

SEMESTER GUIDELINES FOR STUDENTS

Aim :To output Data Scientists,RPA Engineers,IOT-Robotic Programmers,Skilled Programmers, System Managers, System Analysts and Project Managers for the Twenty First Century.

FIRST SEMESTER

2020-21



Name : _____

Department of Computer Applications [M.C.A.]
SreeNarayana Institute of Technology
Vadakkevila P.O. Kollam 10.
www.snit.ac.in

**Guidelines MCA@SNiT
GENERAL DISCIPLINE**

The students should be in their respective classes before 09.25 AM, on all working days.

1. The computer lab and library will be open to the students from 08.45 AM to 04.55 PM. Internet should be used only for academic purpose. Absolute silence has to be maintained inside the computer lab and Library.
2. Female students are not permitted to stay in SNiT campus after office hours (4.45pm) under any circumstances without permission from the principal.
3. Students should be present in the class before faculty arrives and attendance will be taken at the onset of the lecture.
4. Students have to be compulsorily in uniform (*Boys - black pant and light blue shirt. Girls – black pant, light blue shirt and black overcoat*) on Mondays, Tuesdays, Thursdays, Fridays, University Examination dates and other occasions insisted by the Principal.
5. Students are expected to maintain decency and decorum in dress and behavior. Dress code to be followed by the students:
 - a. Men: They have to be formally dressed in pants and shirts, with their shirts tucked inside the pants and belted across. They have to wear the identity tag always inside the campus. They are not permitted to wear any other dress which could be classified as or resembles T Shirts, Dhoti, mundu etc.
 - b. Women: They have to be formally dressed in churidars, with the shawls pleated across and the hair well braided or clipped. They have to wear the identity tag always inside the campus. They are not permitted to wear any apparel that could be classified as or resembles T Shirts and skirts of any form inside the campus.
6. Students are not allowed to loiter in the veranda during class hours. They have to remain inside their respective class rooms, or in the library.
7. The students are not permitted to go out of the campus during working hours, and in case of an emergency, the permission from Principal/Head of the Department /Chief Advisors should be sought.
8. Application for leave, in the prescribed leave form duly signed by their parents is to be submitted to the respective chief advisors prior to availing leave.
9. Mobile phones are strictly banned inside the campus, if found the gadget (any communication device) will be seized by the department and penalised.
10. Ragging in any form is strictly prohibited inside and outside the campus.
11. Students indulging in Malpractices in the class test papers, internal theory/lab examinations, won't be promoted to the higher semesters.
12. Each student should be ready to accept any academic responsibility entrusted to him/her by the Head of the Department/Placement Officer/ Chief Advisor/Subject-in-charges.
13. Irregular attendance, insubordination to teacher, habitual inattention to class work, obscenity in word or act are sufficient reasons for the temporary / permanent dismissal of a student.

14. Attendance to all the seminars, workshops, invited talks, industrial visits, placement drives, occasions of importance to the College and Management, is compulsory.
15. All the co-curricular and extracurricular activities organized by the college are to be attended by all the students without fail.
16. Misuse of college computing and information facilities are liable to invite punishment and penalty under the Information Act 2005.
17. The college properties should be handled with maximum care and everything possible should be done to preserve the cleanliness and tidiness of the building, furniture, library and the premises. Penalty will be imposed on those who disobey.
18. Waste materials should be dropped in waste bin placed on each floors.

ATTENDANCE, LEAVE AND PROMOTION

1. All students are expected to be present in the college on all working days and attend the classes and academic activities in full. In case of any emergency situation or an illness, leave can be availed with the permission of the chief Advisor and H.O.D./Principal using the prescribed application form.
2. Application for leave is to be submitted prior to availing the leave and only in genuine cases, it will be considered within three days after the leave.
3. Students who avail leave in case of emergency without prior permission and go on leave for more than two days should inform the Chief Advisor with valid reasons.
4. When the application for leave is not granted or when leave is not applied for, the days of absence will be considered as unauthorized absence for which no credit can be availed for internal marks.
5. Once the leave is granted, the student should meet the faculty members handling the classes (within five days after availing the leave) and get the details of the leave marked in the respective attendance registers.
6. The actual attendance excluding days of leave should be within the stipulated minimum of 75% for each subject separately to appear for the End Semester Examination and promotion to higher semester.
7. The days of leave, when granted, will be normally considered only for awarding sessional marks.
8. When a student is having acute shortage of attendance (less than 75%) he can apply (to the University) for condonation of attendance, provided he is having sufficient number of days of leave already granted.
9. Students will be permitted to register and appear for successive semester examinations only if:
 - He or she has secured not less than 75% of the attendance in each semester.
 - His or her conduct is good.
 - Students will be permitted to continue subsequent semesters only if they have registered for the previous semester examination.

Any other details not specifically touched upon will be decided by the principal whose decisions shall be final.

New Delhi

NOTIFICATION**Dated 01-07-2009**

Sub: Prevention and prohibition of Ragging in technical Institutions, Universities including Deemed to be Universities imparting technical education.

F.No.37-3/Legal/AICTE/2009 – In exercise of the powers conferred under Section 23 read with Section 10 (b), (g), (p) and (q) of AICTE Act, 1987, the All India Council for Technical Education, hereby makes the following Regulations:-

1. Short title and commencement:-

- (i) These Regulations may be called the All India Council for Technical Education (Prevention and Prohibition of Ragging in Technical Institutions, Universities including Deemed to be Universities imparting technical education) Regulations 2009.
- (ii) They shall come into force on the date of the notification.

2. Objectives:-

In view of the directions of the Hon'ble Supreme Court in SLP No. 24295 of 2006 dated 16-05-2007 and in Civil Appeal number 887 of 2009, dated 08-05-2009 to prohibit, prevent and eliminate the scourge of ragging including any conduct by any student or students whether by words spoken or written or by an act which has the effect of teasing, treating or handling with rudeness a fresher or any other student, or indulging in rowdy or undisciplined activities by any student or students which causes or is likely to cause annoyance, hardship or psychological harm or to raise fear or apprehension thereof in any fresher or any other student or asking any student to do any act which such student will not in the ordinary course do and which has the effect of causing or generating a sense of shame, or torment or embarrassment so as to adversely affect the physique or psyche of such fresher or any other student, with or without an intent to derive a sadistic pleasure or showing off power, authority or superiority by a student over any fresher or any other student, in all higher education institutions in the country, and thereby, to provide for the healthy development, physically and psychologically, of all students, the All India Council for Technical Education,(AICTE) brings forth these Regulations.

3 What constitutes Ragging: - Ragging constitutes one or more of any of the following acts:

- a. A conduct by any student or students whether by words spoken or written or by an act which has the effect of teasing, treating or handling with rudeness a fresher or any other student.
- b. Indulging in rowdy or undisciplined activities by any student or students which causes or is likely to cause annoyance, hardship, physical or psychological harm or to raise fear or apprehension thereof in any fresher or any other student.

- c. Asking any student to do any act which such student will not in the ordinary course do and which has the effect of causing or generating a sense of shame, or torment or embarrassment so as to adversely affect the physique or psyche of such fresher or any other student.
- d. Any act by a senior student that prevents, disrupts or disturbs the regular academic activity of any other student or a fresher.
- e. Exploiting the services of a fresher or any other student for completing the academic tasks assigned to an individual or a group of students.
- f. Any act of financial extortion or forceful expenditure burden put on a fresher or any other student by students.
- g. Any act of physical abuse including all variants of it: sexual abuse, homosexual assaults, stripping, forcing obscene and lewd acts, gestures, causing bodily harm or any other danger to health or person.
- h. Any act or abuse by spoken words, emails, posts, public insults which would also include deriving perverted pleasure, vicarious or sadistic thrill from actively or passively participating in the discomfiture to fresher or any other student;
- i. Any act that affects the mental health and self-confidence of a fresher or any other student with or without an intent to derive a sadistic pleasure or showing-off power, authority or superiority by a student over any fresher or any other student.

4. Actions to be taken against students for indulging and abetting ragging in technical institutions Universities including Deemed to be University imparting technical education:-

1. The punishment to be meted out to the persons indulged in ragging has to be exemplary and justifiably harsh to act as a deterrent against recurrence of such incidents.
2. Every single incident of ragging a First Information Report (FIR) must be filed without exception by the institutional authorities with the local police authorities.
3. The Anti-Ragging Committee of the institution shall take an appropriate decision, with regard to punishment or otherwise, depending on the facts of each incident of ragging and nature and gravity of the incident of ragging.
4. a) Depending upon the nature and gravity of the offence as established the possible punishments for those found guilty of ragging at the institution level shall be any one or any combination of the following:-
 - (i) Cancellation of admission.
 - (ii) Suspension from attending classes.
 - (iii) Withholding/withdrawing scholarship/fellowship and other benefits.
 - (iv) Debarring from appearing in any test/examination or other evaluation process
 - (iv) Withholding results.
 - (v) Debarring from representing the institution in any regional, national or international meet, tournament, youth festival, etc.

- (vi) Suspension/expulsion from the hostel.
- (vii) Rustication from the institution for period ranging from 1 to 4 semesters.
- (viii) Expulsion from the institution and consequent debarring from admission to any other institution.
- (ix) Collective punishment: when the persons committing or abetting the crime of ragging are not identified, the institution shall resort to collective punishment as a deterrent to ensure community pressure on the potential raggers.

