Post Graduate Department of Computer Sciences, The University of Kashmir, Srinagar-190006

Choice Based Credit System Curriculum for Master of Computer Applications (MCA) Programme 2021 – 2023

Eligibility for 2-year MCA degree Programme:

"Passed BCA/ Bachelor Degree in Computer Science Engineering or equivalent Degree.

OR

Passed B.Sc./ B.Com./ B.A with Mathematics at 10+2 Level or at Graduation Level(with additional bridge Courses as per the norms of the concerned University).Obtained at least 50% marks (45% marks in case of candidates belonging to reserved category) in the qualifying Examination

Effective from year-2021

SEMESTER I

Subject Code	Subject Name	Subject Category	Hours/Week			Credit Units
			L	T	P	
Core Courses (14 Credit Units)						
MCA24101CR	Programming with C++	Core	3	0	2	4
MCA21102CR	Database Systems	Core	3	0	2	4
MCA21103CR	Computer Networks	Core	3	0	2	4
MCA21104CR	Accounting and Management Control	Core	2	0	0	2
Discipline Centric Electives (8 Credit Units)						
MCA21105DCE	Discrete Mathematics	DCE	3	1	0	4
MCA21106DCE	Numerical Techniques	DCE	3	1	0	4
MCA21107DCE	Computer Architecture and ALP	DCE	3	0	2	4
OE/GE (2 Credit Units) For Students of Other Departments						
MCA21001OE	Fundamentals of Computers	OE	2	0	0	2

SEMESTER II

Subject Code	Subject Name	Subject Category	Hours/Week			Credit Units
			L	T	P	
Core Courses (14 Credit Units)						
MCA21201CR	Data Structures Using C++	Core	3	0	2	4
MCA21202CR	Python	Core	1	0	2	2
MCA21203CR	Artificial Intelligence	Core	3	0	2	4
MCA21204CR	Software Engineering	Core	3	1	0	4
Discipline Centric Electives (8 Credit Units)						
MCA21205DCE	Web Programming	DCE	3	0	2	4
MCA21206DCE	Cryptography and Network Security	DCE	3	0	2	4
MCA21207DCE	Computer Graphics and Multimedia	DCE	3	0	2	4
OE/GE (2 Credit Units) For Students of Other Departments						
MCA21201OE	Web Designing	OE	2	0	0	2

SEMESTER III

Subject Code	Subject Name	Subject Category	Hours/Week			Credit Units
			L	T	P	
Core Courses (14 Credit Units)						
MCA21301CR	Design and Analysis of Algorithms	Core	3	0	2	4

MCA21302CR	Java Programming	Core	3	0	2	4
MCA21303CR	Operating System	Core	3	0	2	4
MCA21304CR	Machine Learning	Core	2	0	0	2
Discipline Centric Electives (8 Credit Units)						
MCA21305DCE	Theory of Computation	DCE	3	1	0	4
MCA21306DCE	Wireless and Mobile Communication	DCE	3	1	0	4
MCA21307DCE	Organizational Behaviour	DCE	3	1	0	4
OE/GE (2 Credit Units) For Students of Other Departments						
MCA21301OE	Fundamentals of Programming with C	OE	2	0	0	2

SEMESTER IV

Subject Code	**Subject Name**	**Subject Category**	**Hours/Week**			**Credit Units**
			L	**T**	**P**	
Core Courses (14 Credit Units)						
MCA21401CR	Project: Problem Identification	Core	0	4	0	4
MCA21402CR	Project: Problem Analysis	Core	0	6	0	6
MCA21403CR	Project: Dissertation	Core	0	6	0	6
Discipline Centric Electives (8 Credit Units)						
MCA21404DCE	Project : In Home Software Development	DCE	0	4	0	4
MCA21405DCE	Project: In Home Research Component	DCE	0	4	0	4
MCA21406DCE	Project: Industrial Software Development	DCE	0	4	0	4
MCA21407DCE	Project: Industrial Research Component	DCE	0	4	0	4
OE/GE (2 Credit Units) For Students of Other Departments						
MCA21401OE	Management Information System	OE	2	0	0	2

Semester I

COURSE TITLE: Programming with C++							
Course Code: MCA21101CR						**Examination Scheme**	
Total number of Lecture Hours: 48						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practicals(P):**	**2**	**Tutorial (T):**	**0**	**Total Credits**	**4**

Course Objectives

- To introduce students to the basic data types, variables, constants, and literals in C programming and to teach them how to use arithmetic, relational, logical, and bitwise operators.
- To teach students the various control structures, such as if-else, switch statements, and loops (while, do-while, for), and how to effectively use them to control the flow of a program.
- To enable students to understand and work with one-dimensional, two-dimensional, and multi-dimensional arrays, and to manipulate strings and character arrays using standard library functions.
- To develop students' ability to write functions, including prototypes and parameter passing, and to understand storage classes and identifier visibility. To teach recursive functions and their applications.
- To introduce students to advanced topics such as command-line arguments, file processing, structures and unions, and pointers. To explain the scope, lifetime, and multi-file programming.
- To provide a foundation in object-oriented programming with a focus on classes and objects, access specifiers, constructors, destructors, inheritance, polymorphism, and templates in C++. To introduce the concepts of abstraction, encapsulation, and exception handling.

Course Content	**TEACHING HOURS**
Unit I: Fundamentals of C Programming	**12 Hrs**
Data Types, Identifiers, Variables Constants and Literals. Arithmetic Relational Logical and Bitwise. Basic input/output statements, Control structures: if-else statement, Nested if statement, Switch statement Loops: while loop, do while, for loop, Nested loops. Arrays: Declaration; initialization; 2-dimensional and 3-dimensional array, passing array to function, Strings and String functions, and character arrays. Functions; prototype, passing parameters, storage classes, identifier visibility, Recursive functions	
Unit II: Advanced C Programming Techniques	**12 Hrs**
Command-line arguments. Variable scope, lifetime. Multi-file programming, Introduction to macros. File processing in C. Structures and unions: syntax and use, members, structures as function arguments passing structures and their arrays as arguments Pointers: variables, pointers and arrays, pointers to pointers, strings, pointer arithmetic, portability issues, pointers to functions, void pointers, pointer to structure. Introduction to object oriented programming, Abstraction, Encapsulation	
Unit III: Introduction to Object-Oriented Programming in C++	**12 Hrs**
Introduction to classes and objects; Access specifiers, Constructor; destructor; Function overloading; Operator overloading; friend functions; Use of call-by-reference for efficiency. Copy constructor. Inheritance: Single, Multiple, and Multilevel Inheritance ,Virtual functions and Polymorphism/Dynamic binding vs Static binding; Virtual Destructors.	
Unit IV: Advanced Object-Oriented Programming Concepts	**12 Hrs**

Pure virtual function; concrete implementation of virtual functions, Templates: Function Templates, Class Templates, Member Function Template and Template Arguments, namespaces, Exception Handling Concepts, Input and Output: Streams classes, Stream Errors, Disk File I/O with streams.	

Unit I

Lab Sheet 1

Q1. Write a program to demonstrate the use of Output statements that draws any object of your choice e.g. Christmas Tree using '*'

Q2. Write a program that reads in a month number and outputs the month name.

Q3. Write a program that demonstrate the use of various input statements like getchar(), getch(), scanf().

Q4. Write a program to demonstrate the overflow and underflow of various datatype and their resolution?

Lab Sheet 2

Q1. Write a program to demonstrate the precedence of various operators.

Q2. Write a program to generate a sequence of numbers in both ascending and descending order.

Q3. Write a program to generate pascals triangle.

Q4. Write a program to reverse the digits of a given number. For example, the number 9876 should be returned as 6789.

Lab Sheet 3

Q1. Write a program to convert an amount (upto billion) in figures to equivalent amount in words.

Q2. Write a program to find sum of all prime numbers between 100 and 500.

Q3. Create a one dimensional array of characters and store a string inside it by reading from standard input.
Q4. Write a program to input 20 arbitrary numbers in one-dimensional array. Calculate Frequency of each number. Print the number and its frequency in a tabular form.

Unit II

Lab Sheet 1

Q1. Write a C function to remove duplicates from an ordered array. For example, if input array contains 10,10,10,30,40,40,50,80,80,100 then output should be 10,30,40,50,80,100.

Q2. Write a program which will arrange the positive and negative numbers in a one-dimensional array in such a way that all positive numbers should come first and then all the negative numbers will come without changing original sequence of the numbers. Example: Original array contains: 10-15,1,3,-2,0,- 2,-3,2,-9 Modified array: 10,1,3,0,2-15,-2,-2,-3,-9

Q3. Write a program to compute addition multiplication and transpose of a 2-D array.

Q4. Implement a program which uses multiple files for holding multiple functions which are compiled separately, linked together and called by main(). Use static and extern variables in these files.

Lab Sheet 2

Q1. Implement a function which receiver a pointer to a Student struct and sets the values of its fields.

Q2. Write a program which takes five arguments on command line, opens a file and writes one argument per line in that file and closes the file.

Q3. Write a program which creates Student (struct) objects using malloc and stores their pointers in an array. It must free the objects after printing their contents.

Q4. Write a function char* stuff(char* s1,char* s2,int sp, intrp) to stuff string s2 in string s1 at position sp, replacing rp number of characters (rp may be zero).

Lab Sheet 3

Q1. Write a program to input name, address and telephone number of 'n' persons (n<=20). Sort according to the name as a primary key and address as the secondary key. Print the sorted telephone directory.

Q2. Write a program to find the number of occurrences of a word in a sentence ?

Q3. Write a program to concatenate two strings without using the inbuilt function?

Q4. Write a program to check if two strings are same or not?

Q5. Write a program to check whether a string is a palindrome or not?

Q6. Write a program to find the number of vowels and consonants in a sentence?

Unit III

Lab Sheet 1

Q1. Write a program that reverse the contents of a string.

Q2. Write a program to demonstrate the array indexing using pointers.

Q3. Write a program to pass a pointer to a structure as a parameter to a function and return back a pointer to structure to the calling function after modifying the members of the structure?

Q4. Write a program to demonstrate the use of pointer to a pointer.

Q5. Write a program to demonstrate the use of pointer to a function.

Q6. Write a program to demonstrate the swapping the fields of two structures using pointers?

Lab Sheet 2

Q1. Write a program in C++ to define class complex which having two data members viz real and imaginary part ?

Q2. Write a program in C++ to define class Person which having multiple data members for storing the different details of the person e.g. name,age, address, height etc.

Q3. Write a program to instantiate the objects of the class person and class complex ?

Q4. Write a C++ program to add member function that displays the contents of class person and class complex?

Q5. Write a C++ program to demonstrate the use of scope resolution operator?

Q6. Write a program in C++ which creates objects of Student class using default, overloaded and copy constructors.

Lab Sheet 3

Q1. Write a program to demonstrate the use of different access specifiers.

Q2. Write a C++ program to demonstrate the use of inline, friend functions and this keyword.

Q3. Write a C++ program to show the use of destructors.

Q4. Write a program in C++ demonstrates the use of function overloading.

Q5. Write a C++ program to overload the '+' operator so that it can add two matrices.

Q6. Write a C++ program to overload the assignment operator.

Q7. Write a C++ program to overload comparison operator operator== and operator!= .

Q8. Write a C++ program to overload the unary operator.

Unit IV

Lab Sheet 1

Q1. Write a program in C++ which creates a single-inheritance hierarchy of Person, Employee and Teacher classes and creates instances of each class using new and stores them in an array of Person * .

Q2. Write a program in C++ which creates a multiple-inheritance hierarchy of Teacher classes derived from both Person, Employee classes. Each class must implement a Show() member function and utilize scope-resolution operator

Q3. Write a program in C++ demonstrates the concept of function overriding?

Q4. Write a C++ program to show inheritance using different levels?

Q5. Write a C++ program to demonstrate the concepts of abstract class and inner class?

Lab Sheet 2

Q1. Write a C++ program to demonstrate the use of virtual functions and polymorphism?

Q2. Write a C++ program to demonstrate the use of pure virtual functions and virtual destructors?

Q3. Write a C++ program to swap data using function templates.

Q4. Write a C++ program to create a simple calculator which can add, subtract, multiply and divide two numbers using class template.

Lab Sheet 3

Q1. Write a C++ program to demonstrate the concept of exception handling.

Q2. Write a C++ program to create a custom exception.

Q3. Define a class with appropriate data members and member functions which opens an input and output file, checks each one for being open, and then reads name, age, salary of a person from the input file and stores the information in an object, increases the salary by a bonus of 10% and then writes the person object to the output file. It continues until the input stream is no longer good.

Textbooks

1. "C++ The complete Reference" by Herbert Schildt.

Reference Books

1. "C by discovery" by Foster and Foster
2. "Object orientation with C++ Programming" by Robert Lafore.
3. "Let us C" by Yashwant Kanetkar Phi.
4. "Programming in ANSI C" by E. Balaguruswami
5. "The C++ programming language" by Bjarne Stroustrup

COURSE OUTCOMES (CO):

CO1: Students will demonstrate proficiency in using basic data types, control structures, and input/output statements to develop efficient C programs.
CO2: Students will develop complex programs involving arrays, strings, and functions, including recursive functions and multi-dimensional arrays.
CO3: Students will apply advanced C programming concepts such as command-line arguments, file processing, and the use of pointers, structures, and unions to solve real-world problems.
CO4: students will implement object-oriented programming principles in C++, including the creation and management of classes and objects, and applying inheritance, polymorphism, and virtual functions.
CO5: Students will be able to utilize function and class templates, namespaces, and exception handling to enhance program robustness and reusability.
CO6: Students will be able to perform input and output operations using stream classes, handle stream errors, and manage disk file I/O operations effectively.

COURSE TITLE: Database Systems

Course Code:		**MCA21102CR**				**Examination Scheme**	
Total number of Lecture Hours: 48						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practicals(P):**	**2**	**Tutorial (T):**	**0**	**Total Credits**	**4**

Course Objectives

- Grasp the basic concepts of data, information, and knowledge, and the need for and evolution of databases and DBMS. Analyze the characteristics, advantages, and disadvantages of the DBMS approach.
- Describe data models, schemas, and instances, and compare various database models. Understand the Three Schema Architecture, data independence, database languages, interfaces, and DBMS classifications.
- Gain an overview of data modeling and create entity-relationship (ER) models to represent data structures and relationships effectively.
- Understand the basic concepts, characteristics, and constraints of the relational data model. Apply relational algebra operations, including unary, set theory, and binary operations, to manipulate relational data.
- Apply the criteria for good database design. Use functional dependencies and normalization techniques (1NF, 2NF, 3NF, BCNF) to design efficient and reliable database schemas that ensure data integrity and minimize redundancy.
- Learn SQL syntax and functionalities, including data definition, manipulation, and transaction control. Handle constraints, joins, views, synonyms, indexes, subqueries, and locks in SQL. Understand the basics of transaction processing, concurrency control, schedules, serializability, and recovery mechanisms to ensure database consistency and reliability.

