Implementations:

    i. Logging, Auditing and Log Monitoring
    ii. Enforcing password complexity, aging, and lockout
    iii. Implementing Security Access Control
    iv. File locks, database locks
    v. Firewall and Antivirus implementation and setup.
    vi. Intrusion Detection and Malware Detection.

(b) **Prerequisites:** Introduction to Programming, Computing Laboratory

(c) **References:**

  (a) N. Ferguson, B. Schneier: Practical Cryptography, Wiley, 2003.

(b) B. S. Schneier: Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition, John Wiley and Sons, New York, 1995.

(c) F. Rodríguez-Henrıíquez, A. Díaz Pérez, N. A. Saqib, etin Kaya Koc: Cryptographic

(d) Algorithms on Reconfigurable Hardware, Springer.

(e) Luca Giancane: Side-Channel Attacks and Countermeasures, LAP Lambert Academic Publishing, 2012.

(f) Eric Cole: Network Security Bible, 2nd ed., Wiley 2009.

Computing Systems Security II

(a)
1. Cellular networks, Access Technologies, GSM, CDMA, GPRS, 3G networks, Wireless LAN, WLAN security.
2. Operating Systems Security:
   i. Access Control Fundamentals
   ii. Generalized Security Architectures
   iii. Analysis of security in Unix/Linux and problems with the design of its security architecture
   iv. Analysis of security in Windows and problems with its security architecture
   v. Security Kernels: SCOMP design and analysis, GEM-SOS design
   vi. Difficulties with securing Commercial Operating Systems (Retrofitting Security)
   vii. Security issues in Virtual Machine Systems
   viii. Security issues in sandboxing designs: design and analysis of Android.
3. Database Security:
   i. Introduction: Security issues faced by enterprises
   ii. Security architecture
   iii. Administration of users
   iv. Profiles, password policies, privileges and roles
   v. Database auditing

(b) **Prerequisites:** Operating Systems, Computer Networks, Database Management Systems

(c) **References:**

1. Ross Anderson: Security Engineering, 2nd ed., Wiley. Available online: http://www.cl.cam.ac.uk/~rja14/book.html.
2. C.P. Pfleeger, S.L. Pfleeger, J. Margulies: Security in Computing, 5th ed., Prentice Hall, 2015.
3. David Wheeler: Secure Programming HOWTO. Available online: https://dwheeler.com/secure-programs/.
4. Michal Zalewski: Browser Security Handbook, Michael Zalewski, Google. Available online: https://code.google.com/archive/p/browsersec/wikis/Main.wiki.
5. M. Gertz, S. Jajodia (Eds.): Handbook of Database Security, Springer, 2008.
6. B. S. Schneier: Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition, John Wiley and Sons, New York, 1995.
7. A. Menezes, P. C. Van Oorschot and S. A. Vanstone: Handbook of Applied Cryptog- raphy, CRC Press, Boca Raton, 1996.

Coding and Information Theory

(a) Introduction: Basic definitions: codes, dimension, distance, rate, error correction, error detection.

Entropy and its application

Shannon's Theorems: Noiseless coding; Noisy Coding; Shannon Capacity;

Gilbert Varshamov bound; Singleton bound; Plotkin bound

Linear Codes: Properties of linear codes; Hamming codes; Efficient decoding of Hamming codes; Dual of a linear code

Algebraic codes: Reed-Solomon codes; Concatenated codes; BCH codes; Reed-Muller codes; Hadamard codes; Dual BCH codes.

Algorithmic issues in coding: Decoding Reed-Solomon Codes; Decoding Concatenated Codes

List Decoding: List decoding; Johnson bound; List decoding capacity; List decoding from random errors. List decoding of Reed-Solomon codes.

Advanced Topics: Graph Theoretic Codes; Locality in coding: Locally decodable codes, locally testable codes; codes and derandomization.

(b) **Prerequisites:** Probability and Stochastic Processes, Linear Algebra.

(c) **References:**

1. V. Guruswami, A. Rudra, and M. Sudan, Essential Coding Theory, 2025. (Source: `https://cse.buffalo.edu/faculty/atri/courses/coding-theory/book`)
2. F. J. MacWilliams and N. J. A. Sloane, Theory of Error-Correcting Codes. North Holland Publishing Co., 1977.
3. J. H. van Lint, Introduction to Coding Theory. Springer, 1973.
4. V. S. Pleaa and W. C. Huffman (Eds.), Handbook of Coding Theory, I and II, Elsevier, 1998.