- b) An appeal against the order of punishment by the Anti-Ragging Committee shall lie,
 - i. In case of an order of an institution, affiliated to or constituent part, of the University, to the Vice-Chancellor of the University;
 - ii. In case of an order of a University, to its Chancellor.
 - iii. In case of an institution of national importance created by an Act of Parliament, to the Chairman or Chancellor of the institution, as the case may be.
- 5. The institutional authorities shall intimate the incidents of ragging occurred in their premises along with actions taken to the Council from time to time.

(Member Secretary)

Revised M.C.A Scheme with effect from 2020 admissions

Course Code	Name of Course	Faculty	Credits	Duration in hours			Marks		
				L	T	P	Sessional	Wn/Pract	Total
MCA 20C11	Mathematical Foundations for Computing	RDH	3	3	1	-	50	100	150
MCA 20C12	Advanced Operating Systems	SJ	3	3	1	-	50	100	150
MCA 20C13	Data Structures using Java	RDH MMT	3	3	1	-	50	100	150
MCA 20C14	Object Oriented Software Engineering	RPR	3	3	1	-	50	100	150
MCA 20C15	Theory of Computation	NLM	3	3	0	-	50	100	150
MCA 20P11	Lab 1 –Data Structures using JAVA	RDH	2		-	4	50	-	150
MCA 20P12	Lab 2 – UML	NLM	2		-	4	50	100	150
MCA 20G1 X	MOOC-1	MMT	1		-	3	50	100	50
	Total		20	15	4	11	400	700	1100

Class Committee

A class committee consists of teachers of the concerned class, student representatives and advisor constituted by Head of the Department/Principal and shall be held once in every month to inform the student about the nature and weightage of assessment, analyzing the performance of the student in the class after each test and finding the ways and means of improving the student performance. The aim of the class committee is to identify the weak students in any specific subjects and to request the teachers concerned to provide some additional help or guidance or coaching to such weak students as frequently as possible. Two or three subsequent meetings will be held at suitable intervals. During this meeting the student members, representing the entire class shall meaningfully interact and express the opinions and suggestions of the class to improve the effectiveness of teaching-learning process.

Class Committee dates:

Sl.No.	Dates
1	7/12/2020
2	4/1/2021
3	8/2/2021
4	8/3/2021

- University exam details (Theory and practical)**

Bridge Courses: Students who haven't undergone Computer Science as a core or complementary course for their undergraduate program has to study these courses in first semester itself and appear for the exam conducted by the University. A minimum pass in these courses are required for such candidate to register for the 3rd semester of the program and also for the publishing 1st & 2nd semester result. The student will get one chance in both the first and second semester of the program to attend the examination for the bridge courses. But, these marks will not be counted for finding class for the MCA programme. Institutions must ensure the conduct of the Bridge course suggested in the regular mode.

Course Code	Bridge Courses	Credits
MCA20B01	Principles of Programming	3
MCA20B02	Digital Logic and Computer Architecture	3
MCA20B03	Programming C Lab	2
	Total	8

Question Paper Pattern (Bridge courses): The maximum mark for the theory examinations will be 50 and the time duration will be 2 hours. The minimum pass mark will be 40% for individual courses and 50% for consolidated marks of 3 courses. The question paper shall contain 3 parts; Part-A, Part-B and Part-C. Part-A shall be for 10 marks and contains 10 compulsory questions (MCQ/One-word/etc.), of 1 mark, 2 questions from each module. Part-B shall be for 20 marks and shall contain 15 short answer questions, 3 questions of 2 marks from each module out of which 2 questions from each module is mandatory. Part-C shall be for 20 marks and shall contain 10 questions, 2 questions of 4 marks from each module out of which 1 question from each module is mandatory. The lab examination will be for 50 marks, conducted internally by the institution monitored by the head of the Institution/Department, evaluating the skill of candidate.

	One Module				Complete set(5modules) Question		
Section	No.of Questions	No.of questions to attend (mandatory)	Marks for each question	Total Marks	No.of Questions	No.of questions to attend (mandatory)	Total Marks
Part-A	2	2	1	2	10	10	10
Part-B	2	2	2	4	10	10	20
Part-B	2	1	4	4	10	5	20
Marks distributed for a module : 10					Total Marks : 50		

Pattern: The student has to take, generally, five theory papers and two practical courses in the first three semesters. MOOC courses are included in first and fourth semester. Seminar and case study is included in second and third semesters respectively. In fourth semester the student has to undergo a major project work. In each week a student is supposed to get 11 practical hours and hence in every semester a total of 176 hours of practical training in the laboratories. The total contact hours for theory / tutorial /practical comes to around 30 hours/week. The attendance in the theory & practical is compulsory.

Examinations: University Examinations will be conducted at the end of each semester as per the scheme included in this document. Pass Requirements and provisions for classification of successful candidates.

a) A candidate shall be declared to have passed the semester examination in full if the candidate secures not less than 40% marks in written examination and not less than 50% marks in written (University) plus sessional marks put together in each paper. This rule applies to practical also. For the courses which have only sessional marks, a minimum of 50% is required for a pass; otherwise the student has to repeat that semester.

b) For a pass in main project the student has to obtain minimum 50% marks in internal evaluation and 50% marks in external evaluation. Otherwise the candidate has to repeat the 4th semester.

c) If a student fails in one or more courses, he/she needs to reappear only in those courses. The rules for supplementary examinations will be same as that of the existing regulations.

d) Classification of (Pass) results into Distinction, I-Class, II-Class, etc. shall be as per the scheme prior to 2020 admissions.

e) Sessional Marks: The sessional marks are awarded based on 2 class tests and assignments/lab reports for theory / practical. Split up is shown below:

Theory

Assignments (minimum2)	Class tests (minimum2)
50%	50%

Practical:

Performance in the lab (Lab reports and experiments)	Lab tests (minimum2)
50%	50%

Main Project

Topic	Performance	Evaluation
10%	40%	50%

For Seminars, the sessional marks are based on presentation/seminar report and participation. The students are required to present the progress of the main project work twice to the Department Faculty.

Question Paper Pattern (Other than Bridge courses): The maximum mark for the theory examinations will be 100 and the time duration will be 3 hours. The question paper shall contain two parts; Part-A and Part-B. Part-A shall be for 40 marks and shall contain 10 compulsory short answer questions, 2 questions from each

module with 4 marks. Part B shall be for 60 marks and shall contain 15 questions, 3 questions from each module out of which the student has to answer 2 questions from each module and has 6 marks for each question.

	One Module				Complete Question set(5modules)		
Section	No.of Questions	No.of questions to attend (mandatory)	Marks for each question	Total Marks	No.of Questions	No.of questions to attend (mandatory)	Total Marks
Part-A	2	2	4	8	10	10	40
Part-B	3	2	6	12	15	10	60
Marks distributed for a module : 20					Total Marks : 100		

Assignments

[Submit the assignments within 10 days from the date of issue]

First Assignment

Subject	Date of Issue (On or Before)
MFC	23/11/2020
AOS	30/11/2020
DSJ	7/12/2020
OOSE	14/12/2020
TOC	21/12/2020

Second Assignment

Subject	Date of Issue (On or Before)
MFC	11/1/2021
AOS	18/1/2021
DSJ	25/1/2021
OOSE	1/2/2021
TOC	8/2/2021

* **Class tests (Theory)**

50% -> 25 marks.

(Series 1 => 5 marks, Series 2 => 10 marks, Model => 10 marks)

Two series examinations and a model exam will be conducted to enhance student's confidence level to approach University exam. 80% of questions will be from question bank provided in the end of guidelines.

First Series Examination

<i>Subject</i>	<i>Date of Examination</i>
MFC	30/11/2020
AOS	7/12/2020
DSJ	14/12/2020
OOSE	21/12/2020
TOC	4/1/2021
JAVA LAB	7/1/2021
UML LAB	8/1/2021

Second Series Examination

<i>Subject</i>	<i>Date of Examination</i>
MFC	18/1/2021
AOS	25/1/2021
DSJ	1/2/2021
OOSE	8/2/2021
TOC	16/2/2021
JAVA LAB	19/2/2021
UML LAB	20/2/2021

Model Examination

<i>Subject</i>	<i>Date of Examination</i>
MFC	8/3/2021
AOS	10/3/2021
DSJ	12/3/2021
OOSE	15/3/2021
TOC	18/3/2021
JAVA LAB	22/3/2021
UML LAB	23/3/2021

Practical

1. Performance in the lab assignments
(Rough and Fair Records) 50% -> 25 marks.
2. Lab tests 50% -> 25 marks.

Lab Fair Records must be certified (Staff-in-Charge, HOD certification and Office seal) and submitted for Model Lab examination. Those who fail to certify the records will not be permitted to attend model lab exam.