Course Content	**TEACHING HOURS**
UNIT 1: Introduction to Database Systems	**12 Hrs**
Introduction to Data, Information and Knowledge. Database basics – Need and evolution, Database and DBMS. Characteristics of Database Approach, Advantages and disadvantages of DBMS Approach. Database System Concepts and Architecture – Data Models, Schemas, and Instances, Database Modelsand Comparison Three Schema Architecture and Data Independence. Database Languages and Interfaces. DBMS architectures. DBMS Classification. Data Modeling: Overview of Data Modeling, Entity-Relationship (ER) Modeling.	
UNIT 2: Relational Data Model and Database Design	**12 Hrs**

Relational Data Model –Basic Concepts and Characteristics, Model Notation, Model Constraints and Database Schemas, Constraint Violations Relational Algebra – basic concepts, Unary Relational Operations, Algebra Operations from Set Theory, Binary Operations, Additional Relational Operations Criterion for Good Database Design. Database Design through Functional Dependencies & Normalization: Functional Dependencies, Lossless Join, Normal Forms: 1NF, 2NF, 3NF, BCNF.	
UNIT 3: SQL and Advanced Data Definition	**12 Hrs**
Introduction to SQL, Data Types, Data Definition Language, Data Manipulation Language, Specifying Constraints in SQL, Transaction Control Language, SQL Functions, Set Operators and Joins, View, Synonym and Index, Sub Queries and Database Objects, Locks and SQL Formatting Commands.	
UNIT 4: Transaction Processing and Database Recovery	**12 Hrs**
Transaction Processing –Transaction Processing Basics, Concurrency Control, Transaction and SystemsConcepts, Desirable properties of Transactions. Characterizing Schedules and Recoverability, Schedules and Serializability. Concurrency Control - TwoPhase Locking, Timestamp Ordering. Database Recovery – Concepts, Transaction Rollback, Recovery based on Deferred and Immediate Update, Shadow Paging	

Lab Manual

UNIT I

Lab #1

List various users, functions and constraints of the database system for Library Management. b. List various users, functions and constraints of the database system for Banking System.

Lab #2

a. Identify the various tables and draw a diagrammatic schema to represent the database of Library management system. b. Identify the various tables and draw a diagrammatic schema to represent the database of University system.

Lab #3

- Draw ER Model for the database of Library management system. b. Draw ER Model for the database of University management system.

UNIT II

Lab #1

Consider the following schema: Suppliers (sid, sname, address) Parts (pid, pname, color) Catalog (sid, pid, cost) Write relational algebra queries to

- Find the name of suppliers who supply some red parts.
- Find the sids of suppliers who supply some red or green parts
- Find the sids of suppliers who supply some red part or are at Srinagar.
- Find the sids of suppliers who supply some red and some green part.
- Find the sids of suppliers who supply every part.
- Find the sids of suppliers who supply every red part.
- Find the sids of suppliers who supply every red or green part.

Lab #2

a. Consider a schema R(A,B,C,D) and functional dependencies A->B and C->D. Check the decomposition of R into R1(AB) and R2(CD) for lossless join and dependency preservation.

b. R(A,B,C,D) is a relation. Which of the following does not have a lossless join, dependency preserving BCNF decomposition?

1. A->B, B->CD

2. A->B, B->C, C->D

3. AB->C, C->AD

4. A ->BCD

Lab #3

- Using a sample schema and data, demonstrate the use of 1NF, 2NF, 3NF and BCNF

UNIT III

Lab #1

- Create table Student with following attributesandperformthefollowingoperations? i.

Attribute Name	ST_ROLLNO	ST_NAME	ST_ADDRESS	ST_TELNO
Date Type	Number	Varchar	Char	Varchar2
Size	6	30	35	15

- AddnewattributesCity,Street,CountrywithDatatypeVarcharandlength30?
- Modifyfield ST_ROLLNOand change thesize to5?
- RemovecolumnST_ADDRESS?
- DescribetheTableStudent?
- DropTableStudent?
- CopyStructureof onetableto another

- Create Users user1, user2,user3 and perform the following operations
- Grant Session Privilege to the newly created users?
- Grant privileges for creating and manipulation tables?

- Grant data manipulation privileges to various users on tables?
- Grant privileges with grant option.
- Revoke privileges.

Lab #2

a. Create Object ADDRESS and use the object in a Table DDL?
b. Create table Student with following attributes and perform the following operations.

Attribute Name	ST_ROLLNO	ST_NAME	ST_STREET	ST_CITY	ST_State	ST_Country	DTE_REG
DateType	Number	Varchar	Char	Char	Varchar2	Varchar2	Date
Size	6	30	35	30	30	30	

- Insert 10recordsinthetable.
- PerformvariousProjectOperationsusingSelectQuery.
- PerformvariousrestrictoperationsusingSelectQuery.
- Updaterecords inthetable.
- Deleterecordsinthetable.
- Create another table with same structure as existing table without copying the data.
- Create another table along with the structure and data from existing table.

Lab #3

a. Create table Student with ST_ADDRESS as Object Type with following attributes and

Attribute Name	**ST_ROLLNO**	**ST_NAME**	**ST_ADDRESS**				**DT_REG**
			ST_STREET	**ST_CITY**	**ST_State**	**ST_Country**	
Date Type	**Number**	**Varchar**	**Char**	**Char**	**Varchar 2**	**Varchar2**	**Date**
Size	**6**	**30**	**35**	**30**	**30**	**30**	

- Insert 10 records.
- Perform various Project Operations using Select Query.
- Perform various restrict operations using Select Query.
- Update records in the table
- Delete records in the table

b. Create table STUDENT with following attributes and perform the following operations?

Attribute Name	**ST_ROLLNO**	**ST_NAME**	**ST_STREET**	**ST_CITY**	**ST_State**	**ST_Country**	**DTE_REG**
Date Type	**Number**	**Varchar**	**Char**	**Char**	**Varchar 2**	**Varchar2**	**Date**
Size	**6**	**30**	**35**	**30**	**30**	**30**	

- Insert 10 records in the table.
- Perform various Project Operations using Select Query.
- Perform various restrict operations using Select Query using various arithmetic and Logical Operators like

- Less Than
- Greater Than
- Less Than or Equal to
- Greater Than or Equal To
- Equal to
- Not Equal To
- Perform restrict operations using various datatypes like numeric,Characters,Date.

- Perform Update operations using various Arithmetic and Logical Operators on Table STUDENT
- Perform Delete operations using various Arithmetic and Logical Operators on Table STUDENT
- Use Insert and Select Commands together with Arithmetic and Logical Operators.

UNIT IV

Lab #1

a. Perform following Transaction Control Operations on the above table
   - Perform various data manipulation operations the table .
   - Create Five Savepoints from S1 to S5.
   - Rollback to Various savepoints and observe the changes in the table.
   - Perform various DDL operations the table and observe its effect on Savepoint and Rollback on the table.
   - Try to abnormally terminate the application to observe whether data is saved or not.
   - Use Commit and Commit Work commands to save the data permanently.

b. Create table STUDENT with following attributes and perform various DML operations to verify domain constraint

Attribute Name	ST_ROLLNO	ST_NAME	ST_ADDRESS
Date Type	Number	Varchar2	Varchar
Size	6	30	35
Constraint	NOTNull	NotNULL	NOTNULL

Lab #2

a. Create table STUDENT with following attributes and perform various DML operations to verify Validity Integrity.

Attribute Name	ST_ROLLNO	ST_NAME	ST_ADDRESS
Date Type	Number	Varchar2	Varchar
Size	6	30	35
Constraint	CHECK(ROLLNO >20001 and ROLLNO<30001	NotNULL	NOTNULL

b. Create table STUDENT with following attributes and perform various DML operations to verify Entity Integrity using Primary and Unique Keys?

Lab #3

Attribute Name	ST_ROLLNO	ST_NAME	ST_ADDRSS
Date Type	Number	Varchar2	Varchar
Size	6	30	35
Constraint	Primary/UniqueKeys	NotNULL	NOTNULL

a. Create table STUDENT with following attributes and perform various DML operations to verify Referential Integrity using given tables (employee and department)?

Attribute Name	EMP_ID	EMP_NAME	ST_ADDRESS	DEPT_ID
Date Type	Number	Varchar2	Varchar	Number
Size	6	30	35	4
Constraint	PrimaryKey	NotNULL	NOTNULL	Foreign Key

Attribute Name	DID	NAME	Address
Date Type	Number	Varchar2	Varchar
Size	4	30	100
Constraint	Primary Key	NotNULL	NOTNULL

b. Write SQL queries to demonstrate use of Join and various SQL functions

Textbooks

1. Elmasri and Navathe, Fundamentals of Database Systems, 7/e, Pearson, 2016

Reference Books

1. Silberschatz, Korth, & Sudarshan, Database System Concepts, McGraw-Hill, 7/e, 2011.
2. Bayross I. SQL, Pl/SQL: The Programming Language of Oracle, BPB Publications, 2009
3. Michael J. Hernandez ,Database Design for Mere Mortals®: A Hands-on Guide to RelationalDatabase Design, Third Edition, Addison-Wesley Professional, 2013

COURSE OUTCOMES (CO):

CO1: Demonstrate the ability to understand the fundamentals of data, information, and knowledge. Evaluate the need, evolution, and characteristics of databases and DBMS, including their advantages and disadvantages.

CO2: Describe and apply various database system concepts and architectures, including data models, schemas, instances, and the Three Schema Architecture. Understand and use database languages, interfaces, and DBMS classifications.

CO3: Create effective data models using entity-relationship (ER) modeling. Apply relational data model principles and relational algebra operations. Design and normalize database schemas using functional dependencies and various normal forms (1NF, 2NF, 3NF, BCNF).

CO4: Utilize SQL for defining, manipulating, and controlling data. Handle constraints, joins, views, subqueries, and database objects. Understand and apply transaction processing concepts, concurrency control mechanisms, and database recovery techniques to ensure database consistency and reliability.

COURSE TITLE: Computer Networks

Course Code:MCA21103CR						**Examination Scheme**	
Total number of Lecture Hours:48						**External**	**80**
						Internal	**20**
Lecture (L):3	**-**	**Practical's (P):**	**2**	**Tutorial (T):**	**-**	**Total Credits**	**4**

Course Objectives

- ✓ To gain a comprehensive understanding of the core principles of computer networking, including protocol design, protocol layering, algorithm design, and performance evaluation.
- ✓ To acquire detailed knowledge of the OSI model and TCP/IP protocol suite and understand the design issues and protocols used in the data link layer and MAC sublayer.
- ✓ To Understand the design issues of the network layer, including various routing algorithms and congestion control mechanisms.
- ✓ To learn about the protocols used in the transport and application layers, including their design and functionality.

Course Content	**TEACHING HOURS**
UNIT 1:	**-12 Hrs**
Goals and applications of networks. LAN, MAN & WAN architectures. Concept of WAN subnet.Overview of existing networks. OSI Reference Model Architecture, TCP/IP Model and their comparison. Protocol layers and service models. OSI and Internet protocols.	
UNIT 2:	**12 Hrs**
Internetworking concept and architectural model. Connection-oriented and connection-less approaches. Discuss ATM and Ethernet. Concept of Virtual Circuits, Concept of Autonomous systems and Internetwork Routing. Classful IP addresses. Subnetting, Subnet addressing, IP Multicasting. Internet Protocol (IP): connectionless delivery of datagrams (MTU, fragmentation, reassembly). IP header structure. IP Addressing. Efficiency and consistency trade-offs.	
UNIT 3:	**12 Hrs**
Internet control protocols: ICMP, ARP and RARP. Concepts of delay, security, and Quality of Service (QoS). Reliable data transfer. Stop-and-Go evaluation. TCP and UCP semantics and syntax. TCP RTT estimation. Principles of congestion control. Principles of routing. Link-state and distance vector routing. Routing algorithms: Inter- and intra-domain routing. RIP, OSPF, BGP.CIDR. Transport Layer: UDP and TCP concepts. Socket API for Network Programming.	
UNIT 4:	**12 Hrs**
Client-Server application development using TCP & UDP sockets. Basic Server Architectures. Network Security: Overview of threats, cryptography, authentication, and firewalls their components. Encryption techniques and examples of encryption standards. Network management including SNMP. Network troubleshooting.	

Lab Manual

Unit I

Lab Sheet 1

Q1. Network components such as Modem, Gateways, Routers, Switches, Cables etc.

Q2. Various network softwares, services and applications.

Lab Sheet 2

Q1. Network trouble shooting Techniques: Trouble shooting basic TCP/IP problems.

Q2. Commands like ipconfig, getmac, tracert, pathping, arp, ping, netstat, finger etc

Lab Sheet 3

Q1. Straight cabling, Cross cabling, Signal testing, T568A and B wiring standards (including hands on practice)

Unit II

Lab Sheet 1

Q1. Program that prints the address of www.bitmesra.ac.in

Q2. Program that prints all the addresses of www.indianrail.gov.in

Lab Sheet 2

Q1. Program that scans lower ports and prints them.

Q2. Program to list host names from command line, attempt to open socket to each one and print the remote host, the remote port, the local address and the local port.

Lab Sheet 3

Q1. Program for splitting the URLs entered into command line into component parts.

Unit III:

Lab Sheet 1

Q1. Program to list all the interfaces available on a workstation.

Q2. Basics of TCP/IP and UDP/IP socket Programming

Lab Sheet 2

Q1. Program for "echo" client. The Client enters data to the server, and the server echoes the data back to the clients.

Lab Sheet 3

Q1. Program for "echo" Server. The Server listens at the port specified and reads from client and echoes back the result.

Unit IV

Lab Sheet 1

Q1. Basics of Serial Port programming

Lab Sheet 2

Q1. Program to write out "Hello World" to a serial port or to a USB to Serial Converter.