Hardware Security

(a) i. Introduction to Hardware Security

ii. Introduction to Verilog, ISE and Vivado Toolflow

iii. Algorithm to Hardware, Finite Field Architectures, Hardware Design for Finite Inverse

iv. Hardware Implementation of cryptographic algorithms like AES, DES and others

v. Introduction to Side Channel Analysis: Power Analysis, Fault Analysis of Cryptosystems like DES, AES

vi. Power Attack Countermeasures

vii. Threshold Implementation

(b) **Prerequisites:** None

(c) **References:**

1. Hardware Security: Design, Threats, and Safeguards, CRC Press, by Debdeep Mukhopadhyay, and Rajat Subhra Chakraborty.
2. Timing Channels in Cryptography, Springer, by Chester Rebeiro, Debdeep Mukhopadhyay, and Sarani Bhattacharya.
3. Hardware Security: A Hands-on Learning Approach by Swarup Bhunia, and Mark Tehranipoor.

4. Towards Hardware-Intrinsic Security: Foundations and Practice by Ahmad-Reza Sadeghi, and David Naccache (Eds.)
5. Handbook of FPGA Design Security by T. Huffmire, C. Irvine, T.D. Nguyen, T. Levin, R. Kastner, and T. Sherwood.
6. Power Analysis Attacks - Revealing the Secrets of Smart Cards by Stefan Mangard, Elisabeth Oswald, and Thomas Popp.
7. Cryptography Theory and Practice, Third Edition, by Douglas Stinson.

Math Toolkits for CS

(a) Information Theory and its applications: Entropy, Mutual Information, Chain rule, Relative entropy, KL divergence, and its applications in combinatorics and computer science

Spectral Graph Theory: Graph Laplacians and their eigenvalues, connections to random walks and mixing, isoperimetric and Cheeger inequalities, expanders, and algorithmic applications

Convex Geometry: Geometric properties of high-dimensional convex bodies, Fritz John's theorem and isotropy, Brunn-Minkowski's theorem and isoperimetric inequalities, Dvoretzky's theorem, concentration of measure, and applications to include volume computation and convex programming

Fourier Analysis on the Boolean cube: Fundamental results of Fourier analysis on the Boolean cube, and its applications in learning theory, social choice theory, and hardness of approximation

Algebraic Techniques in Combinatorics: Linear algebra background, orthogonality and rank arguments, and the polynomial method

(b) **Prerequisites:** Discrete Mathematics, Probability and Stochastic Processes

(c) **References:**

1. T. M. Cover and J. A. Thomas, Elements of Information Theory. John Wiley, 1991.
2. L. Trevisan, Graph Partitioning, Expanders and Spectral Methods. Lecture Notes, UC Berkeley, 2016.
2. J. Matoušek, Lectures on Discrete Geometry. Springer, 2002.

3. R. O'Donnell, Analysis of Boolean Functions. Cambridge University Press, 2014.
4. L. Babai and P. Frankl, Linear Algebra Methods in Combinatorics. Department of Computer Science, University of Chicago, preliminary version, 1992.
5. J. Matoušek, Thirty-Three Miniatures (Mathematical and Algorithmic Applications of Linear Algebra). American Mathematical Society, 2010.

Secure Multiparty Computations

(a) (i) Definitions of multiparty computation (MPC) security (simulators)
  (ii) Secret Sharing
  (iii) Oblivious Transfer
  (iv) The Goldreich, Micali, and Wigderson (GMW) Protocol
  (v) Garbled Circuits
  (vi) MPC Toolchains
  (vii) Malicious security
  (viii) Oblivious RAM
  (ix) Some advanced topics from the following (but not limited to): Function secret sharing; Private Set Intersection; Private Information Retrieval;Universal Circuits; Oblivious Transfer Extension; Garbled RAM