Semester I P.T.A meeting

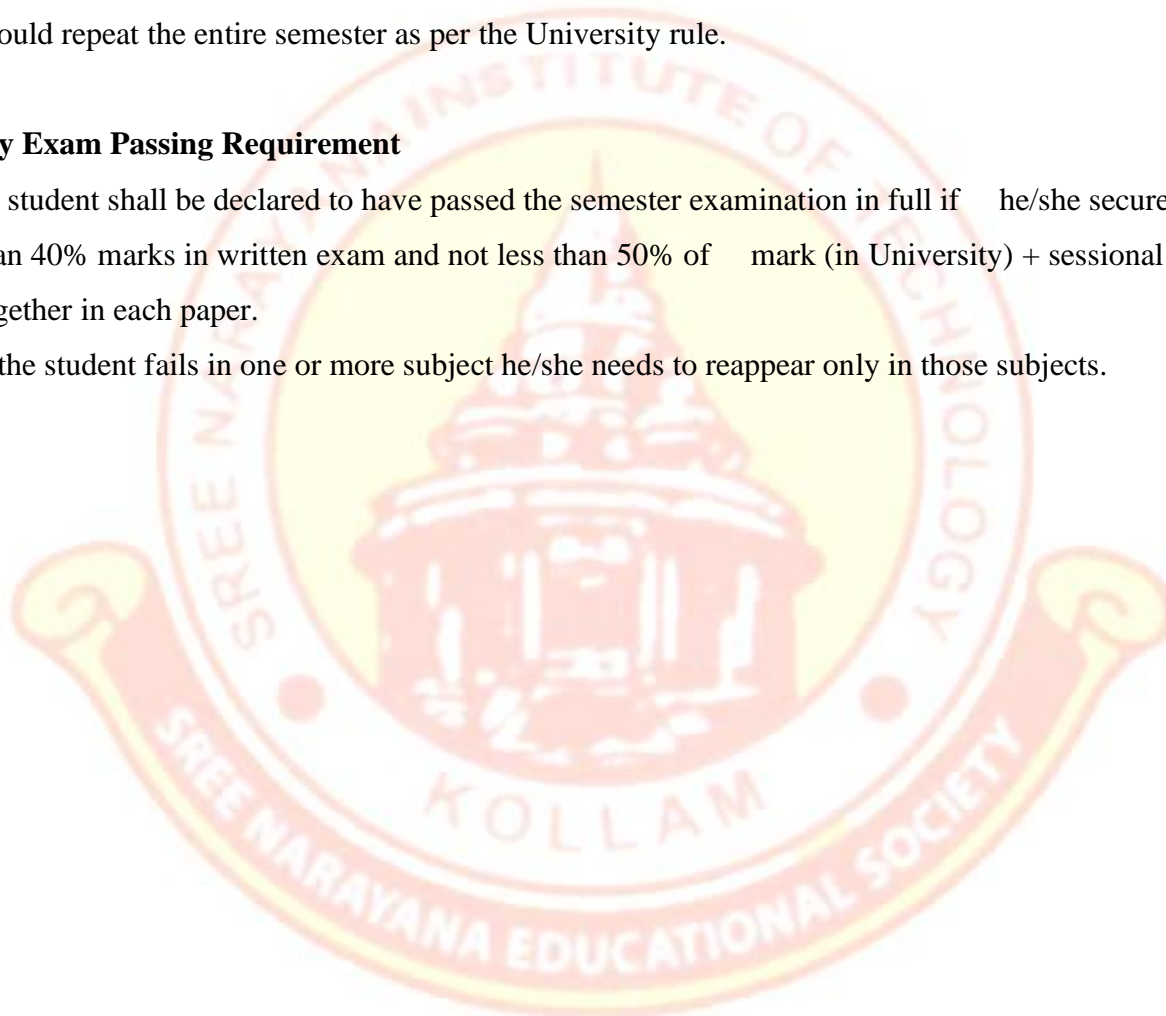
January 15th 2021

Eligibility for writing University Examination

1. A student have to maintain minimum 75% of attendance
2. If a student has more than 65% and less than 75% of attendance, he/she should apply for condonation as per University rules.
3. If the student is having less than 65% of attendance , he/she cannot write University examination and should repeat the entire semester as per the University rule.

University Exam Passing Requirement

1. A student shall be declared to have passed the semester examination in full if he/she secures not less than 40% marks in written exam and not less than 50% of mark (in University) + sessional mark put together in each paper.
2. If the student fails in one or more subject he/she needs to reappear only in those subjects.



SYLLABUS**Course Code: MCA20C11****Credits: 3****MATHEMATICAL FOUNDATIONS OF COMPUTER SCIENCE****COURSE OUTCOMES**

CO1	Compute a generating function and apply them to combinatorial problems
CO2	Apply the inclusion/exclusion principle
CO3	Solve linear algebra problems with linear equations, matrix calculus and vectors
CO4	Discuss the usage of geometric transformations.
CO5	Illustrate different decomposition methods used in linear system of equations
CO6	Explain unconstrained optimization and Linear Programming Problems

COURSE CONTENT

MODULE I: Elementary Combinatorics: Basis of counting, Combinations & Permutations, with repetitions, Constrained repetitions, Binomial Coefficients, Binomial Multinomial theorems, the principles of Inclusion - Exclusion. Pigeon hole principles and its application.

MODULE II: Linear algebra: Matrices, vectors and determinants, Eigen values, Eigen vectors, Eigen value problems, vector differential calculus-Inner product, cross product, gradient of a scalar field, divergence of a vector field and curl of a vector field.

MODULE III: Geometric transformations-Translations, Rotation around the origin, Rigid motions and homogeneous representations, Affine transformations, Coordinate Transformation on Image Arrays.

MODULE IV: Numeric Analysis: Introduction, solution of equations by iteration, numeric linear algebra-Linear Systems: Gauss Elimination, LU factorization, matrix inversion, Least squares method.

MODULE V: Optimization: Basic concepts, Unconstrained Optimization-method of Steepest Descent, Linear Programming-Normal, pivotal reduction of a general system of equations, simplex method.

REFERENCES

_ Discrete mathematics for computer scientists & mathematicians JL Mott, A Kandel, TP Baker PHI.

_ Discrete Mathematical structures Theory and application-Malik & Sen, Course Technology, 2004

_ Ernest Davis, Linear Algebra and Probability for Computer Science Applications, CRC Press, 978-1-4665-0159-1

_ Erwin Kreyszig, Advanced Engineering Mathematics (10th Edition), 2011 John Wiley & Sons, ISBN-13: 978-0-571-72897-9

_ Michael Baron, Probability and statistics For computer scientists (2nd edition), Chapman and Hall/CRC, ISBN 978-0-570-55836-5

Course Code: MCA20C12**Credits: 3****ADVANCED OPERATING SYSTEMS****COURSE OUTCOMES**

CO1	Explore the fundamental concepts of different models of Operating systems
CO2	Understand the design and functions of operating systems.
CO3	Illustrate different process scheduling models in Operating Systems
CO4	Differentiate different types of OS and its significance.
CO5	Understand the file and memory management in UNIX Operating Systems.
CO6	Understand the need and importance of Parallel systems.
CO7	Understand the working of Distributed Systems.
CO8	Compare Real Time OS, Mobile OS and Multi-processor OS.

COURSE CONTENT

MODULE I: Functions of operating systems, Design approaches: layered, kernel based and virtual machine approach, types of advanced operating systems (NOS, DOS, Multiprocessor OS, Mobile OS, RTOS, Cloud OS)

MODULE II: *File Management*- System Structure, User Perspective, Architecture - Buffer cache - File Representation: Structure of file Directories. *Memory Management*- Detailed design of Process Structure: Kernel Data structures for process. Context of a Process: Static and Dynamic. *Parallel Systems and computing*- Shared memory machines, Synchronization, Communication, Shared memory multiprocessor OS.

MODULE III: *Distributed Operating system concepts*- Goals, Distributed Computing Models, Hardware and Software Concepts, Architecture of DOS. Design Issues, Distributed communication, shared memory, synchronization - Distributed Object based System.

MODULE IV: *Multiprocessor Operating System*- Introduction, Basic multiprocessor system architectures, design issues, Threads, Process synchronization: the test and set instruction, the swap instruction, implementation of the process wait. Processor scheduling: Issues, Coscheduling, Smart scheduling, Affinity Based scheduling

MODULE V: *Real Time OS*- Characteristics of Real Time operating Systems, Classification of Real Time Operating Systems, Scheduling in RTOS: Clock driven: cyclic, Event driven: EDF and rate monotonic scheduling. *Mobile OS*- Architecture, Android OS, iOS, Virtual OS, Cloud OS and their design issues.

REFERENCES

- _ Bhatt P.C.P., An Introduction to Operating Systems: Concepts and Practice, 3/e, Prentice Hall of India, 2010.
- _ William Stallings, Operating Systems: Internals and Design Principles, Pearson, Global Edition, 2015.
- _ Mukesh Singhal and Niranjana Shivarathri, Advanced Concepts in Operating Systems, McGraw-Hill Series in Computer Science, 1993.
- _ Andrew S. Tanenbaum, Distributed Operating Systems, Addison – Wesley, 1998
- _ Jean Bacon, Concurrent Systems, Addison – Wesley, 1998
- _ William Stallings, Operating Systems, Prentice Hall, 1995. _ Pradeep K Sinha, “Distributed Operating Systems: Concepts and design”, PHI, 2007.

Course Code: MCA20C13**Credits: 3****DATA STRUCTURES USING JAVA****COURSE OUTCOMES**

CO1	List the advantages of the object oriented programming approach
CO2	Explore the basic programming concept of Java
CO3	Understand the advanced programming capabilities of Java
CO4	Classify the types of data Structures
CO5	Explain the algorithms associated with each type of data structures
CO6	Identify the appropriate data structures to solve different problems
CO7	Implement the algorithms using the advanced features of Java

COURSE CONTENT

MODULE I: Basic Concepts of Java: Object Oriented Programming, Features of Java, Classes, Interfaces, Constructors and Finalizers, Packages, Exception Handling, API.

MODULE II: Advanced Concepts in Java: Multithreading, Event Handling, Applets, Graphics, Text, AWT Controls, Layout Manager and Menus. I/O Streams, File Class and Operations.

MODULE III: Data Structures: Introduction, Abstract Data Types, Linear and Non Linear Data Structures, Searching: Linear Search, Binary Search, Implementation in Java, Sorting: Insertion Sort, Bubble sort, Selection Sort, Merge Sort, Quick Sort, Implementation in Java, Stacks: Representation, Implementation of Stack Operations in Java, Queues: Representation, Implementation of Queue Operations in Java.