Lab Sheet 3

Q1. Simple RPC Programming. (Introductory level)

Textbooks
Andrew Tanenbaum, "Computer Networks", 6th Edition by Pearson, 2022
Reference Books
Behrouz A. Foruzan - Data communication and Networking, 6th edition, TMH, 2022

COURSE OUTCOMES (CO):

Upon successful completion of this course, learners will be able to:

1. **List the functionalities of different layers** in both the OSI and TCP/IP reference models.
2. **Analyze complex networking problems**, including the concepts of internetworking, and the differences between connection-oriented and connection-less approaches.
3. **Describe the principles of switching and routing algorithms** used in computer networks.
4. **Distinguish between TCP and UDP** formats and procedures, understanding their respective uses and characteristics.
5. **Identify, formulate, and analyze complex networking issues**, applying principles and concepts learned throughout the course.

COURSE TITLE: Accounting and Management Control

Course Code: MCA21104CR						**Examination Scheme**	
Total number of Lecture Hours: 24						**External**	**40**
						Internal	**10**
Lecture (L):	**2**	**Practicals(P):**	**Tutorial (T):**	**-**	**Total Credits**		**2**

Course Objectives

- Gain knowledge of the contributions made by key figures in the evolution of management, including Taylor, Mayo, and Fayol.
- Develop a comprehensive understanding of the core functions of management, including planning and decision-making.
- Understand the fundamental principles, objectives, and branches of accounting.
- Familiarize with the rules of debit and credit and the practical systems of bookkeeping, including the preparation of cashbooks, profit & loss accounts, and balance sheets.
- Develop the ability to prepare and interpret basic financial statements to assess an organization's financial health.

Course Content	TEACHING HOURS
UNIT 1:	**12- Hrs**
Evolution of Management: - Contribution of Taylor, Mayo & Fayol, Different approaches of management, role of manager, tasks of a professional manager, Management & its functions. Level of Management, managerial skills at various levels. Planning & Decision making: - Definition, Nature for planning, importance, Process of planning, decision making, nature importance & process, types of plans	
UNIT 2: Probability, Counting, and Relations	**12- Hrs**
Accounting, meaning, definition, objectives, accounting principles, branches of accounting, uses & limitations of Accounting, Basic Accounting Procedure –, rules of debit & credit, Practical system of book keeping – Cashbook, types of cash book, Profit & loss Account – meaning, Need & preparation, Balance Sheet- Meaning, need & Preparation,	

Reference Books

- Principles & Practice of Management – L. M. Prasad
- Management – Theory & Practice – C. B. Gupta
- Basics of Accounting – Jain & Narang
- Basic of Accounting – T. S. Grewal

COURSE OUTCOMES (CO):

CO1 Students will be able to explain the contributions of Taylor, Mayo, and Fayol to modern management theories and how these have shaped contemporary management practices.

CO2: Students will be able to define, plan, and implement effective organizational strategies.

CO3: Students will demonstrate a clear understanding of accounting principles and procedures, applying

them to maintain accurate financial records.
CO4: Students will gain proficiency in preparing key financial documents, such as cashbooks, profit & loss accounts, and balance sheets..

COURSE TITLE:DISCRETE MATHEMATICS

Course Code: MCA21105DCE						**Examination Scheme**	
Total number of Lecture Hours: 48						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practicals(P):**	**Tutorial (T):2**	**-**	**Total Credits**		**4**

Course Objectives

- Understand and apply fundamental concepts of propositional logic, truth tables, and logical equivalence.
- Demonstrate proficiency in handling predicates, quantifiers, and operations on sets, including cardinality.
- Utilize various methods of proof, including direct proof, indirect proof, and mathematical induction, to solve problems and prove the correctness of algorithms.
- Apply counting techniques such as permutations, combinations, and the Pigeonhole Principle to solve problems in discrete mathematics
- Analyze and apply principles of discrete probability, including advanced counting techniques like the inclusion-exclusion principle and solving recurrence relations.
- Interpret and analyze relations, digraphs, and basic graph theory concepts, including connectivity, paths, circuits, and graph coloring, using appropriate mathematical tools and representations.

Course Content	TEACHING HOURS
UNIT 1: Foundations of Discrete Mathematics	**12- Hrs**
Proposition, Logic, Truth tables, Propositional Equivalence, Logical Equivalence, Predicates and Quantifiers; Sets: operations on sets, Computer representation of sets, Cardinality of a Set Functions: Domain, Range, One-to-One, Onto, Inverses and Composition, Sequences and summations, Growth of functions. Methods of Proof: Direct Proof, Indirect Proof, Mathematical Induction for proving algorithms; Counting techniques – Permutations, Combinations, The Pigeonhole Principle.	
UNIT 2: Probability, Counting, and Relations	**12- Hrs**
Discrete Probability, Advanced Counting Techniques: Inclusion-Exclusion, Applications of Inclusion exclusion principle, recurrence relations, solving recurrence relation. Relations: Relations and their properties, Binary Relations, Equivalence relations, Diagraphs, Matrix representation of relations and digraphs. Computer representation of relations and digraphs; Transitive Closures, Warshall's Algorithm, Problem solving on Warshall's Algorithm.	
UNIT 3: Ordered Sets and Graph Theory	**12- Hrs**
Partially Ordered Sets (Posets), External elements of partially ordered sets, Hasse diagram of partially ordered set, isomorphic ordered set, Lattices: Properties of Lattices, complemented Lattices. Graph theory: Introduction to graphs, Graph Terminology Weighted graphs, Representing Graphs, Connectivity of Graphs: Paths and Circuits, Eularian and	

Hamiltonian Paths, Matrix representation of graphs. Graph Coloring and its applications.	
UNIT 4: Trees, Boolean Algebra, and Groups	**12-Hrs**
Trees: Rooted trees, Application of trees: Binary Search Trees, Decision Trees, Prefix Codes, Tree traversal, trees and sorting, spanning trees, minimal spanning trees. Finite Boolean algebra, Functions on Boolean algebra, Boolean functions as Boolean polynomials. Groups and applications: Subgroups, Semigroups, Monoids Isomorphism, Homomorphism.	

TUTORIAL

Unit 1

Tutorial Sheet #1

1. Find whether $(p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)$ is a tautology or a contradiction?
2. Show that: $\neg (p \vee (\neg p \wedge q))$ and $(\neg p \wedge \neg q)$ are logically equivalent by using the propositional laws.
3. Let P(x, y, z) : "x + y = z".
Find the truth values of the following: A) P(1, 2, 3) B) P(0, 0, 1)
4. How many students must be in a class to guarantee that at least two students receive the same score in the final exam, if the exam is graded on a scale from 0 to 100 points?

Tutorial Sheet #2

1. Each user on a computer has a password, which is six to eight characters long, where each character is an uppercase letter or a digit. Each password must contain at least one digit. How many possible passwords are there?
2. A playoff between two teams consists of at most five games. The first team that wins three games wins the playoff. In how many different ways can the playoff occur? Use tree diagram.
3. A young pair of rabbits (one of each sex) is placed on an island. A pair of rabbits does not breed until they are two months old. After they are two months old, each pair of rabbits produces another pair each month. Find a recurrence relation for the number of pairs of rabbits on the island after n months, assuming that no rabbits ever die.
4. Conjecture a simple formula for anif the first 10 terms of the sequence {an} are: 1, 7, 25, 79, 241, 727, 2185, 6559, 19681, 59047.

Tutorial Sheet #3

1. Show that the set of all integers is countable. 2. Give a direct proof of the theorem "If n is an odd integer, then n 2 is odd."
3. Express the statement "Everyone has exactly one best friend" as a logical expression involving predicates, quantifiers with a domain consisting of all people, and logical connectives.
4. Use a membership table to show that $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

Unit 2

Tutorial Sheet #1

1. How many onto functions are there from a set with six elements to a set with three elements?

2. Suppose there are seven coins, all with the same weight, and a counterfeit coin that weighs less than the others. How many weighing's are necessary using a balance scale to determine which of the eight coins is the counterfeit one? Give an algorithm for finding this coin.
3. Is the divides relation on the set of positive integers reflexive, symmetric, antisymmetric, and transitive?
4. What are the sets in the partition of the integers arising from congruence modulo

Tutorial Sheet #2

1. What is the probability that when two dice are rolled, the sum of the numbers on the two dice is 7?

2. An urn contains four blue balls and five red balls. What is the probability that a ball chosen at random from the urn is blue?
3. How many ways are there to assign five different jobs to four different employees if every employee is assigned at least one job?

Tutorial Sheet #3

1. Draw the Hasse diagram of (D(75), divides), where the set D(75) represents the set of all positive divisors of 75.
2. Which elements of the poset ({2, 4, 5, 10, 12, 20, 25}, /)are maximal, and which are minimal?

3. Find a compatible total ordering for the poset ({1, 2, 4, 5, 12, 20}, /).
4. Draw the Hasse diagram for the partial ordering {(A,B) | A ⊆B} on the power set P(S) where S = {a, b, c}.

Unit 3

Tutorial Sheet #1

1. Find out the transitive closure of any relation R using Warshall's Algorithm.
2. Let R be the relation on the set of real numbers such that aRb if and only if a − b is an integer. Is R an equivalence relation?
3. Let R be the relation on the set of people such that xRy if x and y are people and x is older than y. Show that R is not a partial ordering.

Tutorial Sheet #2

1. How many edges are there in a graph with 12 vertices, each of degree 4?
2. A connected graph has an Euler path but not an Euler circuit iff it has exactly two vertices of odd degree. Verify this theorem by drawing a graph of the said property.
3. What is the chromatic number of the graph Cn, where n ≥ 3? (Cn is the cycle with n vertices.)

Tutorial Sheet #3

1. Show that Knhas a Hamilton circuit whenever n ≥ 3.
2. Suppose that a connected planar simple graph has 20 vertices, each of degree
3. Into how many regions does a representation of this planar graph split the plane?
3. Use Dijkstra's algorithm to find the length of a shortest path between any two verticesin some weighted connected graph.

Unit 4

Tutorial Sheet #1

1. Use Prim's algorithm to find a minimum spanning tree of any graph.
2. Is the set Z (a set of integers) monoid under usual operation of +, - ?
3. Form a binary search tree for the following words in alphabetical order. mathematics, physics, geography, zoology, meteorology, geology, psychology, and chemistry

Tutorial Sheet #2

1. What is the chromatic number of the complete bipartite graph Km,n, where m and n are positive integers?
2. How can we find out whether two graphs are isomorphic or not?
3. Show that C6 is bipartite. Also show that K3 is not bipartite.

Tutorial Sheet #3

1. What is the significance of Erdos number with regards to Paths in Collaboration Graphs?
2. How can backtracking be used to decide whether a graph can be colored using n colors?
3. What is the value of following prefix expression? + - * 2 3 5 / ↑ 2 3 4

Textbooks

KENNETH H. ROSEN "Discrete Mathematics and Its Applications" The Random House/Birkhauser Mathematics series

Reference Books

- LIU, "Elements of Discrete Mathematics", Tata McGraw Hill
- SCHAUMS, "Discrete Mathematics", Tata McGraw Hill.

- KOLMAN/REHMAN, "Discrete Mathematical Structures", Pearson Education
- NICODEMI "Discrete Mathematics", CBS

COURSE OUTCOMES (CO):

CO1: Ability to Apply Logical Reasoning and Proof Techniques: Students will demonstrate proficiency in using propositional and predicate logic to construct valid arguments and proofs. They will apply methods such as direct proof, indirect proof, and mathematical induction to solve problems and analyze algorithms.

CO2: Competence in Counting and Probability Analysis: Students will be able to apply counting techniques such as permutations, combinations, and the Pigeonhole Principle to solve discrete probability problems. They will analyze recurrence relations and apply advanced counting principles like the Inclusion-Exclusion Principle.

CO3: Understanding and Application of Graph Theory and Relations: Students will acquire knowledge of graph theory, including graph representations, connectivity, paths, cycles, and graph coloring. They will understand properties of relations, matrix representations of relations and digraphs, and algorithms like Warshall's Algorithm for transitive closure.

CO4: Proficiency in Structural Analysis and Algebraic Concepts: Students will demonstrate proficiency in analyzing structures such as partially ordered sets (Posets), lattices, trees, and Boolean algebra. They will apply concepts of functions, groups, and monoids to solve problems in various applications, including decision trees, sorting, and Boolean functions.

COURSE TITLE: Numerical Techniques

Course Code:MCA 21106DCE						**Examination Scheme**	
Total number of Lecture Hours: 48						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practicals(P):**		**Tutorial (T):**	**1**	**Total Credits**	**4**

Course Objectives

- **Understand Numerical Foundations**: Comprehend the principles behind floating point representation, arithmetic, and error types in numerical computations.
- **Master Iterative Techniques**: Develop proficiency in iterative methods for solving non-linear equations, including convergence analysis and implementation of algorithms.
- **Solve Linear Systems**: Apply direct and iterative methods to solve simultaneous linear equations, analyzing their stability and convergence properties.
- **Explore Advanced Numerical Techniques**: Utilize advanced techniques such as eigenvalue problems, matrix decompositions, interpolation, and numerical integration in practical applications.
- **Apply Interpolation and Approximation**: Implement interpolation techniques including Lagrange Polynomial, cubic splines, and least square approximation for accurate data representation.
- **Implement Differential Equation Solvers**: Develop skills in implementing numerical methods like Euler's Method, Modified Euler, and Runge-Kutta methods for solving ordinary differential equations.

Course Content	**TEACHING HOURS**
UNIT 1	**-12 Hrs**
Introduction to Floating Point Representation and Arithmetic, Normalized Floating Point Representation of Numbers Approximations & Errors: Types of Errors: Programming, Data, Computer & Arithmetic, Round off and Truncation Errors, Accuracy and Precision, Measures of Accuracy, Error Propagation. Iterative Methods for Non-Linear Equations: Brute Force method, Bisection, Secant, Newton-Raphson, Method of False Position, Derivation of Mathematical Formulas and Implementation	
UNIT 2	**- 12 Hrs**
Advanced Techniques for Non-Linear Equations: Fixed-Point Iteration, Muller's Method, Convergence Analysis	

Solution of Simultaneous Linear Equations: Direct Methods: Gauss Elimination, Gauss-Jordan, LU Decomposition, Pivoting Strategies, Iterative Methods: Gauss-Seidel, Jacobi Methods, Convergence and Stability Analysis Eigenvalue Problems and Matrix Decompositions: Power Method, QR Algorithm, Singular Value Decomposition, Applications	
UNIT 3	**-12 Hrs**
Interpolation Techniques: Lagrange Polynomial, Newton's Forward and Backward Difference Methods, Cubic Splines and Hermite Interpolation, Least Square Approximation: Linear and Polynomial Regression, Taylor Series, Chebyshev Polynomials, Fourier Series and Transform Numerical Differentiation and Integration: Trapezoidal Rule, Simpson's Rule, Adaptive Quadrature Methods	
UNIT 4	**-12 Hrs**
Numerical solution of Differential Equations using Taylor Series Method, Euler's Method, Modified Euler's Method, Runge-Kutta Methods, Predictor Corrector Formula, Higher order Differential Equations, Comparison of Runge-Kutta, Predictor and Correction Methods. Derivation of mathematical formulas and implementation of these methods	

Tutorial

Tutorial Sheet 1

Unit I:

Q1.Define different types of errors.