(b) **References:**

1. David Evans, Vladimir Kolesnikov, and Mike Rosulek: A Pragmatic Introduction to Secure Multi-Party Computation, Ingram 2019.
2. Carmit Hazay and Yehuda Lindell: Efficient Two-party Protocols- Techniques and Constructions, Springer.
3. Ronald Cramer, Ivan Damgaard and Jesper Buus Nielsen: Secure Multiparty Computation and Secret Sharing - An Information Theoretic Approach, Cambridge Press.
4. Recent relevant papers

Topics in Cryptology

(a) In-depth study of some contemporary and emerging areas in cryptology. The list will be updated from time to time. Topics may include, but are not limited to, the following.

   i. Algorithms for factoring and discrete logarithms,
   ii. Authenticated encryption,
   iii. Tweakable Enciphering schemes,
   iv. Hash-based Cryptography,
   v. Isogeny-based Cryptography,
   vi. Multivariate Cryptography,
   vii. Homomorphic Encryption,
   viii. Secure Multiparty Computation,

(b) **Prerequisites:** Cryptology I

(c) **References:**

   1. Cynthia Dwork, Aaron Roth: The Algorithmic Foundations of Differential Privacy (Foundations and Trends in Theoretical Computer Science).
   2. Jonathan Katz and Yehuda Lindell: Introduction to Modern Cryptography, Second Edition, CRC Press.
   3. Rafael Pass and Abi Shelat, A Course in Cryptography, Lecture notes. Available online: https://www.cs.cornell.edu/courses/cs4830/2010fa/lecnotes.pdf
   4. Internet Resources and research papers.

Topics in Privacy

(a) Cryptographic foundations: Homomorphic Encryption, group signatures, blind signatures, anonymous credential management, commitment schemes, zero-knowledge proofs, proof of knowledge, SNARK, oblivious transfer, secure multiparty computation, Oblivious RAM, private set intersections, private information retrieval.

Perturbation, K-anonymity, L-diversity

Differential privacy

De-anonymization techniques

Privacy-preserving analytics

Applications: Mixnets, Onion Routing (TOR), e-cash, e-voting, location privacy, profiling, Web Privacy (Online tracking and advertising), Bitcoin, Zerocash etc.

Privacy for outsourced data

Privacy risk analysis

Ethical Aspects of privacy: Privacy compliance, GDPR, HIPAA etc.

(b) **Prerequisites:** None

(c) **References:**

1. C. Dwork and A. Roth, The Algorithmic Foundations of Differential Privacy. Foundations and Trends in Theoretical Computer Science.
2. J. Katz and Y. Lindell, Introduction to Modern Crypography. 2nd ed., CRC Press.
3. R. Pass and A. Shelat, A Course in Cryptography. Lecture notes. (Source: https://www.cs.cornell.edu/courses/cs4830/2010fa/lecnotes.pdf)

Quantum Computations and Cryptography

(a) **Introduction and Basics**

- Brief history about the emergence of quantum computation and cryptograhy.
- Brief review of fundamental concepts of the quantum theory: Axioms of Hilbert space quantum mechanics; indeterminism; interference; uncertainty; superposition; entanglement.
- Quantum bits; Reversible computation; Measurement; Schmidt decomposition
- Density operators; General quantum measurement(POVM); Evolution of quantum state, partial trace
- Quantum no-cloning theorem

Quantum Algorithms

- Bernstein–Vazirani algorithm
- Simon's algorithm

- Deutsch's algorithm
- Deutsch-Jozsa algorithm
- Quantum Fourier Transform and Quantum phase estimation
- Shor's factoring Algorithm
- Grover 's search algorithm

Quantum Cryptography

- Motivation: why quantum cryptography
- No-cloning theorem and its cryptographic implications
- Bell inequalities and CHSH game
- Entanglement in cryptographic tasks
- Security definitions in cryptography
- Classical vs. quantum adversaries
- Privacy amplification against quantum adversaries
- Quantum Key Distribution. Detail description of BB84 protocol

(b) **Prerequisites:** Data Structures and Algorithms, Design and Analysis of Algorithms, Digital Design and Computer Architecture

(c) **References:**

1. M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information. Cambridge University Press, 2010.
2. J. Watrous, The Theory of Quantum Information. Cambridge University Press, 2018.
3. T. Vidick, Introduction to Quantum Cryptography. Cambridge University Press, 2023.