MODULE IV: Linked Lists: Representations, Linear Linked List, Doubly Linked List, Circular Linked List, Implementation of operations on Linked Lists in Java, Hashing : Hash functions, Hash Tables, Chaining.

MODULE V: Trees: Representation, Operations on Trees, BST, AVL Trees, Red-Black Trees, B-Trees, Implementation of Trees in Java, Graphs: Representation, Operations on Graphs, Spanning Trees, implementation of Graphs in Java.

REFERENCES**Text books**

1. John Hubbard, Data Structures with Java, 2ed (Schaum's Outlines)
2. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, "Introduction to Algorithms", MIT Press
3. Paul J. Deitel and Harvey Deitel , Java How to Program
4. Schildt Herbert, Java: The Complete Reference
5. Michael T. Goodrich, Data Structures and Algorithms in Java

Additional and Web –Resources

1. https://onlinecourses.nptel.ac.in/noc20_cs85/preview
2. <https://enos.itcollege.ee/~jpoial/algorithms/GT/Data%20Structures%20and%20Algorithms%20in%20Java%20Fourth%20Edition.pdf>

Course Code: MCA20C14**Credits: 3****OBJECT ORIENTED SOFTWARE ENGINEERING**

CO1	Discuss about the steps in Unified Approach.
CO2	Draw UML diagrams including class diagram, activity diagram, use case diagram and sequence diagram for a given problem statement
CO3	Illustrate the steps in Object Oriented Analysis and Object Oriented Design.
CO4	Explain the principles and practices in Agile Software development methodology.
CO5	Compare software quality assurance techniques.
CO6	State Myer's principle for debugging.
CO7	Design an object oriented system to solve any real life problem.

COURSE CONTENT

MODULE I: Introduction – Object Oriented Systems development life cycle. Object oriented Methodologies- Booch's Methodology - Rumbaugh's Methodology, Jacobson's Methodology- Patterns and Frameworks.

MODULE II: Fundamentals of Object Oriented design using Unified Modeling Language, UML- Use case diagram- Class diagram- Sequence diagram- collaboration diagram-State chart diagram- Activity diagram- Component diagram- deployment diagram.

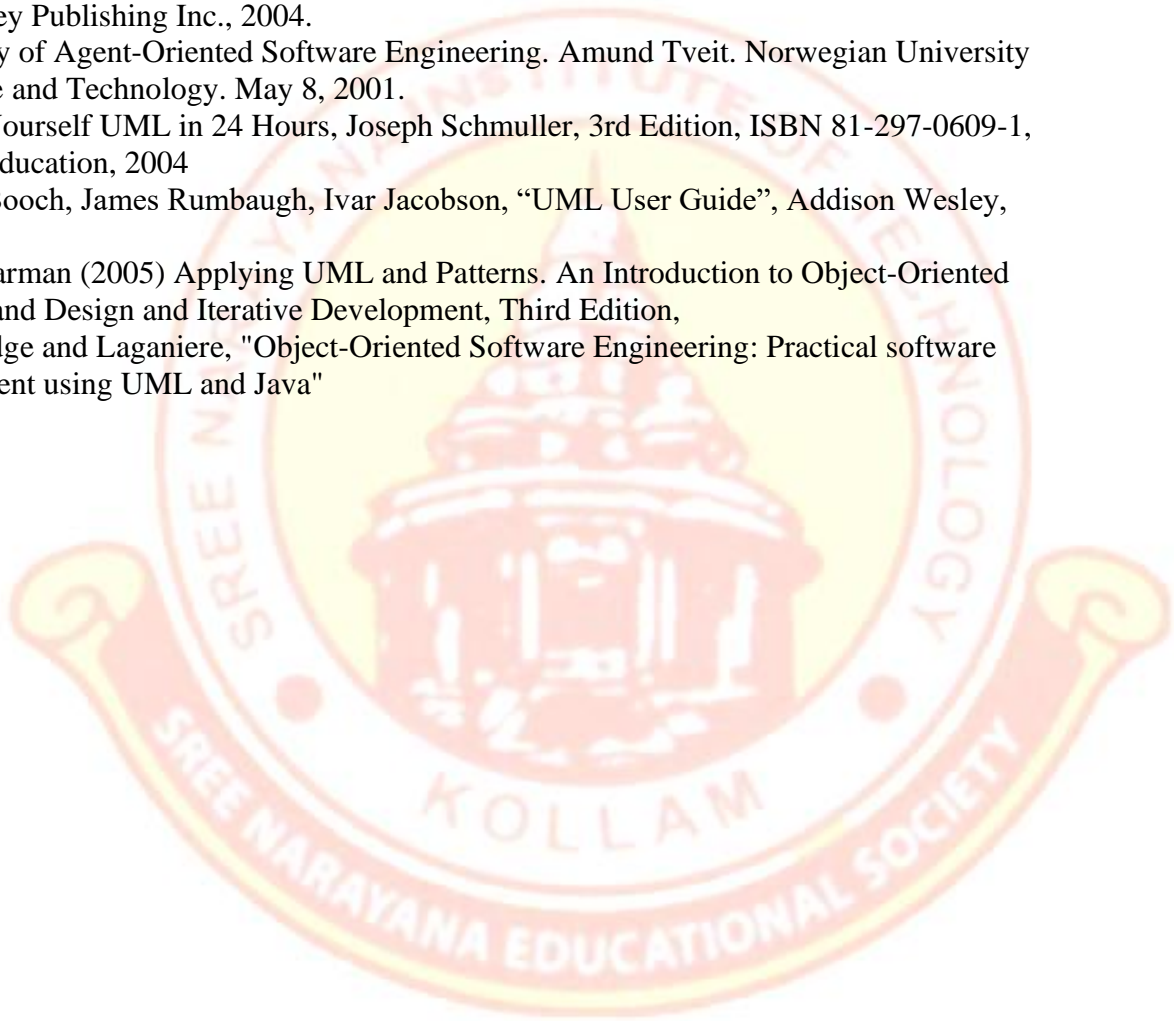
MODULE III: Object oriented analysis: Use Case Model- Identifying use cases identifying actors- Documentation- Object analysis - Classification-different approaches- Identifying Classes- Identifying Object relationships – Attributes and Methods.

MODULE IV: Object oriented design: Design process- Design axioms - Corollaries- Design Patterns- Designing Classes - Designing Protocols and Class visibility - Defining Attributes- Designing Methods-Guidelines for identifying bad design.

MODULE V: Agile Software Development - Agile Practices & Principles- Software Quality Assurance- Bugs and Debugging- Testing Strategies- Developing test cases- Developing test plans- Debugging Principles- Introduction to Agent Oriented Software Engineering.

LEARNING RESOURCES**REFERENCES****TEXT BOOKS**

- _ Ali Bahrami, Object Oriented Systems Development, Tata McGraw-Hill, 1999
- _ Martin Fowler, UML Distilled, Second Edition, PHI/Pearson Education, 2002.
- _ James Rumbaugh, Ivar Jacobson, Grady Booch, The Unified Modelling Language Reference Manual, Addison Wesley, 1999.
- _ Stephen R. Schach, Introduction to Object Oriented Analysis and Design, Tata McGraw-Hill, 2003.
- _ Hans-Erik Eriksson, Magnus Penker, Brain Lyons, David Fado, UML Toolkit, OMG Press Wiley Publishing Inc., 2004.
- _ A survey of Agent-Oriented Software Engineering. Amund Tveit. Norwegian University of Science and Technology. May 8, 2001.
- _ Teach Yourself UML in 24 Hours, Joseph Schmuller, 3rd Edition, ISBN 81-297-0609-1, Pearson Education, 2004
- _ Grady Booch, James Rumbaugh, Ivar Jacobson, "UML User Guide", Addison Wesley, 2002.
- _ Craig Larman (2005) Applying UML and Patterns. An Introduction to Object-Oriented Analysis and Design and Iterative Development, Third Edition,
- _ Lethbridge and Laganriere, "Object-Oriented Software Engineering: Practical software development using UML and Java"



Course Code: MCA20C15**Credits: 3****THEORY OF COMPUTATION****COURSE OUTCOMES**

CO1	Exploring the basics of Language
CO2	Construction of Deterministic & Non Deterministic Automata
CO3	Exploring Grammars – Ambiguity - Regular Expressions & Regular Grammars
CO4	Discussing Context Free Languages & Greibach forms
CO5	Discussing Pumping Lemma & construction of Push down automata
CO6	Exploring various types of Turing Machines – Mealy & Moore machines
CO7	Discuss Decidability & Halting Problem

COURSE CONTENT

MODULE I : Concepts of Automata Theory : Strings, Alphabet, Language- Finite state machine, definitions, finite automaton model, acceptance of strings, and languages, Finite Automata: NFA , DFA , Finite Automata with Epsilon transitions NFA to DFA conversion, minimisation of DFA, Finite Automata with Epsilon Transition- Finite Automata with output- Moore and Mealy machines.