Q2.Let X= 0.005998. Find relative error if x is truncated to 3 decimal digits

Q3.Let X= 0.005998. Find relative error if x is truncated to 3 decimal digits.

Tutorial Sheet 2

Unit I:

Q1.What do you mean by approximation and error?

Q1.Find the root of the equation 2x-x-3 = 0 graphically.

Q2.Find the root of the equation correct to three decimal digits using False Position Method. cos x -3x+1 = 0

Tutorial Sheet 3

Unit I:

Q1.What is the difference between accuracy and precision? Define the two ways for measuring accuracy.

Q2.Find the root of the equation correct to three decimal digits using Bisection Method. X3 – 2X - 5 =0 Write the programming implementation of Bisection method for the above question.

Q3.Prove Newton-Raphson method analytically.

Tutorial Sheet 1

Unit II:

Q1.Prove Newton-Raphson method analytically

Q2.What are the various methods to obtain solutions of non-linear equations?

Q3.Solve the following system of linear equations using Gauss Seidel Method, correct to three decimal digits. 10x1 + x2 + 2x3 = 44 2x1 + 10x2 + x3 = 51 x1 + 2x2 + 10x3 = 61 Give the programmatic implementation of Gauss Seidel method

Tutorial Sheet 2

Unit II:

Q1.Give the programmatic implementation of Gauss Jordon method

Q2.What is the difference between the Gauss-Jordon and Gauss-Elimination?

Q3.Give examples of various direct and iterative methods to obtain solutions of non-linear equation.

Tutorial Sheet 3

Unit II:

Q1.Solve the following system of linear equations using Gauss Elimination Method.

$$2x1 + 8x2 + 2x3 = 14$$

$$x1 + 6x2 - x3 = 13$$

$$2x1 - x2 + 2x3 = 5$$

Q2.Give the programming implementation of Gauss elimination method.

Q3.Solve the following system of linear equations using Gauss Jordon Method.

$$2x1 - 2x2 + 5x3 = 13$$

$$2x1 + 3x2 + 4\,x3 = 20$$

$$3x1 - x2 + 3x3 = 10$$

Tutorial Sheet 1

Unit III:

Q1.What are the ways to approximate a function by a polynomial? Describe each in brief

Q2.We want to compute sin(x) correct to three significant digits. Obtain a series with minimum number of terms using Taylor series.

Q3.We want to compute sin(x) correct to three significant digits. Obtain a series with minimum number of terms using Chebyshev series.

Tutorial Sheet 2

Unit III:

Q1.Give the programmatic implementation of Chebsyshev series.

Q2.Give the derivation of trapezoidal method.

Q3.Give the programmatic implementation of trapezoid method.

Tutorial Sheet 3

Unit III:

Q1.State newton's methods of interpolation – forward difference, backward difference

Q2.State Linear Regression and Polynomial Regression

Q3.Give the programming implementation of Taylor Series

Tutorial Sheet 1

Unit IV:

Q1.Give a brief idea about runge-kutta (RK) methods.

Q2.Given dy/dx = xy with y(1) =5. Find solution correct to decimal positions in the interval [1,1.3] using RK second order method (step size h=0.1)

Q3.Provide the programmatic implementation of RK 2nd order method.

Tutorial Sheet 2

Unit IV:

Q1.Given dy/dx = xy with y(1) =5. Find solution correct to decimal positions in the interval [1,1.3] using RK second order method (step size h=0.1)

Q2.Explain the following terms with suitable examples.

a. Differential equation

b. Solution of Differential equation

c. Order of differential equation

Q3.Using Modified Euler's method, find the solution of the following differential equation dy/dx = x+y2, for x= 1.1, 1.2, 1.3. Given that y=1, x=1.

Tutorial Sheet 3

Unit IV:

Q1.Give the derivation of Modified Euler's method.

Q2.Differentiate between the following: a. Single step and multiple step methods b. Ordinary and partial derivate c. Ordinary and partial differential equations Give programmatic implementation of Least Square Method for curve fitting

Textbooks

1. S.C. Chapra & R.P. Canale: "Numerical Methods for Engineering", Tata McGraw Hill.

Reference Books

1. Krishenmurty and Sen: "Numerical Algorithms"
2. V. Rajaraman: "Computer Oriented Numerical Methods", Prentice Hall of India.
3. Grewal, B. S.: "Higher Engineering Mathematics", Hindustan Offset Problems Series.

COURSE OUTCOMES (CO):

CO1: Students will explain floating point representation, identify various types of errors, and understand their impact on accuracy and precision.

CO2: Students will apply and implement iterative methods such as Brute Force, Bisection, Secant, Newton-Raphson, and Method of False Position for solving non-linear equations.

CO3: Students will gain proficiency in direct methods like Gauss Elimination, Gauss-Jordan, and LU Decomposition, as well as iterative methods like Gauss-Seidel and Jacobi.

CO4: Students will apply techniques such as the Power Method, QR Algorithm, and Singular Value Decomposition for solving eigenvalue problems and performing matrix decompositions.

CO5: Students will derive and programmatically implement interpolation techniques including Lagrange Polynomial, Newton's Methods, and Cubic Splines, as well as least squares approximation methods.

CO6: Students will use numerical methods like Trapezoidal Rule, Simpson's Rule, and Runge-Kutta Methods for differentiation, integration, and solving ordinary differential equations (ODEs).

COURSE TITLE: Computer Architecture

Course Code:		**MCA21107DCE**				**Examination Scheme**	
Total number of Lecture Hours: 48						**External**	**80**
						Internal	**20**
Lecture(L):	**3**	**Practicals(P):**	**2**	**Tutorial(T):**	**0**	**Total Credits**	**4**

Course Objective:

- To introduce the basic concepts and principles of computer architecture.
- To explore the design and organization of processors.
- To study the design and implementation of memory hierarchies.
- To examine the interface between the computer system and peripherals, including buses and storage devices.

Course Content	TEACHING HOURS
UNIT 1:	**- 12 Hrs**
8086 Microprocessor: 8086 Microprocessor Architecture (BIU, EU, Instruction Queue), Software Model (General Purpose Registers, Segment Registers, Flag & Other Registers). Segmentation. 8086 Pin Functions, Minimum and Maximum Mode, The 8086 Memory System 8086 Basic Programming: 8086 Programming Model, 8086 Instruction Formats, Addressing Modes. .	
UNIT 2:	**- 12 Hrs**
The 8086 Instruction Set. , Assembly Language Programming: Significance, Assemblers and Linkers, TASM Directives – Data Definitions, Named-constants, User-defined, Segments, Subroutines, Macros, Modular-code. Programming with Data Transfer, Arithmetic and Logical Instructions: Data Transfer, Arithmetic, Logical/Bit Manipulation Instructions	
UNIT 3:	**-12 Hrs**
Branching and Looping: Unconditional and Conditional Jump instructions, Decision making and looping, Loop instructions, ASCII and BCD Arithmetic, Processor Control Instructions. Shift Instructions, Rotate Instructions and String Instructions Stacks: Defining a stack, Push and Pop Instructions	
UNIT 4:	**12- Hrs**
Procedures: Defining and Calling procedure. CALL and RET instructions, Parameter Passing Methods, far procedure Macros: Working with macros, additional assembler directives INT 21H: INT 21H Keyboard Services, Display Services, and File Manipulation Services. Input/Output Instructions.	

LAB MANUAL

UNIT 1:

LabSheet 1.

This week students will learn how to declare, initialize and access variedsized variables by using Assembler Directives and 8086 instructions.

a. Write a program that declares and initializes two integer variables (one 8-bit wide and another 16- bit wide), and then assigns new values to them using 8086instructions.

b. Write a program that declares and initializes a String array (byte array) of 10 elements, and then assigns new values to each element individually using 8086instructions.

LabSheet 2.

This week students will learn how to use INT 21H service to read integers and strings from keyboard and display them on screen.

a. Writeaprogramthatreadsanintegervaluefromkeyboardusing INT21Hkeyboardservic e,stores itinmemory,anddisplaysitusingINT21Hdisplayserviceafterdoingnecessary ASCIIconv ersion.

b. Write a program that declares and initializes a String array(byte array), and uses INT 21H display service to displays all elementsindividually.

LabSheet3.

Thisweekstudentswilllearnhowtoperformarithmetic operations of 8- bitintegervalues

a. Write a program that reads two 8-bit integers from keyboard (using INT 21H) and displays their sum and difference (using INT 21H after doing necessary ASCIIconversion).

b. Write a program that reads two 8-bit integers from keyboard (using INT 21H) and displays their multiplication and division result (using INT 21H after doing necessary ASCIIconversion).

UNIT 2:

LabSheet 1.

This week students will learn how to perform various logical operations on integer values.

Write to program reads two integers from keyboard (using INT 21H) and displays the result ofthe AND, OR, XOR, CMP and TEST operation (using INT 21H after doing necessary ASCII conversion).

LabSheet 2

This week students will learn how to use a subroutine to recursively solve a problem

a. Write a program that defines a subroutine that uses recursion to calculate factorial of an integer read from keyboard.

Lab Sheet 3.

This week students will learn how to use Macros. a. Write a program that uses a Macro to exchange the values of two 16-bit integer variables.

UNIT 3:

LabSheet 1.

This week students will learn how to use 8086 instructions for looping and decision making

a. Write a program that reads an integer from keyboard (using INT 21H service), and iteratively calculates its factorial.

b. Write a program that declares and initializes an array of 10 elements each 8-bit wide, reads an 8- bit integer from keyboard, searches its existence through the array, and displays the result of the searchoperation.

c. Write a program that declares and initializes an array of 10 elements each 8-bit wide, and sorts its elements in ascendingorder.

LabSheet 2:

This week students will learn how to perform rotate and shift operations.
a. Writetoprogramreadstwointegersfromkeyboard(usingINT21H)anddisplaystheresul tofthe SHL,SHR,SAR,ROL,ROR,RCL,andRCRoperation(usingINT21Hafterdoingnece ssaryASCII conversion).

LabSheet 3.

This week students will learn how to define a subroutine, pass parameters to it, and return value from it.
a. Write a program that defines a subroutine, which takes two 8-bit integers as parameters via Registers, calculates their sum, and returns the result to thecaller.
b. Write a program that defines a subroutine, which takes two 8-bit integers as parameters via Stack, calculates their sum, and returns the result to thecaller.

UNIT 4:

LabSheet 1:

This week students will learn how to read and write files residing on secondary storage using INT 21H service.
a. Write a program that opens an existing text file in the current working directory, and display its contents.
b. Write a program thatcreatesafileincurrentworkingdirectory,writestextualdatatoit(rea d from keyboard), and closesit.

LabSheet 2.

This week students will learn how to write, install and use a custom software-interrupt.
a. Write a program that creates a subroutine to display "hello world!", installs the subroutine as ISR, and subsequently uses it via INTinterface.

LabSheet 3.

This week students will learn how to write a simple device driver for VGA.
a. Writeaprogramthatdefinesasubroutine,whichtakesthreeparameters – row,column,addressof theString,andusesmemorymappedI/Otodisplayitonscreen.Theprogramcallsthissubroutineto display a String inputted viakeyboard

Textbooks

- **"Computer Architecture: A Quantitative Approach"** by John L. Hennessy and David A. Patterson
- **"Computer System Architecture"** by Mano Morris.
- **"Computer Organization & Architecture"** by William Stalling
- **"Computer architecture and parallel processing"** by Kai Hwang and F.A.Briggs

Reference Books

- **"Computer Architecture: Concepts and Evolution"** by Gerrit A. Blaauw and Frederick P. Brooks Jr.
- **"Computer Architecture and Organization: An Integrated Approach"** by Miles J. Murdocca and Vincent P. Heuring
- **"Computer Architecture: Pipelined and Parallel Processor Design"** by Michael J. Flynn

COURSEOUTCOMES(CO):

CO1: Describe the components of a computer system and explain their functionalities.
CO2: Demonstrate concepts of parallelism in hardware/software.
CO3: Describe architectural features of advanced processors
CO4: Interpret performance of different pipelined processors
CO5: Apply concepts of memory hierarchy, input/output systems, and parallel architectures to solve practical problems.

COURSE TITLE: Fundamentals of Computers

Course Code: MCA21101OE						**Examination Scheme**	
Total number of Lecture Hours: 24						**External**	**40**
						Internal	**10**
Lecture (L):	**2**	**Practicals(P):**	**-**	**Tutorial (T):**	**-**	**Total Credits**	**2**

Course Objectives

- To understand basics of computer and working with OS.
- To develop working skills with productivity tools, graphics designing and Internet.
- To acquire basic programming skills.
- To apply computing in problem

Course Content	**TEACHING HOURS**
UNIT 1:	**12 Hrs**
Introduction about computers : Computer basics, Characteristics, applications & Limitations. Functional Block Diagram of Computer, Computer Architecture : Classification of Computers on the basis of Purpose, signal and size and probability. Evolution of Computer from !st generation to %th generation, Data representation in memory	
UNIT 2:	**12 Hrs**
Hardware : Input devices used: Keyboard, mouse, OMR, OCR,MICR,BCR, Scanner, internal structure of CPU : Registers, ALU, Motherboard, HD, Memory, Cache and Virtual Memory, Magnetic Disk, Optical Disk and Flash memory, Software :- Types, Languages and their types, Operating System, its types and various functions , Basic introduction about interface, its types , GUI (DOS and Windows)	

Textbooks : Fundamentals of Computers by V. Rajaraman and N Abadala 6th edition

References:

Computer Fundamentals by Pradeep Sinha

COURSE OUTCOMES (CO):

CO1: Converse in basic computer technology
CO2: Formulate opinion about the impact of computers on society
CO3: Possess the knowledge of basic hardware peripherals

SEMESTER II

COURSE TITLE: Computer Graphics and Multimedia

Course Code			**: MCA21207DCE**			**Examination Scheme**	
Total number of Lecture Hours: 40						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practical(P):**	**2**	**Tutorial (T):**	**0**	**Total Credits**	**4**

Course Objectives:

- **Understand Computer Graphics Principles:** Grasp the fundamental concepts and applications of computer graphics, including graphic display devices and 2D/3D transformations.
- **Implement Graphics Algorithms:** Develop skills in essential graphics algorithms such as line and circle drawing, clipping, filling, and hidden surface removal.
- **Work with Curves and Surfaces:** Apply mathematical techniques for curves and surfaces, including spline and Bezier methods, for creating complex graphical models.
- **Integrate Multimedia Elements:** Learn about multimedia concepts, file formats, and storage solutions to create and manage multimedia content effectively. Introductory concepts of AR &VR technologies.