MODULE II: Regular Expressions & Languages: Regular expressions, Finite Automata & Regular Expression operations, Conversion of Finite Automata to Regular expressions. Converting Regular Expressions to Automata - Algebraic Laws for Regular Expressions. Pumping Lemma for Regular Languages, Application of Regular Expressions

MODULE III: Context free grammar: Derivation trees, and sentential forms. Parse Trees - Ambiguity in Grammars, Chomsky normal form, Greibach normal form, Pumping Lemma for Context Free Languages. Enumeration of properties of Context Free Language, PDA ,Acceptance of Context Free Language, Deterministic PDA ,Application of CFG

MODULE IV: Turing Machines: Definition , Transition diagram, Design & Roles of Turing machine, Church-Turing Thesis, Modular Construction of complex Turing machines, Types of Turing machines Extensions of Turing machines, Non-Deterministic Turing Machines. Restricted Turing Machines

MODULE V

Complexity Theory: Intractable Problems: Definition of P and NP problems, NP complete and NP hard problems. Decidability and Undecidability - Decidability & Halting problems - Undecidability and Reducibility in TOC,

MASTER OF COMPUTER APPLICATION

REFERENCES**Text Books**

1. John E Hopcroft ,Rajeev Motwani and Jeffrey D Ullman “ Introduction to Automata Theory, Languages, and Computation “, Third Edition Pearson
2. Michael Sipser “ Introduction to the Theory of Computation “ Cengage Publishers ,2003
3. John C Martin , Introduction to Languages and the Theory of Computation , TMH 2013

Reference Text :

1. Dexter C Kozen , Automata and Computability , Springer 1999

Course Code: MCA20P11

Credits: 2

LAB 1 – DATA STRUCTURES USING JAVA**COURSE OUTCOMES**

CO1	Acquire knowledge in the object oriented programming concept
CO2	Design the algorithms to match with the given problem specifications
CO3	Implement linear and non linear data structures using the advanced features of Java
CO4	Familiarize the debugging concept in Java Code
CO5	Formulate real world applications with the help of appropriate algorithms
CO6	Generate different test cases for testing the validity of the developed programs
CO7	Write technical report based on the results of the experiments

COURSE CONTENT**List of Experiments:**

1. Searching : Implement Linear and binary search
2. Sorting: Implement Insertion Sort, Bubble sort, Selection Sort, Merge Sort, Quick Sort
3. Stacks: Perform stack operations
4. Applications of Stacks: Perform evaluation of expressions
5. Queues: Perform queue operations.
6. Application of queues: Simulate a queue in a real life situation
7. Linked List : Implementation of linked lists
8. Hashing : Apply hashing functions for searching
9. Trees: Familiarize the different operations on Trees
10. BST : Implement a BST
11. AVL Trees: Implement an AVL tree
12. Red-Black Trees: Implement Red-Black Trees
13. B-Trees: Generate B-Trees and perform the operations
14. Graphs – Perform the graph traversals
15. Spanning Trees : Implementation of Spanning trees

REFERENCES**Text books**

1. Lab manual for Data structures through Java, V V Muniswami
2. Java Lab Manual by Madhu Mathi
3. Data Structures in Java A laboratory Course, Sandra Andersen

Additional and Web -Resources

1. <https://www.udemy.com/course/data-structures-and-algorithms-in-java/>
2. https://onlinecourses.nptel.ac.in/noc20_cs85/preview
3. <https://enos.itcollege.ee/~jpoial/algorithms/GT/Data%20Structures%20and%20Algorithm%20in%20Java%20Fourth%20Edition.pdf>

Course Code: MCA20P12**Credits: 2****LAB 2 – UNIFIED MODELLING LANGUAGE (UML)****COURSE OUTCOMES**

CO1	Implement basic SQL commands including creation of database, tables and insertion, updation, deletion and selection of table contents.
CO2	Perform joining of tables and implementing nested queries.
CO3	Draw UML diagrams including class diagram, use case diagram, activity diagram, collaboration diagram, sequence diagram and state chart diagram.

COURSE CONTENT

Lab exercises related with the following should be implemented in this course.

1. Implementing UML diagrams

- Class diagram
- Use case diagram
- Activity diagram
- State chart diagram
- Sequence diagram
- Collaboration diagram

Sample Problem Statement

There is a need to develop software for tracking the library loan records. Library patrons may borrow books, magazines, compact discs and audio tapes. A library must manage each copy of a library item. For eg., a library may have five copies of the book “Wings of the Fire”. Each borrower has library card. Each type of library item has a check out period and a maximum number of renewals. For example the children’s books may be checked out for a month while books for the adults may be checked only for two weeks. Ordinary books can be renewed once; books with a pending request may not be renewed. The system must record the actual return date and any fine that was paid. Draw the UML diagrams for the given problem statement.

- Draw Class Diagram
- Draw Use Case diagram
- Draw Activity diagram
- Draw Sequence Diagram

Course Code: MCA20G1X**Credits: 1****MOOC 1**

Each student has to successfully complete one MOOC course in the first semester from the topics related to (Communicative English, Research Methodology, Technical Writing, Entrepreneurship, Environmental studies, Cyber law or any new topic/technology introduced recently. etc.) with a minimum period of eight (08) weeks offered through E-learning platforms like SWAYAM, Coursera, etc..

BRIDGE COURSES**Semester I****Course Code: MCA20B01****Credits: 3****PRINCIPLES OF PROGRAMMING COURSE OUTCOMES****COURSE OUTCOMES**

CO1	Awareness about programming paradigms, programming environments ,need of debugging
CO2	Overview of functional programming languages and its evolution
CO3	Overview of Logical programming languages and its evolution
CO4	Introduction to syntax and semantics.
CO5	Awareness about Types of Grammars and parsing
CO6	Introduction to Scripting languages
CO7	Introduction to Object oriented programming language, OOP concepts using different languages

COURSE CONTENT

MODULE I: Introduction to Programming Paradigms, Characteristics of Programming Languages, Programming language paradigms: Imperative, Object-Oriented, Functional, Logic, Event Driven and Concurrent Programming, Programming environments: Compilers, Interpreters, Interactive development tools, Debugging tools.

MODULE II: Functional Programming languages – LISP, Language overview, processes in functional programming, evaluation. Two descendants of LISP- SCHEME Language, Common LISP .Meta Language (ML), Haskell Language (basics only). Overview and Evaluation of ALGOL 60, COBOL, BASIC, PL/I. Early Dynamic languages-APL, SNOBOL, Origin and Characteristics. Beginning of Data Abstraction – SIMULA, overview. Early Descendents of ALGOLs-PASCAL, C, Overview and Evaluation. **MODULE III:** Logical Programming language – PROLOG, Ada, Overview and Evaluation. Extended Logic Programming, Object Oriented Programming – Smalltalk, Language overview and Evaluation. Combining Imperative and Object Oriented features-C++, Overview, evaluation, Related languages-Objective C, Delphi, Go (Basics only). Imperative Based Object Oriented language –Java, Overview, Evaluation. Scripting languages-Origin and characteristics of Perl, JavaScript, PHP, Python ..NET languages-C#, Overview, Evaluation. Mark-up/Hybrid Languages-XSLT, JSP (Basics only).

MODULE IV: Describing Syntax and semantics-Language Recognizers, Language Generators, Grammars and Recognizers-Attribute Grammars, Dynamic semantics, Axiomatic Semantics. Lexical and syntax analysis-Parsing , Top down and bottom up parsers(Basics only).Names ,Bindings and scopes-Names, variables, Concept of binding-static and dynamic, scopes-static and dynamic. **MODULE V:** Scripting Language- key Concepts, Imperative programming languages-variables, type checking, scope, data type, arithmetic expressions, control flow, sub programs. Object Oriented Programming languages-C++ and its support for OOP, Java and OOPS,C # and OOPS, Python and OOPS, Event handling in Java.

REFERENCES

- Text books** ● R. Sebesta, Concepts of Programming Languages, Addison Wesley
- Principles of Programming languages – A Paradigm Approach – Syed Buhari, McGrawHill,
 - Debashish Jana, Java and Object-Oriented Programming Paradigm, Prentice Hall., 2008.
 - Van Roy, Haridi, Concepts, Techniques and Models of Computer Programming, MIT Press,2004.
 - Robert Lafore, Object-Oriented Programming with C++ ● Bruce Eckel , Thinking in Java
 - J. Reynolds, Theories of Programming Languages, Cabridge University Press.
 - Duckett Jon, Beginning Web Programming with HTML, XHTML and CSS, Wrox, 2004.

Semester I

Course Code: MCA20B02

Credits: 3

DIGITAL LOGIC AND COMPUTER ARCHITECTURE**COURSE OUTCOMES**

CO1	Able to understand of Digital logic fundamentals and computer architecture
CO2	Identify the organization of computer memory and peripherals
CO3	Explain the data transfer and control mechanisms in digital computers
CO4	Summarise the computer arithmetic and processor organization
CO5	Evaluate the performance of processors through parallel processing

COURSE CONTENT

MODULE I: Digital computers, Logic gates, Combinational logic , circuits: Adders, Subtractors, Encoders, Decoders, Multiplexers, Demultiplexers. Sequential circuits Flipflops, Counters, Shift Registers, Number representation and arithmetic operations, Digital codes, Error detection and correction.

MODULE II: Memory organization: Memory hierarchy – Main memory – Auxiliary memory – Associative memory – Cache memory – Virtual memory, Input-output organization: Peripheral devices – I/O interface – Asynchronous data transfer-DMA-Input/output processor(IOP)

MODULE III: CPU: Register and stack organization – Instruction formats – Addressing modes – Data transfer and manipulation – Program control – RISC Control Unit: Control Signals Control memory- Hardwired Control -Microprogrammed Control

MODULE IV: Pipelining – Arithmetic and instruction pipeline – RISC pipeline –vector processing- Array Processors, Parallel Processing and Performance: Hardware Multithreading

MODULE V: Vector (SIMD) Processing - Graphics Processing Units, Shared-Memory Multiprocessors, Cache Coherence - Write-Through Protocol, Write-Back protocol, Snoopy Caches, Directory-Based Cache Coherence , Message-Passing Multicomputer.

LEARNING RESOURCES

Text books:

1. M. Morris Mano, "Computer System Architecture", Pearson Education .
2. John P Hayes, Computer Architecture and Organization, McGraw-Hill Book Company.
3. Thomas.L.Floyd, "Digital Fundamentals", Pearson Education.

REFERENCES

1. Computer organization And Embedded Systems, Hamacher, Vranesic, Zaky, Manjikian, McGraw-Hill.
2. Manish Saraswat, 'Computer Architecture And Organisation', Vayu Education Of India.
3. Tanenbaum A.S, 'Structured Computer Organization', Prentice Hall of India
4. Malvino A. P. and Donald P. Leach, "Digital Principles and Applications", McGraw Hill Publications.

Semester I
PROGRAMMING C LAB
COURSE OUTCOMES

Course Code: MCA20B03

Credits: 2

CO1	Awareness about Compilation and Execution of programs, Pre-processor commands, syntax, data types, storage class, built in functions in C, Working with Formatted Input and Output.
CO2	Develop program skills to implement different operators in C–Arithmetic, Logical, Assignment, bit wise, sizeof () and conditional operator.
CO3	Develop program skills for decision making- using if, if...else, nested if, Conditional operator, switch, and skills for using loop structures, nested loops, and loop control statements-break, continue
CO4	Develop C program for defining and calling function, Concept of call by value, call by reference, Demonstrate the difference between iteration and recursion in terms of C programming
CO5	Implementation of arrays, structures & pointers
CO6	String handling and File Management concepts

COURSE CONTENT

MODULE I : Introduction to Pre-processor commands, syntax, comments, built in functions, Working with formatted input and output. Small programs implementing basic data types – integer, float, char, variables and constants. Program using storage class –static; Operators in C - arithmetic operators, Logical operators-AND, OR, NOT, assignment operators bit wise operators, conditional operator and sizeof() operator.

MODULE II: Decision making in C using -if, if ...else, nested if statements. switch...case statement, entry and exit control loops structures-for loop, while loop, do...while loop; Nested loops; Loop control statements – break, continue and goto.

MODULE III: Functions in C- built in and user defined functions-Local and global variables; actual and formal parameters; simple example of declaring and using functions with (i) no argument, no return (ii) with argument, no return,(iii) no argument, with return,(iv) with argument, with return; function calls, call by value and call by reference; function recursion.

MODULE IV: Arrays in C – One dimensional and two dimensional arrays. Implement pointers- pointers to arrays and array of pointers - (Programs to demonstrate implementation of 1 D and 2 D arrays using pointers).

MODULE V: Concept of header files, C program to implement basic string and mathematical functions, Structures -accessing structure members, pointer to structure, File handling programs in C – open, read and write operations in file.

REFERENCES

Text books • Ashok N.Kamthene, Programming in C, Pearson Education, Third edition.

Additional and Web –Resources • E Balaguruswamy , Programming in ANSI C, Mc Graw hill, Eighth Edition.

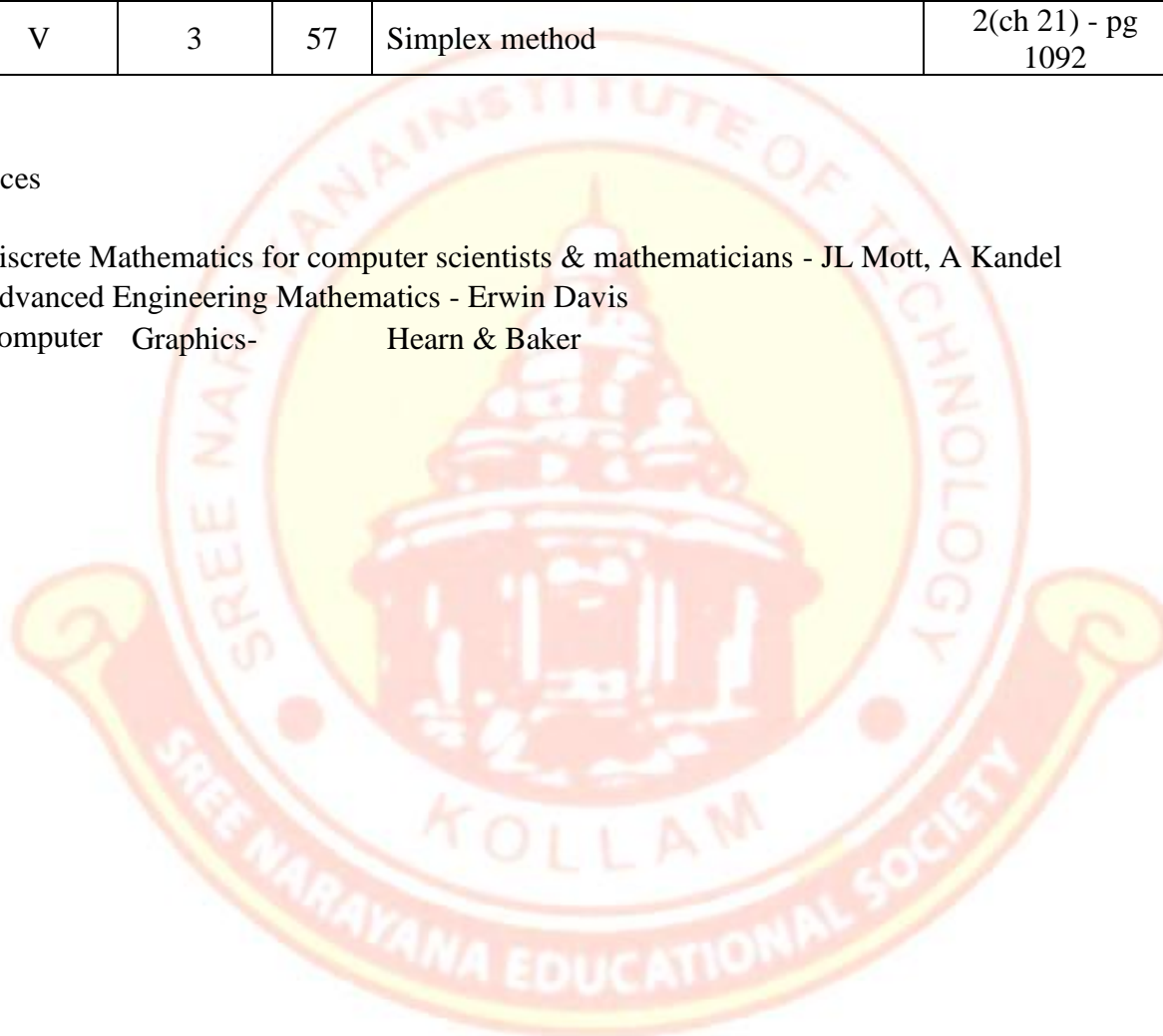
DETAILED LECTURE PLANS**Mathematical Foundations for Computing**

Sl. No.	Module No.	No: of hours	Cum. No. of hours	Topic	Source
1	I	1	1	Basis of counting	1(ch 2) - Pg 126
2	I	4	5	Combinations and permutation	1(ch 2) - pg 143
3	I	2	7	with repetition, Constrained repetition	1(ch 2) - pg 162, 172
4	I	2	9	Binomial Coefficients, theorem	1(ch 2) - pg 189
5	I	2	11	Binomial multinomial theorems	1(ch 2) - pg 201
6	I	2	13	Principles of Inclusion - Exclusion	1(ch 2) - pg 211
7	I	1	14	Derangements	1(ch 2) - pg 226
8	I	3	17	Generating functions	1(ch 3) - pg 237
9	I	1	18	Pigeon hole principles and its application.	1(ch 2) - pg 67
10	II	1	19	Matrices, Vectors	2(ch 7) - pg 327
11	II	2	21	Determinants	2(ch 7) - pg 370
12	II	1	22	Eigen values, eigen vectors	2(ch 7) - pg 386
13	II	2	24	Vector differential calculus - Inner and cross product	2(ch 8) - pg 436, 442
14	II	1	25	Gradient of a scalar field	2(ch 8) - pg 475
15	II	2	27	Divergence of a vector field	2(ch 8) - pg 486
16	II	1	28	Curl of a vector field	2(ch 8) - pg 483
17	III	1	29	Geometric transformations	
18	III	3	32	Translations, Rotation around the origin	
19	III	1	33	Rigid motions and homogeneous representations	
20	III	1	34	Affine transformations	
21	III	4	38	Coordinate transformation on Image Arrays	
22	IV	1	39	Numeric Analysis - Introduction	2(ch 18) - pg 919
23	IV	2	41	Solution of equations by iteration	2(ch 18) - pg 925
24	IV	1	42	Gauss Elimination	2(ch 19) - pg 972
25	IV	2	44	LU factorization	2(ch 19) - pg 981

26	IV	1	45	Matrix Inversion	2(ch 19) - pg 985
27	IV	2	47	Least Squares method	2(ch 19) - pg 1000
28	V	2	49	Optimization- Basic concepts, Unconstrained Optimization	2(ch 21) - pg 1084
29	V	3	52	Method of Steepest Descent	2(ch 21) - pg 1086
30	V	2	54	Linear Programming	2(ch 21) - pg 1089
31	V	3	57	Simplex method	2(ch 21) - pg 1092