Course Content	**TEACHING HOURS**
UNIT 1: Unit Heading	**10Hrs**
Introduction to Computer Graphics, Applications of Computer Graphics, Graphic Display Devices: Refresh Cathode Ray Tubes, Raster-scan Displays, Random-Scan displays, Color CRT Monitors, Concept of Double Buffering, Lookup tables. 2-D Graphics: Cartesian and Homogeneous Coordinate Systems, Line drawing algorithms (Bressenham's and DDA), Circle and Ellipse Drawing Algorithms.	
UNIT 2: Unit Heading	**10Hrs**
2-Dimensional Transformations, Concepts of Window & Viewport, Window to Viewport Transformations, Normalization transformation (3L) Composite Transformations: General pivot point rotation, General fixed point scaling, reflection w.r.t line $y=x$, reflection w.r.t line y=x (4L) Transformation between coordinate systems, affine transformations, Raster methods for transformations (3L)	
UNIT 3: Unit Heading	**10Hrs**
Filling techniques: Boundary and Flood-fill algorithms (2L) Clipping, Line Clipping Algorithms (Cohen-Sutherland Algorithm), 3-D Graphics, Projections: perspective and parallel projection transformations. (5L) 3-Dimensional Transformations, Hidden Surface Removal Techniques, Z-Buffer Algorithm, Back Face Detection (3L)	
UNIT 4: Unit Heading	**10Hrs**
Curves and Surfaces: Spline specification, Interpolated& Approximated Splines. spline representation, cubic spline interpolation methods, Bezier Splines, Bezier Curves, Cubic Bezier Curves, Bezier Surfaces. (3LIntroduction to multimedia elements: Images (BMP, PCX), sound (WAV, MP3) Multimedia storage formats: CDs and DVDs).	

Lab Manual

Lab Sheet 1
Unit I:
Q1. Write a C++ program to draw line.
Q2. Write a C++ program to draw circle.
Q3. Write a C++ program to draw pixel.
Lab Sheet 2
Unit I:
Q1. Write a C++ program to draw line using DDA algorithm.
Q2. Write a C++ program to implement Brenham's algorithm to draw line.
Lab Sheet 3
Unit I:
Q1. Write a C++ program to implement Mid-Point Algorithm to draw Circle.
Q2. Write a C++ program to implement Mid-Point Algorithm to draw Ellipse.

Lab Sheet 1
Unit II:
Q1. Write a program to apply Translation to 2D shapes
Q2. Write a program to apply Scaling to 2D shapes
Q3. Write a program to apply reflection along X axis to 2D shapes
Q4. Write a program to apply reflection along Y axis to 2D shapes
Q5. Write a program to apply translation and reflection to 2D shapes

Lab Sheet 2
Unit II:
Q1. Write a program to apply rotation to 2D shapes
Q2. Write a program to apply X-shearing to 2D shapes
Q3. Write a program to apply Y-shearing to 2D shapes
Q4. Write a program to apply reflection along y=x line to 2D shapes
Q5. Write a program to apply translation and shearing to 2D shapes

Lab Sheet 3
Unit II:
Q1. Write a program to apply reflection along y=-x line to 2D shapes
Q2. Write a program to apply translation and rotation to 2D shapes
Q3. Write a program to apply scaling and shearing to 2D shapes
Q4. Write a program to apply scaling and translation to 2D shapes
Q5. Write a program to apply scaling and reflection to 2D shapes

Lab Sheet 1
Unit III:
Q1. Write a program to apply composite scaling and rotation to 2-Dimensional shapes.
Q2. Write a program to apply composite translation and rotation to 2-Dimensional shapes.
Q3. Write a program to clip the lines fallen outside the window using Cohen Sutherland line clipping.
Lab Sheet 2
Unit III:
Q1. Write a program to apply scaling and rotation to 3-Dimensional shapes.
Q2. Write a program to apply scaling and translation to 3-Dimensional shapes.
Q3. Write a program to apply translation and rotation to 3-Dimensional shapes.
Lab Sheet 3
Unit III:

Q1. Write a program to apply composite scaling and rotation to 3-Dimensional shapes.
Q2. Write a program to apply composite translation and rotation to 3-Dimensional shapes.
Q3. Write a program to apply composite translation and scaling to 3-Dimensional shapes.

Lab Sheet 1
Unit IV:
Q1. Write a program to implement line attributes.
Q2. Write a program to implement circle attributes.
Q3. Write a program to implement ellipse attributes.
Lab Sheet 2
Unit IV:
Q1. Write a program to draw Bezier Curve.
Q2. Write a program to draw Cubic Bezier Curve.
Lab Sheet 3
Unit IV:
Q1. Write a program to draw Bezier surfaces.
Q2. Write a program to generate fractal images.

Textbooks

1. Hearn and Baker "Computer Graphics" 2nd Edition, Pearson Education.
2. Fundamentals of Multimedia" by Ze-Nian Li and Mark S. Drew

Reference Books

1. W.M.Newman and Sproull. "Principles of interactive Computer Graphics" ,TMH
2. Steven Harrington." Computer Graphics a Programming Approach" McGraw Hill.
3. Plastock and Kelley. "Schaums outline of theory and problems of computer Graphics"
4. David F Frogers and J Alan Adams. "Procedural Elements of Computer Graphics" McGraw Hill
5. David F Rogers and J Alan Adams. "Mathematical Elements of Computer Graphics" McGraw Hill
6. James. D. Foley, A Van dam etal "Computer Graphics" Pearson.
7. Sinha and Udai , "Computer graphics", TMH

COURSE OUTCOMES (CO):

CO1: Apply Graphics Principles: Understand and apply core concepts of computer graphics and transformations.

CO2: Implement Algorithms: Develop and execute line drawing, clipping, and filling algorithms.

CO3: Create Curves and Surfaces: Design and manipulate graphical models using spline and Bezier techniques.

CO4: Manage Multimedia: Integrate and manage multimedia elements and file formats.

COURSE TITLE: Software Engineering

Course Code: MCA21204CR						**Examination Scheme**	
Total number of Lecture Hours: 40						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practicals(P):**	**-**	**Tutorial (T):**	**1**	**Total Credits**	**4**

Course Objectives

- **Understand Software Engineering Fundamentals**: Gain knowledge of the nature, goals, and challenges of software engineering and its historical context.
- **Apply Software Development Processes**: Learn and utilize various software development models, including Waterfall, Agile, and Spiral.
- **Measure Software Processes and Projects**: Analyze software processes using measures, metrics, and models like CMMI and COCOMO.
- **Master Requirements Engineering**: Develop skills in eliciting, analyzing, modeling, and validating both functional and non-functional requirements.
- **Design Engineering Proficiency**: Understand design principles, modularity, and patterns, and apply function-oriented and object-oriented design methodologies.
- **Achieve Competence in Software Testing and Reliability**: Understand core testing concepts and techniques, and explore software reliability and reengineering processes.

Course Content	**TEACHING HOURS**
UNIT 1: Fundamentals of Software Engineering	**10 Hrs**
Concept and Nature of Software: Concept and Nature of Software, Software Crisis, Software Engineering – Concept, Goals and Challenges, Software Engineering Approach. Software Development Process, Process Models - Waterfall Model, Evolutionary and Throwaway Prototyping Model, Incremental and Iterative Models, Spiral Model, Agile Process Model, Component based and Aspect Oriented development Software Process and Project Measurement: Measures, Metrics and Indicators, Size -Oriented Metrics vs. Function - Oriented Metrics, Capability Maturity Model Integration (CMMI). COCOMO Model.	
UNIT 2: Requirements Engineering	**10 Hrs**
Introduction to Requirements Engineering - Why, What and Where. Requirements Types: functional and nonfunctional requirements. Requirement Engineering Framework. Requirement Elicitation Process and Techniques. Requirement Analysis and Modelling, Requirements prioritization, verification, and validation.	
UNIT 3: Design Engineering	**10 Hrs**

Basics of Design Engineering - Abstraction, Architecture, Patterns, Separation of concerns, Modularity, Functional Independence, refinement, Refactoring. Function oriented design, Design principles, Coupling and Cohesion, Design Notations & Specifications, Structured Design Methodology. Object-Oriented Design - Design Concepts, Design Methodology, Object-oriented analysis and design modeling using Unified Modeling Language (UML), Dynamic & Functional Modeling, Design Verification.	
UNIT 4: Software Testing and Reliability	**10 Hrs**
Software Testing – Concepts, Terminology, Testing & Debugging, Adequacy Criteria, Static vs. Dynamic Testing, Black Box vs. White Box Testing. Structural testing and its techniques. Functional Testing and its techniques, Mutation testing, Random Testing. Non-Functional Testing like Reliability, Usability, Performance and Security Testing. Introduction to Software Reliability: Basic Concepts, Correctness Vs Reliability, Software Reliability metrics, Operational Profile, Reliability Estimation and Predication, Reliability and Testing. Concept of Software reengineering, reverse engineering and change management.	

Tutorial

UNIT I

Tutorial #1

a. How is Software Engineering different from other Engineering fields?

b. Study and compare different software process models

c. Identify the suitable applications for the individual process model.

Tutorial #2

a. Calculate the function points for the following data. The total CAV is 36. Number of user inputs=15 - Simple:- 5, Average:- 7, Complex:- 3 Number of user outputs=14 - Simple:- 5, Average:- 5, Complex:-4 Number of user inquiries=8 - Simple:- 2, Average:- 3, Complex:- 3 Number of files =6 - Simple:- 3, Average:- 1, Complex:- 2 Number of external interfaces=13 - Simple:- 4, Average:- 7, Complex:- 2

b. Based on the result calculate the various metrics like productivity, Quality, Cost, Documentation.

Tutorial #3

a. Calculate the effort, duration and average persons required for basic CoCoMo model for 70000 LOC assuming project type is semi-detached.

b. Calculate the effort, duration and average persons required for intermediate CoCoMo model for 50000 LOC assuming project type is organic and EAF is 2.92.

c. Calculate the effort, duration and average persons required in basic CoCoMo model for organic project type given that total FP is 651 and the 1 FP=2500 LOC

UNIT II

Tutorial #1

a Identify the different requirements of the application for application like Library Management System.

b. Identify the different requirements of the application for application like University System.

Tutorial #2

a Classify the requirements into functional and non-functional requirements for Library Mgmt. System.

b. Classify the requirements into functional and non-functional requirements for University System.

Tutorial #3

a. Prepare a requirement document (SRS) for the same as per the IEEE standard for Library Mgmt. System.

b. Prepare a requirement document (SRS) for the same as per the IEEE standard for university System.

UNIT III

Tutorial #1

Which of the following design principle(s) have been violated in the following scenarios?

a) Abstraction b) Decomposition and Modularization c) Coupling & Cohesion d) Encapsulation e) Sufficiency, Completeness and Primitiveness
f) All

i. An algorithm documented as part of design is not understandable by the programmers
ii. Important information of a module is directly accessible by other modules.
iii. Too many global variables in the program after implementing design.
iv. Unfulfilled requirements in the code after the design has been implemented.
v. Code breaks in unexpected places.
vi. All data of all classes in public.
vii. Cyclic dependencies among classes
viii. Huge class doing too many unrelated operations.
ix. Several unrelated functionalities/tasks are carried out by a single module.

Tutorial #2

Design the system using structured design for Library Management System by using DFD, ER diagrams and structure chart whichever applicable.

i. Identify various processes, data store, input, output etc. of the system.
ii. Use processes at various levels to draw the DFDs.
iii. Identify various modules, input, output etc. of the system
iv. Use various modules to draw structured charts.

Tutorial #3

Design the system using Object-Oriented design for Library Management System using UML modeling technique appropriately and

i. Identify various processes, use-cases, actors etc. of the system
ii. Ientify various elements such as classes, member variables, member functions etc. of the class diagram. Draw the class diagram.
iii. Identify various elements such as various objects of the object diagram. Draw the object diagram.
iv. Identify various elements states and their different transition of the state-chart diagram. Draw the state-chart diagram.
v. Identify various elements such as controller class, objects, boundaries, messages etc. of the sequence diagram. Draw the sequence diagram as per the norms.
vi. Identify various elements such as for the sequence diagram of the collaboration diagram. Draw the collaboration diagram as per the norms
vii. Identify various elements such as different activity their boundaries etc. of the activity diagram. Draw the activity diagram.
viii. Identify various elements of the component diagram such as the various components like client, server, network elements etc. Draw the component diagram.
ix. Identify various elements such as the hardware components of the deployment **diagram.**

Draw the deployment diagram.

UNIT IV

Tutorial #1

a. Write test cases for login page of your university admission system.

b. Write test cases for simple calculator program.

c. Write test cases for online examination module.

Tutorial #2

Due to surge in online examination requirements, a company is intending to test its software capable of examining 5000 students at a time for MCQs.

Indicate the performance testing strategy required to ensure that it is capable of supporting 5000 simultaneous users.

Tutorial #3

a. Calculate the reliability of the software product using sample data.
b. Calculate various reliability metrics using sample data and discuss applicability of each metric.