References

- 1 Discrete Mathematics for computer scientists & mathematicians - JL Mott, A Kandel
- 2 Advanced Engineering Mathematics - Erwin Davis
- 3 Computer Graphics- Hearn & Baker



ADVANCED OPERATING SYSTEM

Sl. No	Module	No : of hrs	Cum . No. of hrs	Topic	Mode
1	1	1	1	Introduction	PPT
2	1	1	2	Functions of OS	PPT
3	1	2	4	Design approaches: layered, kernel based and virtual machine approach,	PPT
5	1	1	5	types of advanced operating systems NOS, DOS	PPT
6	1	1	6	Multiprocessor OS, Mobile OS	PPT
7	1	1	7	RTOS, Cloud OS	PPT
8	2	1	8	File Management- System Structure, User Perspective, Architecture	PPT
11	2	1	9	Buffer cache - File Representation: Structure of file Directories.	PPT
12	2	1	10	Memory Management- Detailed design of Process Structure: Kernel Data structures for process	PPT
14	2	1	11	Context of a Process: Static and Dynamic.	PPT
17	2	1	12	Parallel Systems and computing- Shared memory machines,	PPT
18	2	2	14	Synchronization, Communication	PPT
19	2	1	15	Shared memory multiprocessor OS.	PPT
20	3	1	16	Distributed Operating system concepts- Goals,	PPT
21	3	2	18	Distributed Computing Models	PPT
22	3	1	19	Hardware and Software Concepts	PPT
23	3	1	20	Architecture of DOS	PPT
26	3	2	22	Design Issues	PPT

27	3	1	23	Distributed communication	PPT
28	3	1	24	shared memory	PPT
29	3	2	26	synchronization - Distributed Object based System	PPT
32	4	2	28	Multiprocessor Operating System- Introduction, Basic multiprocessor system architectures	PPT
33	4	2	30	design issues	PPT
34	4	2	32	Threads, Process synchronization	PPT
35	4	1	33	the test and set instruction	PPT
36	4	1	34	the swap instruction	PPT
37	4	1	35	implementation of the process wait.	PPT
38	4	2	37	Processor scheduling: Issues	PPT
40	4	2	39	Coscheduling, Smart scheduling, Affinity Based scheduling	PPT
41	5	1	40	Real Time OS- Characteristics of Real Time operating Systems	PPT
42	5	1	41	Classification of Real Time Operating Systems	PPT
43	5	2	43	Scheduling in RTOS: Clock driven: cyclic, Event driven	PPT
44	5	2	45	EDF and rate monotonic scheduling	PPT
45	5	2	47	Mobile OS- Architecture, Android OS	PPT
46	5	3	50	iOS, Virtual OS, Cloud OS and their design issues.	PPT

DATA STRUCTURES USING JAVA

Sl. No.	Module No.	No: of hours	Cum. No. of hours	Topic	Source
1	I	2	2	Basic Concepts of Java: Object Oriented Programming	5(Ch 2)
2	I	2	4	Features of Java	5(Ch 1)
3	I	2	6	Classes,Constructors and Finalizers	5(Ch 6)
4	I	1	7	Interfaces	5(Ch 9)
5	I	1	8	Packages	5(Ch 9)
6	I	2	10	Exception Handling	5(Ch 10)
7	I	1	11	API.	5(Ch 15,16)
8	II	2	13	Multithreading	5(Ch 11)
9	II	2	15	EventHandling	5(Ch 22)
10	II	3	18	Applets,Graphics,Text,AWT Controls	5(Ch 21,23,24)
11	II	3	21	Layout Manager and Menus.	5(Ch 24)
12	II	2	23	I/O Streams,	5(Ch 19)
13	II	2	25	File Class and Operations.	5(Ch 19)
14	III	1	26	Introduction to Data structures, ADT	1(ch 1 - pg 3)
15	III	2	28	Searching - Linear and Binary Search & its implementations	2(ch 4), 1(ch 2- pg 30)
16	III	2	30	Sorting algorithms - Insertion, Bubble, Selection & its implementation	2(ch 4,9), 1(ch 14 - pg 256)
17	III	1	31	Mergesort algorithm & its implementation	1(pg 260), 2(ch 9), 3(pg 448)
18	III	2	33	Stack & its implementation	2(ch 6), 1(pg 103), 3(pg 135)
19	III	2	35	Quick sort algorithms & its implementation	2(ch 6), 1(pg 263), 3(pg 467)
20	III	2	37	Queue & its implementation	2(ch 6), 1(pg 117), 3(pg 149)

21	IV	5	42	Linked list & its implementation	2(ch 5), 1(pg 46), 3(pg 159), 4(pg 236)
22	IV	3	45	Hashing, Chaining	2(ch 9), 1(pg 148), 4(pg 256)
23	V	1	46	Trees	1(pg 186), 2(ch 7), 3(pg 229)
24	V	3	49	BST & its implementation	2(ch 7), 1(pg 235), 3(pg 382), 4(pg 286)
25	V	2	51	B-tree insertion, deletion & its implementation	2(ch 7), 1(pg 233), 4(pg 484)
26	V	3	54	AVL & its implementation	2(ch 7), 1(pg 237), 3(pg 393)
27	V	2	56	Red - Black trees & its implementation	3(pg 416), 4(pg 308)
28	V	2	58	Graph & its implementation	2(ch 8), 1(pg 285), 3(pg 539)
29	V	1	59	Spanning trees & its implementation	3(pg 596), 4(pg 624)

References

- 1 Data Structures with Java 2ed (Schaum's Outlines) - John Hubbard
- 2 Lipschutz-Data Structures (SIE).
- 3 Data Structures and Algorithms in Java - Michael T. Goodrich
- 4 Introduction to Algorithms - Thomas H. Cormen, Charles E. Leiserson, Rivest, Clifford Stein
- 5 Schildt Herbert, Java: The Complete Reference

OBJECT ORIENTED SOFTWARE ENGINEERING

Sl No	Module	No. Hrs	Cum No. Hrs	Topic	Source	CO
1	1	1	1	Introduction to Object oriented development	1	CO1
2	1	1	2	Object oriented systems development life cycle	1(39-54)	CO1
3	1	4	6	Object oriented methodologies-Booch methodology, Rumbaugh methodology, Jacobson's methodology	1(61-71)	CO1
4	1	2	8	Patterns and frameworks	1(72-77)	CO1
5	2	2	8	Fundamentals of Object oriented design using Unified Modeling Language	1(92,93)	CO2
6	2	2	10	UML-Use case diagram	1(101)	CO2
7	2	10	20	Class diagram, Sequence diagram, collaboration diagram, state chart diagram, Activity diagram, component diagram, deployment diagram	1(94)	CO2
8	3	3	23	Object oriented analysis: Use case Model-Identifying use cases, Identifying actors-Documentation	1(101)	CO2
9	3	3	26	Object analysis-classification-different approaches	1(151-172)	CO3
10	3	2	28	Identifying classes	1(177)	CO3
11	3	2	30	Identifying object relationships -Attributes and methods	1(178-183)	CO3
12	4	2	32	Object oriented design-design process	1(199-201)	CO3
13	4	3	35	Design axioms-corollaries- Design patterns	1(202-208)	CO3
14	4	2	37	Designing classes-designing protocols and class visibility	1(217-220)	CO3
15	4	3	40	Defining attributes-Designing methods-guidelines for identifying bad design	1(221-224)	CO3
16	5	2	42	Agile software development-Agile practices and principles	11(37-67)	CO4
17	5	2	44	Software quality assurance	1(325-328)	CO5
18	5	3	47	Bugs and debugging	1(337)	CO6
19	5	2	49	Testing strategies-developing test cases-developing test plans	1(328-337)	CO6
20	5	2	51	Debugging principles	1(337)	CO6

21	5	2	53	Introduction to agent oriented software engineering	wikipedia	C07
----	---	---	----	---	-----------	-----

Ali Bahrami, Object Oriented Systems Development, Tata McGraw-Hill, 1999

- Martin Fowler, UML Distilled, Second Edition, PHI/Pearson Education, 2002.
- James Rumbaugh, Ivar Jacobson, Grady Booch, The Unified Modelling Language Reference Manual, Addison Wesley, 1999.
- Stephen R. Schach, Introduction to Object Oriented Analysis and Design, Tata McGraw- Hill, 2003.
- Hans-Erik Eriksson, Magnus Penker, Brain Lyons, David Fado, UML Toolkit, OMG Press Wiley Publishing Inc., 2004.
- A survey of Agent-Oriented Software Engineering. Amund Tveit. Norwegian University of Science and Technology. May 8, 2001.
- Teach Yourself UML in 24 Hours, Joseph Schmuller, 3rd Edition, ISBN 81-297-0609-1, Pearson Education, 2004
- Grady Booch, James Rumbaugh, Ivar Jacobson, "UML User Guide", Addison Wesley, 2002.
- Craig Larman (2005) Applying UML and Patterns. An Introduction to Object-Oriented Analysis and Design and Iterative Development, Third Edition,
- Lethbridge and Laganieri, "Object-Oriented Software Engineering: Practical software development using UML and Java"