Textbooks

1. Pfleeger and Atlee, Software Engineering: Theory and Practice, 4th Edition, Pearson, 2010

Reference Books

2. Sommmerville, Ian - Software Engineering. Pearson , 9/e , 2011.
3. Pankaj Jalote - An Integrated approach to Software Engineering, Narosa Publication.
4. Software Engineering: Principles and practice, 3rd Edition, Hans Van Vliet, Wiley.
5. James F. Peters Software Engineering – An Engineering Approach, Wiley& Sons.
6. Roger Pressman, Software Engineering: A Practitioners Approach",McGraw-Hill Publications

COURSE OUTCOMES (CO):

CO1: Students will explain the nature of software, the software crisis, and the goals and challenges of software engineering.

CO2: Students will implement appropriate software development models such as Waterfall, Agile, and Spiral based on project needs.

CO3: Students will assess software processes using metrics and models like CMMI and COCOMO.

CO4: Students will perform requirement elicitation, analysis, modeling, prioritization, verification, and validation.

CO5: Students will apply design principles and object-oriented design methodologies using UML.

CO6: Students will execute various testing techniques and evaluate non-functional requirements like reliability and performance.

COURSE TITLE: Artificial Intelligence

Course Code: MCA21203CR						**Examination Scheme**	
Total number of Lecture Hours:						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practicals(P):**	**2**	**Tutorial (T):**	**0**	**Total Credits**	**4**

Course Objectives

- To present an overview of artificial intelligence (AI) principles and approaches .
- To develop a basic understanding of the building blocks of AI as presented in terms of Knowledge representation, inference, logic, and learning.
- To have understating of different search problems and their solution using various algorithms
- To have basic understanding of role of AI and ML along with IOT, expert systems

Course Content	TEACHING HOURS
UNIT 1:	**10-Hrs**
Introduction and historical perspective, Turing Test. Expert Systems, Forward chaining, backward chaining, Conflict Resolution. Agents: Intelligent agents, Agents and Environment, Structure of Agents Knowledge Representation: Propositional Logic, First Order Logic, Inference in First Order Logic, Prepositional Versus First Order Logic	
UNIT 2:	**10-Hrs**
Fuzzy Logic, Fuzzification, Fuzzy Sets, Operations on Fuzzy Sets, Hedges, Reasoning in Fuzzy Logic. Mamdani Inference Search Algorithms – Local search algorithms: Gradient ascent, Simulated Annealing, Genetic Algorithm.	
UNIT 3:	**10-Hrs**
Inductive Learning: Inductive learning algorithms. Categories of inductive learning algorithms. Rule extraction with inductive learning algorithms, Decision trees, ID3 algorithm. AQ algorithm, SAFARI algorithm Applications of Inductive Learning Machine Learning: Supervised, Unsupervised and Reinforcement Learning	
UNIT 4:	**10 Hrs**
Neural Networks: Neuron as a basic building element of an ANN. Activation functions, Perceptron. Learning with a perceptron. Limitations of a perceptron. Multilayer Neural Networks, Training by Error Back Propagation	

Self Organising Nets, Kohonen Self-Organising Net Convolutional Neural Networks	

Lab Manual

Unit I

Lab Sheet 1

1. Build an expert system and demonstrate forward chaining inferencing.

Lab Sheet 2

1. Build an expert system and demonstrate backward chaining inferencing.

Lab Sheet 3

1. Build an expert system and demonstrate conflict resolution process.

Unit II

Lab Sheet 1

1. Build a Fuzzy inference system for the Tipping Problem

Lab Sheet 2

1. Using Fuzzy Logic solve the following Tipping problem: Given two sets of numbers between 0 and 5 (where 0 is for very poor, and 5 for excellent) that respectively represent quality of service and quality of food at restaurant, what should tip be?

Lab Sheet 3

1. Solve 2-input 1-output project risk prediction problem using Mamdani Inference. Make necessary assumptions.

Unit III

Lab Sheet 1

1. Create a decision tree for a given dataset using ID3 algorithm

Lab Sheet 2

1. Implement Classification and Regression Tree (CART) algorithm for any relevant dataset.

Lab Sheet 3

1. Demonstrate inductive learning on any application of your choice.

Unit IV

Lab Sheet 1

1. Implement single layer perceptron.

Lab Sheet 2

1 Demonstrate Neural Networks using different activation functions

Lab Sheet 3

1. Implement Back-propagation Algorithm

Textbooks

Artificial Intelligence – A Modern Approach, Stuart Russel, Peter Norvig, PHI/Pearson Education.

Reference Books

1. Machine Learning by Tom M. Mitchel, McGraw-Hill publication
2. Introduction to Machine Learning by EthemAlpaydin, The MIT Press.
3. Artificial Intelligence and Expert Systems by Patterson PHI
4. Advances in Deep Learning by M. Arif Wani, Springer

COURSE OUTCOMES (CO):

Four to Six course outcomes to be listed by the course instructor

CO1: Apply AI techniques to solve problems of Game Playing, Expert Systems, Machine Learning.
CO2:. Formulate an efficient problem space for problem solving and Represent knowledge using the appropriate technique.
CO3: Design and develop expert systems to solve uncertainty problems.

CO4: Analyze real world problems and implement the concepts of AI in different domains

COURSE TITLE: Data Structures Using C++

Course Code: MCA21201CR						**Examination Scheme**	
Total number of Lecture Hours: 48						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practicals(P):**	**2**	**Tutorial (T):**	**0**	**Total Credits**	**4**

Course Objectives

- Understand and implement linear data structures such as arrays and linked lists, including operations like insertion, deletion, and searching.
- Master stack and queue operations, including their representations in memory and implementations using arrays and linked lists.
- Comprehend tree structures including binary trees, binary search trees, AVL trees, and B-trees, along with their traversal techniques and applications.
- Learn graph terminology, representations, traversal techniques, and practical applications in computer science.
- Explore advanced data structures such as threaded binary trees, M-way search trees, and various types of heaps.
- Study file organization techniques including sequential, relative, and indexed sequential file organizations, as well as multiple key file organizations like inverted files and multi-list organizations.

Course Content	**TEACHING HOURS**
Unit I: Linear Data Structures	**12 Hrs**
Data types/objects/structures, Data structures and its types, Representation and implementation. Linear Data Structures: Array representation, operations, applications and limitations of linear arrays, 2- dimensional arrays, matrices, common operations of matrices, special matrices, array representation of Sparse matrices. Linked Lists: Representation of Linear Linked List, Operations like creating, search an element inserting an element, deleting an element, reversing a list, merging two list, Deleting entire list. Linked list application, Polynomial Manipulation, Representing Sparse Matrices	
Unit II: Stack and Queues	**12 Hrs**
Stack, Representation of stack in memory, Operations on Stacks, Implementation of Stack using arrays and linked list, Multiple Stacks: Representing two stacks and more than two stacks, Applications of stacks: Parenthesis Checker, Infix to postfix procedure, Evaluating expressions in postfix notation, Sparse Matrix Representation. Implementation of recursion using stack, Queues, Representation of Queue in Memory, Operations on Queue, Implementation of Queue using arrays and linked list, Circular Queue and its operations, Representation and implementation, Multiple Queues, DEQUE, Priority Queue, ,Linked Queue, Multiple Priority queue, Heap Representation of a Priority Queue, Applications of Queues.	

Unit III: Tree and Graph Data Structures	**12 -Hrs**
Trees, Definitions, terminologies and properties, Binary tree representation, traversals and applications, Threaded binary trees, Binary Search Trees, AVL Trees, M-way Search Trees, B-trees, B*-trees. Graphs, Terminology, Graph representations, Traversal Techniques, Operations on Graphs, Applications of Graphs	
Unit IV: Advanced Data Structures and Algorithms	**12 - Hrs**
Minimum spanning trees, Shortest Path Algorithms in Graphs, Eulerian Tour, Hamiltonian Tour Direct Address Tables, Hash Table, Different Hash functions, resolving collisions, rehashing, Heap Structures, Binomial Heaps, Leftist Heaps. File Organizations: Sequential File Organization, Relative File Organization, Indexed Sequential File Organization, Multiple Key File Organizations: Inverted File and Multi-List Organizations.	

Lab Manual

UNIT I

Lab Sheet 1:

Q1. Write a program in C++ to insert, delete and update the contents of an array.

Q2. Write a program in C++ to perform various operations on matrices.

Q3. Write a program to multiply two sparse matrices?

Q4. Write a program in C++ to implement different string manipulation operations?

Lab Sheet 2:

Q1. Write a program to implement singly linked list?

Q2. Write a program to implement different operations like adding a node at beginning, end, center, after a certain element, after a certain count of nodes in a linkedlist.

Q3. Write a program to implement different operations like deleting a node at beginning, end, center, after a certain element, after a certain count of nodes in a linkedlist.

Q4. Write a program in C++ to reverse a linked list by changing the link in the nodes?

Q5. Write a program to add two polynomials represented as linked list?

Lab Sheet 3:

Q1. Write a program in C++ to multiply two polynomials represented as linked lists?

Q2. Write a program in C++ to implement a doubly linked list?

Q3 Write a program to implement different operations like adding a node at beginning, end, center, after a certain element, after a certain count of nodes in a doubly linkedlist.

Q4. Write a program to implement different operations like deleting a node at beginning, end, center, after a certain element, after a certain count of nodes in a doubly linkedlist.

Q5 Write a program to implement different operations of a circular linked list.

UNIT II

Lab Sheet 1:

Q1. Write a program to implement various operations on an array based stack?

Q2. Write a program to implement various operations on an stack represented using linked list.

Q3. Write a program to demonstrate the use of stack in checking whether the arithmetic expression is properly parenthesized?

Q4. Write a program to demonstrate the use of stack in converting an arithmetic expression from infix to postfix?

Q5. Write a program to demonstrate the use of stack in evaluating an arithmetic expression in postfix notation?

Lab Sheet 2:

Q1. Write a program to demonstrate the use of stack in implementing quicksort algorithm to sort

an array of integers in ascending order?
Q2. Write a program to demonstrate the implementation of various operations on a linear queue represented using a linear array.
Q3. Write a program to demonstrate the implementation of various operations on a Circular queue represented using a linear array.

Lab Sheet 3:

Q1. Write a program to demonstrate the implementation of various operations on a queue represented using a linked list?
Q2. Write a program to demonstrate the use of multiple stacks?

UNIT III

Lab Sheet 1:

Q1. Write a program in C++ to create a binary tree?
Q2. Write a program to implement the traversal techniques of a binary tree?

Lab Sheet 2:

Q1. Write a program to delete a node in a binary search tree?
Q2. Write a program to implement the different operations of an AVL tree?
Q3. Write a program to implement the different operations of a threaded binary tree.
Q4. Write a program to implement the different operations of a M-way search tree?

Lab Sheet 3:

Q1. Write a program to implement the different operations of a B- tree?
Q2. Write a program in C++ to implement the different operations of a B+tree?
Q3. Write a program in C++ to implement the different operations of a B* tree?
Q4. Write a program in C++ to Multi-dimensional binary searchtrees.
Q5.Write a program in C++ to implement the graph using different representations?
Q6. Write a C++ program to illustrate the traversal of a graph using Breadth FirstSearch? Q7. Write a C++ program to illustrate the traversal of a graph using Depth FirstSearch?

UNIT IV

Lab Sheet 1:

Q1. Write a program in C++ to find the edges of a spanning tree using Prims Algorithm? Q2. Write a program in C++ to find the shortest path in a graph using Warshalls Algorithm. Q3. Write a C++ program to in C++ to find the shortest path in a graph using Modified Warshalls Algorithm. Q4. Write a C++ program to in C++ to find the shortest path in a graph using Dijkstra's Algorithm.
Q5. Write a C++ program in C++ to implement Euler Graphs?

Lab Sheet 2:

Q1. Write a program in C++ to implement Hamilton Graphs?
Q2. Write a program in C++ to implement Planner Graphs?
Q3. Write a program to C++ to implement Kruskals Algorithm?
Q4. Write a program to C++ to find the cycles in a graph?

Lab Sheet 3:

Q1. Write a C++ program to implement various hashing techniques?
Q2. Write a C++ program to demonstrate the concept of rehashing?
Q3. Write a C++ program to create Max and Min heaps?
Q4. Write a C++ program to create Binomial and Leftist heaps?

Textbooks

2. SartajSahni, "Fundamentals of Data Structures in C++", Galgotia Pub

Reference Books

6. M.Radhakrishnan and V.Srinivasan, "Data Structures Using C" ISTE/EXCEL BOOKS
7. Weiss Mark Allen, "Algorithms, Data Structures, and Problem Solving with C++", Addison Wesley.
8. O.G. Kakde& U.A. Deshpandey, "Data Structures and Algorithms" ISTE/EXCEL BOOKS

9. Aho Alfred V., Hopperoft John E., UIlmanJeffreyD., "Data Structures and Algorithms", Addison Wesley
10. Drozdek, "Data Structures and Algorithms", Vikas,
11. Tanenbaum A. S. , "Data Structures using 'C' "

COURSE OUTCOMES (CO):

CO1: Students will be able to implement and manipulate linear data structures such as arrays, linked lists, and matrices, including operations like insertion, deletion, and traversal.

CO2: Students will demonstrate proficiency in implementing and applying advanced data structures such as stacks, queues, trees (binary trees, AVL trees), graphs, and various heaps (binomial heaps, leftist heaps) to solve complex problems.

CO3: Students will understand and apply different file organization techniques such as sequential, relative, and indexed sequential file organizations, and multiple key file organizations like inverted files and multi-list organizations.

CO4: Students will develop analytical and problem-solving skills by applying appropriate data structures and algorithms to solve practical problems related to data storage, retrieval, and manipulation in computer science applications.

COURSE TITLE: Python

Course Code: MCA21202CR						**Examination Scheme**	
Total number of Lab Hours:						**External**	**40**
						Internal	**10**
Lecture (L):	**1**	**Practicals(P):**	**2**	**Tutorial (T):**	**-**	**Total Credits**	**2**

Course Objectives:

- Develop Proficiency in Python Programming: Equip students with fundamental programming skills in Python, focusing on syntax, control structures, data types, and basic operations.
- Apply Advanced Python Concepts to Real-World Problems: Enable students to use advanced Python features, including object-oriented programming, data analysis libraries, and web scraping techniques, to solve practical problems.
- Enhance Problem-Solving and Analytical Skills: Strengthen students' abilities to think logically and analytically by solving programming challenges and developing projects that require the application of Python in various contexts.