THEORY OF COMPUTATION

Sl. No.	Module No.	No. of Hrs	Cumulative no. of Hrs	Topic	Mode(BB/PPT)	Source
1	I	3	3	Concepts of automata theory	PPT	1(33-37)
2	I	2	5	Finite Automata-NFA,DFA	PPT	1(38-45)
3	I	2	7	NFA to DFA conversion	PPT	1(52-59)
4	I	2	9	Minimization of DFA	BB	2(288-293)
5	I	2	11	Moore and Mealy machines	BB	Internet
6	II	2	13	Regular expressions, Algebraic laws for regular expression	PPT	1(77-83)
7	II	2	15	Conversion of finite automata to regular expression	PPT	1(83-88, 91-92)
8	II	2	17	Conversion of regular expression to finite automata	PPT	2(38-45)
9	II	2	19	Applications of regular expressions	PPT	2(43-53)
10	II	2	21	Pumping Lemma for regular languages	PPT/ BB	2(66-72)
11	III	2	23	Context free grammar	BB	2(72-75)
12	III	2	25	Parse trees	BB	2(75-79)
13	III	2	27	Chomsky normal form, Greibach normal form	BB	2(85-96)
14	III	2	29	Pumping lemma for CFGs	BB	2(115-124)
15	III	2	31	Properties of CFGs	PPT	2(248-254)
16	III	3	34	PDA	PPT	2(254-258)
17	IV	2	36	Turing machine: Definition, Transition diagram, Church-Turing thesis	PPT	2(258-264)
18	IV	2	38	Design and roles of TM	BB/ PPT	2(267-272)
19	IV	2	40	Types of TM	PPT	3(66-82)
20	IV	2	42	Extensions of TM-Non-deterministic TMs	PPT	3(215-235)

21	IV	2	44	Restricted TMs	PPT	3(484-507)
22	V	2	46	Intractable problems:Defintion of P and NP problems	PPT	3(537-556)
23	V	3	49	NP complete and NP hard problems	PPT	3(590-608)
24	V	1	50	Decidability and Undecidability	PPT	3
25	V	2	52	Decidability and Halting Problems	BB	3
26	V	2	54	Undecidability and Reducability in TOC	BB	3

Text Books:

1. Software Engineering-A practitioner's approach, R.S.Pressman, McGraw Hill
2. Fundamentals of software engineering, Rajiv Mall, PHI
3. Software Engineering, Ian Sommerville, Pearson Education Asia Ed.
4. An integrated approach to software engineering, jalote P., Springer Publications

Model Examination Question Papers

SREE NARAYANA INSTITUTE OF TECHNOLOGY
FIRST SEMESTER MCA, MODEL EXAMINATION 2021

MCA20C12 ADVANCED OPERATING SYSTEM

Time:3 Hours

Maximum Marks: 100

Part A

(Answer all questions. All questions carry equal marks)

1. What is Kernel?
2. Write short notes on Network Operating System.
3. Give the structure of file directories?
4. What is a process.
5. How Distributed OS is different from a normal OS?
6. How Synchronization is important in DOS.
7. What are Multi Processor Operating Systems?
8. Give your points related with Threads?
9. Write the features of Real time OS.
10. Compare iOS and Android OS?

(10 x 4=40 Marks)

Part B

(Answer any two questions from each module. Each question carries 6 marks).

MODULE I

11. Explain the functions of Operating System
12. What are the three design approaches to OS. Explain.
13. Explain types of Operating Systems.

MODULE II

14. Explain the architecture of File Management
15. How Memory Management works in OS. Explain.
16. Draw and explain state chart diagram for University Admission management System.

MODULE III

17. Explain Distributed Operating System models.
18. Describe the architecture of DOS.
19. Write in detail the design issues in DOS.

MODULE IV

20. Explain basic multi-processor architectures.
21. Write in detail Process Synchronization.
22. Explain various process scheduling mechanisms.

MODULE V

23. Explain characteristics of RTOS in detail with the help of figures and diagrams.
24. Describe scheduling process in RTOS.
25. Give and explain architecture of Android OS.

(10x6=60 Marks)

SREE NARAYANA INSTITUTE OF TECHNOLOGY
FIRST SEMESTER MCA, MODEL EXAMINATION 2021

MCA20C13 DataStructures using Java

Time:3 Hours

Maximum Marks: 100

Part A

(Answer all questions. All questions carry equal marks)

1. How do java programs maintain platform independency with the help of JVM.
2. What do you mean by constructor overloading in Java?
3. What is AWT?
4. Write the lifecycle of applets.
5. Define ADT
6. Distinguish between stack and queue.
7. How do you insert an element in a circular linked list?
8. What is hashing?
9. Distinguish between AVL and B – trees.
10. Brief on the graph operations.

(10 x 4=40 Marks)

Part B

(Answer any two questions from each module. Each question carries 6 marks).

MODULE I

11. Explain the features of Java.
12. Discuss Exception handling in Java.
13. What is Interface and Package?

MODULE II

14. Explain how multithreading is done in Java.
15. Discuss the advantages and disadvantages of applets. Develop an applet that receives two numbers as input from the user and displays the sum of these numbers on the screen.
16. Explain file class and its methods.

MODULE III

17. Explain the binary search with an example.
18. Write a Java code to implement selection sort.
19. Explain the different Queue operations.

MODULE IV

20. Write notes on different types of linked list.
21. List the different hash functions with example.
22. Explain chaining.

MODULE V

23. How do you search an element in a BST? Explain.
24. Explain the concept of B – trees.
25. Explain Kruskal's algorithm for minimum spanning tree.

(10x6=60 Marks)

SREE NARAYANA INSTITUTE OF TECHNOLOGY
FIRST SEMESTER MCA, MODEL EXAMINATION 2021
MCA20C14 OBJECT ORIENTED SOFTWARE ENGINEERING

Time:3 Hours

Maximum Marks: 100

Part A

(Answer all questions. All questions carry equal marks)

1. What is OMT?
2. Distinguish patterns and frameworks.
3. What is activity diagram?
4. Write the importance of deployment diagram.
5. What is CRC?
6. List the important guidelines for defining attributes.
7. What is coupling?
8. How to refine attributes in object oriented systems?
9. Write a short note on quality assurance.
10. What is agent oriented software engineering?

(10 x 4=40 Marks)

Part B

(Answer any two questions from each module. Each question carries 6 marks).

MODULE I

11. Describe the steps in object oriented system development.
12. Explain macro and micro development process.
13. Briefly explain Jacobson methodology for object oriented development.

MODULE II

14. Draw and explain UML class diagram for Library Management System.
15. Distinguish UML sequence and collaboration diagram.
16. Draw and explain state chart diagram for University Admission management System.

MODULE III

17. With the example of an ATM system explain the various steps performed in identification of use case. Draw appropriate diagrams.
18. Describe any two approaches for identifying classes.
19. Explain any two object relationships.

MODULE IV

20. Explain various object oriented design corollaries.
21. Describe various class visibility modes.
22. Explain how to design methods in object oriented systems.

MODULE V

23. Explain various testing strategies.
24. Describe Myer's debugging principles.
25. Explain agile software development process.

(10x6=60 Marks)

Sree Narayana Institute of Technology
First Semester MCA Model Question Paper
MCA20C15 – Theory of Computation

Time: Three Hours

Maximum Marks: 100

Part A

(Answer all questions. All questions carry equal marks)

1. What is a finite automaton? Differentiate between NFA and DFA.
2. Write the applications of finite state machines.
3. What is regular expression? Write about operators in regular expression and their precedence.
4. What is a string and regular language?
5. What is ambiguous grammar?
6. What is a parse tree?
7. Discuss transition diagram with an example.
8. Write about the programming techniques in Turing machine.
9. Definition of P and NP problems.
10. Differentiate recursive and recursively enumerable languages.

(4 * 10 = 40)

Part B

(Answer any two questions from each module. Each question carries 6 marks)

Module I

11. Explain the conversion of NFA to DFA.
12. Explain finite automata with epsilon-transitions.
13. Explain minimization of DFA.

Module II

14. Explain the conversion of regular expression to finite automata.
15. Discuss the algebraic laws of regular expression.
16. Explain the properties of regular languages.

Module III

17. Discuss about pumping lemma for context free languages.
18. Define grammar? What is Chomsky normal form?
19. Write short notes on the applications of CFG.

Module IV

20. Explain about extensions of Turing machines.
21. Write a note on restricted Turing machines.
22. What are the applications of Turing machine?

Module V


23. Discuss briefly about NP complete problem.
24. Which are the undecidable problems about Turing Machines.
25. Discuss reducibility in TOC.

(10 * 6 = 60)

Academic Calendar

Schedules	MI 2020 Admission
Commencement of Classes	16 th November 2020
First Series Test	30 th November, 7, 14, 21 December, 4 th January 2021
First Series Lab	7, 8 January 2021
Second Series Test	18, 25 January 2021, 1, 8, 16 February
Second Series Lab	19, 20 January 2021
Model Theory Exam	8, 10, 12, 15, 18 March 2021
Model Lab exam	22, 23 March 2021
First Assignment	23, 30 November, 7, 14, 21 December
Second Assignment	11, 18, 25 January, 1, 8 February
Expected University examination	April 2021
Expected University Practical examination	April 2021
Registration to next Semester	May/June
Next Semester Classes	May/June

M1 approximate working days: NOV-13 days, DEC-19 days, JAN-23 days, FEB-20 days. Total = 75 days.

 SREE NARAYANA INSTITUTE OF TECHNOLOGY, KOLLAM. Application for Leave (for students)	
Name of applicant	
Class and Roll No.	
Date for which leave is requested for	
Details of any class exams, assignments during leave	
Reason for absence	
No. of days of leave granted so far during the current semester	
Date of application	
Signature of applicant	
Recommendation of Parent/ Guardian with Name and signature	
Recommended	Sanctioned
Chief Advisor / Advisor	Head of Department
	Principal
The days of leave will be considered only for the purpose of awarding sessional marks.	

Students can take photocopy of this page and submit.