Course Content	TEACHING HOURS
Unit I:	**12 Hrs**
Understanding Python variables, Python basic Operators, python blocks , Data Types, Declaring and using Numeric data types: int, float, complex Using string data type and string operations Defining list and list slicing Use of Tuple data type: Python Program Flow Control Conditional blocks using if, else and elif Simple for loops in python, For loop using ranges String, list and dictionaries Use of while loops in python Loop manipulation using pass, continue, break and else Programming using Python conditional and loops block	
Unit II:	**12 Hrs**
Python Functions, Modules And Packages, Organizing python codes using functions Organizing python projects into modules, Importing own module as well as external modules Understanding Packages Powerful Lamda function in python, Programming using functions, modules and external packages, Python String, List And Dictionary Manipulations.	

Lab Manual

- **Week 1**

1. Install Python and set up IDEs like Jupyter Notebook or VS Code
2. Write a "Hello, World!" program.
3. Write a program to perform basic arithmetic operations: addition, subtraction, multiplication, and division.
4. Write a program to print your name and age.

- **Week 2**

1. Write a program to create variables of different data types (int, float, complex, string) and print their values.
2. Write a program to perform string operations: concatenation, slicing, and repetition.
3. Write a program to demonstrate arithmetic, logical, and relational operations.

- **Week 3**

1. Write a program to create a list, perform slicing, and append elements to it.
2. Write a program to demonstrate the use of tuple data type and its operations.
3. Write a program to find the length, maximum and minimum value of a list.

- **Week 4**

1. Write a program to demonstrate the use of if, else, and elif statements.
2. Write a program to print the first 10 natural numbers using a for loop.
3. Write a program to print a pattern using nested loops (e.g., a pyramid).

- **Week 5**

1. Write a program to iterate over a string, list, and dictionary using loops.
2. Write a program to demonstrate the use of while loops.
3. Write a program to manipulate loops using pass, continue, break, and else.

- **Week 6**

1. Write a program to read a user's name and print a greeting message.
2. Write a program to read from and write to a text file.
3. Write a program to take multiple inputs from the user and print them.

- **Week 7**

1. Write a program to define and call a function that adds two numbers.
2. Write a program to demonstrate the use of lambda functions.
3. Write a program with a function that takes a list as an argument and returns the sum of all its elements.

- **Week 8**

1. Write a program to create and import a custom module.
2. Write a program to use an external library (e.g., math or random).
3. Write a program to organize code into a package.

- **Week 9**

1. Write a program to define a class and create objects.
2. Write a program to demonstrate inheritance.
3. Write a program to show polymorphism using method overriding.

- **Week 10**

1. Write a program to read and write to a file using different modes.
2. Write a program to handle exceptions using try, except, and finally blocks.
3. Write a program to create a directory, write a file in it, and handle any exceptions that occur.

- **Week 11**

1. Write a program to find all occurrences of a pattern in a string using regular expressions.
2. Write a program to validate an email address using regular expressions.
3. Write a program to replace all instances of a substring in a string using regular expressions.

- **Week 12**

1. Write a program to connect to a MySQL database.
2. Write a program to perform basic CRUD operations in a MySQL database.
3. Write a program to retrieve and display data from a database.

- **Week 13**

1. Write a program to perform basic array operations using NumPy.
2. Write a program to create and manipulate DataFrame objects using Pandas.
3. Write a program to plot a simple graph using Matplotlib.
4. Write a program to perform a basic statistical analysis using SciPy.

Textbook: Kenneth A. Lambert, The Fundamentals of Python: First Programs, Cengage Learning, ISBN: 978-1111822705.

Reference Books:

1. David Beazley , Brian K. Jones “Python Cookbook”, 3rd Edition. O’Reilly Publications
2. Jake VanderPlas “Python Data Science Handbook” O’Reilly Publications
3. David Beazley, “Python Essential Reference (4th Edition) “ Addison Wesley
4. Vernon L. Ceder,” The Quick Python Book, Second Edition”, Manning Publications
5. Brett Slatkin ,”Effective Python”

COURSE OUTCOMES (CO):

CO1: Students will be able to write, debug, and execute Python programs, effectively using functions, data structures, and file handling operations.

CO2: Students will design and implement a project that integrates various Python concepts, demonstrating their ability to apply their knowledge to real-world applications.

CO3: Students will exhibit strong problem-solving skills, utilizing Python to tackle complex challenges and develop efficient, effective solutions.

COURSE TITLE: Cryptography and Network Security

Course Code:MCA21206DCE						**Examination Scheme**	
Total number of Lecture Hours:48						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practical's (P):**	**2**	**Tutorial (T):**	**-**	**Total Credits**	**4**

Course Objectives

- ✓ **To gain a comprehensive understanding of the OSI Security Architecture and fundamental security concepts.**
- ✓ **To develop proficiency in cryptographic techniques and number theory.**
- ✓ **To master key management and authentication protocols.**
- ✓ **To apply cryptographic methods to network security and intrusion detection.**

Course Content	TEACHING HOURS
UNIT 1:	**12 Hrs**
Part 1: The OSI Security Architecture, Security Attack – Threats, Vulnerabilities, and Controls, Types of Threats (Attacks) Part 2: Security Services – Confidentiality, Integrity, Availability, Authentication, Access Control and Non-repudiation; Security Mechanism. Part 3: Introduction to Number Theory: Prime Number Generation and Testing for Primality, Fermat's and Euler's Theorems, Modular Arithmetic, Euclidean and Extended Euclidean Algorithm, Euler's Phi Function.	
UNIT 2:	**12 Hrs**
Part 1: Introduction to Cryptology. Types of Encryption Systems – Based on Key, Based on Block; Confusion and Diffusion; One-time pad, Block Ciphers and Data Encryption Standard Part 2: Block Cipher Modes of operation, Advanced Encryption Standard. Stream Ciphers, Random Number Generation. Shift Register based stream Ciphers, RC4 Part 3: Public-Key Cryptography. RSA Cryptosystem	
UNIT 3:	**12 Hrs**
Part 1: Double and Triple Encryption. Key Management, Diffie-Hellman Key Exchange Part 2: Digital Signatures, The RSA signature scheme, Hash Functions, The Secure Hash Algorithm SHA-1 Part 3: Message Authentication Codes, HMAC and CBC-MAC, Message Digest	
UNIT 4:	**12 Hrs**
Part 1: IP Security, Authentication Header, Encapsulating Security Payload, Electronic Mail Security Part 2: Network intrusion Detection system using machine learning: Supervised and Unsupervised. General IDS model and Taxonomy. IDS Signatures. Part 3: DDoS Attacks. Specification and rate based DDoS. Defending against DoS attacks in scout: signature based solutions.	

Lab Manual

Unit I

Week 1:

Network Troubleshooting commands

a. Ipconfig, Ping, Traceroute, Netstat b. NSLookUp, ARP, Hostname

Week 2:

Demonstrate Packet Sniffing and Analysis of Network Traffic
a. Analyse Network Traffic using Wireshark
b. Demonstrate the Analysis of Network traffic over HTTP protocol

Week 3:

Demonstrate Network Penetration Testing
a. Demonstrate use of various Network Penetration testing tools
b. Use Hashcat to crack hashes on passwords.

Unit II

Week 4:

Substitution Ciphers
a. Discuss in detail working of Substitution based cryptographic Primitives
b. Implement Caesar Cipher Encryption Decryption

Week 5:

Polyalphabetic Substitution Ciphers
a. Demonstrate Symmetric Digraph Substitution
b. Implement Playfair Cipher and Vigenere Encryption Decryption

Week 6:

Transposition Cipher
a. Demonstrate Transposition ciphers.
b. Implement Vigenere Cipher and Rail Fence (Row Column Transformation)

Unit III

Week 7:

Asymmetric Key Cryptography
a. Discuss the implementation of Public key Cryptography
b. Implement RSA Algorithm

Week 8:

Symmetric Key Cryptography with Stream Ciphers
a. Demonstrate working of Stream Ciphers
b. Implement RC4 Algorithm

Week 9:

Symmetric Key Cryptography with Block Ciphers
a. Discuss in detail the implementation details of Block Ciphers
b. Implement DES Algorithm

Unit IV

Week 10:

Intrusion Detection
a. Demonstrate Anomaly Detection and its various types
b. Perform Intrusion Detection System using SNORT

Week 11: IDS using Machine learning Algorithms
a. Discuss Various classifiers for Intrusion Detection
b. Implement IDS using Random Forest on NSL KDD DataSet.

Week 12:

Network Testing
a. Discuss Security Auditing and Network Discovery
b. Implement Vulnerability Scanning using NMAP tool

Software's / Tools Required.

1. Wireshark
2. SNORT
3. Net Stumbler
4. NMAP
5. Python
6. Hashcat

7. Turbo C/ JAVA / Python

Textbooks

- Paar, Christof, and Jan Pelzl. Understanding cryptography: a textbook for students and practitioners. Springer Science & Business Media, 2009.
- William, S., and Cryptography Stalling. "Network Security, 4/E." Prentice Hall. (2006).
- Forouzan, Behrouz A., and Debdeep Mukhopadhyay. Cryptography and network security (Sie). McGraw-Hill Education, 2011. • Endorf, C., Schultz E and Mellander J, "Intrusion Detection and prevention". McGraw Hill. 2003

Reference Books

- Paar, Christof, and Jan Pelzl. Understanding cryptography: a textbook for students and practitioners. Springer Science & Business Media, 2009.
- Introduction to Modern Cryptography (Chapman & Hall/CRC Cryptography and Network Security Series) Jonathan Katz , Yehuda Lindell

COURSE OUTCOMES (CO):

Upon successful completion of this course, learners will be able to:

1. **Explain the fundamental concepts of cryptography**, including symmetric and asymmetric encryption, hashing, digital signatures, and key management.
2. **Understand the historical development and relevance of cryptographic techniques** in modern security protocols.
3. **Apply various cryptographic algorithms**, such as AES, RSA, ECC, and SHA, to secure data and communications.
4. **Analyze and critically evaluate the strengths and weaknesses of different cryptographic protocols**, such as SSL/TLS, IPsec, and PGP.
5. **Design and implement network intrusion detection systems**, integrating cryptographic solutions to protect against various security threats and attacks.

COURSE TITLE: Web Programming

Course Code: MCA21205DCE						**Examination Scheme**	
Total number of Lecture Hours: 48						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practicals(P):**	**2**	**Tutorial (T):**		**Total Credits**	**4**

Course Objectives:

- Gain a comprehensive understanding of fundamental web technologies, including HTML, XHTML, and CSS.
- Learn the principles of responsive and accessible web design using CSS and various layout techniques.
- Develop proficiency in JavaScript programming for client-side web development, including DOM manipulation and event handling.
- Acquire skills in server-side scripting using PHP to create dynamic and interactive web applications.
- Understand how to integrate and manage databases within web applications using MySQL.
- Combine client-side and server-side technologies to build complete, functional web applications.

Course Content	**TEACHING HOURS**
UNIT 1:	**-Hrs**
Adobe Photoshop Environment, Interface tour of Photoshop and Palettes, Color Modes and Resolutions, Using different Photoshop tools. Working with Layers Grouping and Smart objects, Image Adjustments, Layer Masking and Layer Clipping, Using Blending Options, Filters, Photoshop actions, Animation tools Markup Language, Basic Structure of HTML , Meta Tags, Document Structure Tags, Formatting Tags, Text Level formatting, Block Level formatting, List Tags, Hyperlink tags, Image and Image maps, Table tags, Form Tags, Executable content tags, Tables as a design tool, Forms, Creating Forms.	12
UNIT 2:	**- Hrs**
Style Sheets: Different approaches to style sheets, Using Multiple approaches, Linking to style information in s separate file, Setting up style information. Java Script: JavaScript Objects, JavaScript Security, Operators: Assignment Operators, Comparison Operators, Arithmetic Operators, Logical Operators, String Operators, Special Operators, ? (Conditional operator), ,(Comma operator), delete, new, this, void Statements : Break, comment, continue,	12

delete, do … while, export, for, for…in, function, if…else, import, labelled, return, switch, var, while, with, Core JavaScript (Properties and Methods of Each) : Array, Boolean, Date, Function, Math, Number, Object, String, regExp Document Object Model, Events and Event Handlers.	
UNIT 3:	**-Hrs**
PHP, Server-side web scripting, Installing PHP, Adding PHP to How PHP scripts work, Basic PHP syntax PHP data types, PHP Variables, Operators in PHP, Conditional Statements, Loops (If, If else and Switch) Strings, Arrays and Array Functions, Numbers, PHP Function: User-Defined Functions, Inbuilt functions, Basic PHP errors / problems, Working with Forms, designing a Form, $_GET and $_POST, HTML and PHP code, User Input, Form Validation, Cookies, File uploading, Sessions	12
UNIT 4:	**-Hrs**
Advanced PHP and MySQL: PHP MySQL Integration, Creating a database connection, Selecting the DB, Basics of SQL, SQL Syntax, CRUD Operations, Inserting data in database, Inserting data with a File Retrieving data from Database, Retrieving data with specific criteria, Updating records, Searching the records, Alter table structure, Deleting the records Dropping tables. Emailing with PHP.	12

Lab Manual

Unit I

Week 1:

1. Open any picture and make use of rectangular and elliptical selection tools to select portions of the image and paste it in another image. Also make use of move tools.
2. Make use of the Lasso- and Polygonal Lasso Selection Tools, Copy, Paste Into, Move Tool, Zoom Tool, Quick Select Tool (or Magic Wand Tool), Invert Selection, Copy, Paste Transform tools for editing an image.
3. Edit any image using the following tools, Paint Bucket Tool, Color Picker, Brush Tool.
4. Select an image and make use of Text Tool, Selection Tools, Copy, Paste, Transform, Move Tool, Opacity, Eraser Tool to perform different operations
5. Select any image of your choice and make use of the Brush Tool, Smudge Tool, Dodge Tool, Burn Tool, Layer Styles, Modes, The Shape Tools, the Styles palette.

Week 2

1. Applying different filters on an image and make use of different layers.
2. Create a page banner from scratch using browser-safe colors
3. Make the illusion of an image fitting inside your text using clipping mask.
4. Create an Animation for Rocket Launch and Moving Ball

Week 3

1. Create a html page with demonstrates the use of formatting tags image tags and other basic tags.
2. Create the different types of list, tables in html
3. Create a table with the relevant tags and attributes
4. Create a html form in the table layout covering major form elements

Unit II

Week 4

1. Link an external style sheet with styles for basic tags.
2. Create a CSS code for applying design on the webpage.
3. Using a DIV tag and CSS code design a web page.
4. Create a CSS code and use id and Class identifiers.

Week 5

1. Write a JavaScript program to sum the multiples of 3 and 5 under 1000?
2. Write a JavaScript Code for checking type of triangle where three sides are given.
3. Write a JavaScript code to convert a Decimal Number int Roman Number?
4. Write a JavaScript function to test whether a string ends with a specified string

Week 6

1. Write a JavaScript to check whether a given string is palindrome or not.
2. Write a program using Java Script that checks if two matrices have identical values in all the elements
3. Write a JavaScript program to check a credit card number and validate an email address using JavaScript Regular Expressions?
5. Write a JavaScript program to implement DOM?

Unit -III

Week 7

1. Create a simple HTML form and accept the user name and display the name through PHP echo statement
2. Write a PHP program to remove duplicates from a sorted list.
3. Write a PHP program to compute the sum of the prime numbers less than 100
4. Write a PHP program to print out the sum of pairs of numbers of a given sorted array of positive integers which is equal to a given number?

Week 8

1. Write a program to calculate and print the factorial of a number using a for loop.
2. Write a PHP script using nested for loop that creates a chess board?
3. Write a program that inputs a number from the user and display all armstong numbers upto the number entered using loops?
4. Write a function to reverse a string.

Week 9

1. Write a PHP code to Validate and form and provide results on the other web page
2. Wite a PHP code to implement various string functions used in PHP.
3. Write a PHP code for uploading a file in a specific folder on the server.
4. Write a PHP code so sort an array using any sorting technique?

Week 10

1. Write a PHP script to get time difference in days and years, months, days, hours, minutes, seconds between two dates
2. Write a PHP function to get start and end date of a week (by week number) of a particular year
3. Write a PHP script to generate random 11 characters string of letters and numbers
4. Write a PHP function to create a human-readable random string for a captcha.

Unit - IV

Week 11

1. Write the mysql code to create the database represented by following E-R diagram . Keep all the referential integrity constraints into consideration?
2. Insert the dummy data inside the tables making any assumptions as required if any ?
3. Write a SQL statement to insert records into the table countries to ensure that the country_id column will not contain any duplicate data and this will be automatically incremented and the column country_name will be filled up by 'N/A' if no value assigned for that column.
4. Write a SQL statement to insert rows in the job_history table in which one column job_id is containing those values which are exists in job_id column of jobs table.
5. Write a SQL statement to insert rows into the table employees in which a set of columns department_id and manager_id contains a unique value and that combined values must have exists
into the table departments.
6. Write a SQL statement to insert rows into the table employees in which a set of columns department_id and job_id contains the values which must have exists into the table departments

and jobs.

Week 12

1. Write a query to display the name (first_name, last_name) and salary for all employees whose salary is not in the range $10,000 through $15,000.
2. Write a query to display the name (first_name, last_name) and salary for all employees whose salary is not in the range $10,000 through $15,000 and are in department 30 or 100.
3. Write a query to display the first_name of all employees who have both "b" and "c" in their first name.
4. Write a query to get the total salaries payable to employees.
5. Write a query to get the minimum salary from employees table.
6. Write a query to get the maximum salary of an employee working as a Programmer.
7. Write a query to get the average salary and number of employees working the department 90.
8. Write a query to find the name (first_name, last_name) and hire date of the employees who was hired after 'Jones'.
9. Write a query to get the department name and number of employees in the department
10. Write a query to find the employee ID, job title, number of days between ending date and starting date for all jobs in department 90.
11. Write a query to display the department ID and name and first name of manager.
12. Write a query to display the department name, manager name, and city.
13. Write a query to display the job title and average salary of employees.
14. Write a query to display job title, employee name, and the difference between salary of the employee and minimum salary for the job
15. Write a query to get the DATE value from a given day (number in N).
16. Write a query to get the firstname, lastname who joined in the month of June.
17. Write a query to get the years in which more than 10 employees joined.
18. Write a query to get first name of employees who joined in 1987.
19. Write a query to get department name, manager name, and salary of the manager for all managers whose experience is more than 5 years.
20. Write a query to get employee ID, last name, and date of first salary of the employees.
21. Write a query to get first name, hire date and experience of the employees
22. Write a query to get the department ID, year, and number of employees joined.
23. Write a query to update the portion of the phone_number in the employees table, within the phone number the substring '124' will be replaced by '999'.
24. Write a query to get the details of the employees where the length of the first name greater than or equal to 8.
25. Write a query to display the first word from those job titles which contains more than one words
29. Write a query to display the first eight characters of the employees' first names and indicates the amounts of their salaries with '$' sign. Each '$' sign signifies a thousand dollars. Sort the data in descending order of salary.
26. Write a query to display the employees with their code, first name, last name and hire date who hired either on seventh day of any month or seventh month in any year

Week 13.

1. Create a PHP-MySQL connection which connects to the hr database using PHP objects ?
2. Create a form to add using sign in and sign out, update and delete employee to the hr database?
3. Create a login, logout for every employee and list all the employee in the database?
4. Write a php script which emails the login details to the new employee along with his salary

details fetch from the hr database?
5. Write a php script which demonstrates the use of sessions and cookies which inserting in the database?

Textbooks

1. Learning PHP, MySQL & JavaScript by Robin Nixon, O'Reilly Media.
2. JavaScript and JQuery: Interactive Front-End Web Development by Jon Duckett

Reference Books

1. Web Design The complete Reference, Thomas Powell, Tata McGrawHill
2. HTML and XHTML The complete Reference, Thomas Powell, Tata McGrawHill
3. JavaScript 2.0 : The Complete Reference, Second Edition by Thomas Powell and Fritz Schneider
4. PHP: The Complete Reference By Steven Holzner, Tata McGrawHill

COURSE OUTCOMES (CO):

CO1: HTML and XHTML Proficiency: Students will be able to create well-structured HTML and XHTML documents, utilizing semantic elements and multimedia integration.

CO2: CSS Styling and Layout: Students will demonstrate the ability to apply CSS for styling web pages, including advanced techniques like Flexbox and Grid layouts.

CO3: JavaScript Development: Students will be able to write JavaScript code to handle user interactions, manipulate the DOM, and implement AJAX for asynchronous data fetching.

CO4: PHP and Form Handling: Students will be proficient in writing PHP scripts for server-side processing, form handling, and user input validation.

CO5: Database Operations: Students will be able to connect web applications to MySQL databases, perform CRUD operations, and manage data securely.

CO6: Complete Web Application Development: Students will be capable of developing and deploying full-stack web applications that integrate HTML, CSS, JavaScript, PHP, and MySQL.

COURSE TITLE:WEB DESIGNING

Course Code: MCA21201OE						**Examination Scheme**	
Total number of Lecture Hours: 20						**External**	**40**
						Internal	**10**
Lecture (L):	**2**	**Practicals(P):**	**Tutorial (T):**	**-**	**Total Credits**		**2**

Course Objectives

- Develop proficiency in HTML for structuring content and CSS for styling, including layout, typography, and responsive design.
- Learn to implement interactivity using JavaScript and enhance user experience through animations and dynamic content.
- Develop a portfolio of diverse web projects demonstrating design skills, creativity, and practical application of web design principles.

Course Content	TEACHING HOURS
UNIT 1: HTML	**10- Hrs**
HTML: Understanding HTML, create a Web Page, Linking to other Web Pages, Publishing HTML Pages, Text Alignment and Lists, Text Formatting Fonts Control, Hyper Links and link within a Page, Creating HTML Forms, Creating Web Page Graphics, Putting Graphics on a Web Page, Custom Backgrounds and Colors.	
UNIT 2: CSS	**10- Hrs**
Concept of CSS, Creating Style Sheet, CSS Properties, CSS Styling (Background, Text Format, Controlling Fonts), Working with block elements and objects, Working with Lists and Tables, CSS Id and Class, Box Model (Introduction, Border properties, Padding Properties, Margin properties), Creating page Layout and Site Designs.	

Textbooks

HTML and CSS: Design and Build Webs by Jon Duckett

Reference Books

- Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics, Fifth Edition Robbins, Jennifer
- Schaum's Outline of HTML by David Mercer

COURSE OUTCOMES (CO):

CO1: Proficiency in Web Technologies: Students will demonstrate proficiency in HTML, CSS, and JavaScript, enabling them to create well-structured, visually appealing, and interactive websites.

CO2: Responsive Design Skills: Students will be able to design and develop responsive websites that provide optimal viewing and interaction experience across a wide range of devices and screen sizes.

CO3: Enhanced User Experience: Students will apply design principles and interactive elements effectively to enhance user experience and usability of websites.

SEMESTER III

COURSE TITLE: Design and Analysis of Algorithms

Course Code:	**MCA21301CR**					**Examination Scheme**	
Total number of Lecture Hours: 48						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practicals(P):**	**2**	**Tutorial (T):**		**Total Credits**	**4**

Course Objectives

- Gain a solid foundation in algorithms, their analysis, and the growth of functions.
- Apply asymptotic notations and techniques to study the time and space complexity of algorithms.
- Explore and apply methods such as recurrences, the Master Method, and randomized algorithms.
- Utilize divide and conquer, greedy, dynamic programming, backtracking, and branch and bound strategies to solve complex problems.
- Learn about P, NP, NP-hard, and NP-complete problems, and understand the significance of Cook's Theorem.
- Evaluate the need for and implement approximation algorithms for solving complex optimization problems.

Course Content	**TEACHING HOURS**
UNIT 1: Fundamentals of Algorithm Analysis	**12 Hrs**
Introduction to Algorithms, Analysis of Algorithms, Growth of Functions, Asymptotic notations (3L) Recurrences, Substitution method, Iteration method, Recursion trees (4L) The Master Method, Time and Space Complexity study of some basic algorithms. (3L)	
UNIT 2: Advanced Algorithmic Techniques	**12 Hrs**
Randomized Algorithms: Identifying the repeated element, Primality testing, Advantages and Disadvantages. (3L) Divide and Conquer Strategy: Binary search, Quick sort, Merge sort (3L) Greedy Method, General method, Knapsack problem, Single source shortest paths.(4L)	
UNIT 3: Optimization and Search Strategies	**12 Hrs**
Dynamic programming Strategy: All pair shortest paths, Traveling salesman problems. (3L) Backtracking Strategy: 8-Queen problem, Sum of subsets, Knapsack problem.(4L) Branch and Bound Strategy: Least Cost Branch and Bound, 8-Queen Problem(3L)	
UNIT 4: Computational Complexity and Approximation Algorithms	**12 Hrs**

Lower boundary theory, Lower bound theory through reductions, P and NP problems. NP hard and NP complete problems, Cook's Theorem (5L) Approximate Algorithms and their need, The vertex Cover Problem, The traveling salesman problem, The subset sum problem (5L)	

LAB MANUAL

Unit 1:

LabSheet1:

1. Write a program for Linear Search.
2. Implement recursive solution to the Tower of Hanoi puzzle.

LabSheet2:

1. Write a program for iterative binary search.
2. Sort a given set of n integer elements using Quick Sort method and compute its time complexity. Run the program for varied values of n > 5000 and record the time taken to sort. Plot a graph of the time taken versus n on graph sheet. The elements can be read from a file or can be generated using the random number generator. Demonstrate using C how the divide -and- conquer method works along with its time complexity analysis: worst case, average case and best case.

LabSheet3:

1. Print all the nodes reachable from a given starting node in a digraph using BFS method.
2. Obtain the Topological ordering of vertices in a given digraph

Unit 2:

LabSheet1:

1. Write a program for recursive binary search.
2. Write a program for Merge Sort.

LabSheet2:

1. Write a program for finding maximum and minimum number using Divide and conquer method.
2. Write a program to sort given set of elements using heap

. LabSheet3:

1. Implement Knapsack Problem using greedy method.
2. Write a program for Single Source Shortest path algorithm using greedy method.

Unit 3:

LabSheet1:

1. Implement 0/1 knapsack using dynamic programming.
2. Write a program for travelling salesman problem using Dynamic programming.

LabSheet2:

1. Implement BFS.
2. Implement DFS.

LabSheet3:

1.Write C programs to implement All-Pairs Shortest Paths problem using Floyd's algorithm.
2. Implement 8-Queens problem and analyze its time complexity.

Unit 4:

LabSheet1:

1. Implement N-Queens problem using Backtracking.
2. Write a program for Vertex Cover Problem..

LabSheet2:

Design and implement in C to find a subset of a given set S = {Sl, S2,.....,Sn} of n positive integers whose SUM is equal to a given positive integer d. For example, if S ={1, 2, 5, 6, 8} and d= 9, there are two

solutions {1,2,6}and {1,8}. Display a suitable message, if the given problem instance doesn't have a solution.

LabSheet3:

1. Compute the transitive closure of a given directed graph using DFS

Textbooks

1. Horowitz, Sahni, Rajasekaran "Fundamentals of Computer Algorithms",Galgotia Publications

Reference Books

1. Coremen, Leiserson, Rivest,Stein, "Introduction to Algorithms", 2nd edition, PHI.
2. Michael T. Goodrich, Roberto Tamassia "Algorithm Design and Applications" , Wiley
3. Aho, Hopcroft and Ullman, "The Design and Analysis of Computer Algorithms", Pearson

COURSE OUTCOMES (CO):

CO1: Demonstrate the ability to analyze and evaluate the efficiency of algorithms using asymptotic notations and various complexity analysis techniques.
CO2: Solve problems using advanced algorithmic techniques, including recurrences, the Master Method, and randomized algorithms.
CO3: Implement divide and conquer, greedy, and dynamic programming strategies to efficiently solve computational problems like sorting, knapsack, and shortest paths.
CO4: Apply backtracking and branch and bound techniques to tackle complex problems such as the 8-Queen problem and least-cost optimization.
CO5: Gain a deep understanding of computational complexity concepts, including P, NP, NP-hard, and NP-complete problems, and the significance of Cook's Theorem.
CO6: Evaluate the necessity of approximation algorithms and effectively apply them to solve optimization problems like the vertex cover, traveling salesman, and subset sum problems.

COURSE TITLE:JAVA PROGRAMMING

Course Code: MCA21302CR						**Examination Scheme**		
Total number of Lecture Hours: 48						**External**	**80**	
						Internal	**20**	
Lecture (L):	**3**	**Practicals(P):**	**2**	**Tutorial (T):**	**-**	**Total Credits**		**4**

Course Objectives

- Understand the fundamental principles of Java programming language, including its syntax, semantics, and basic constructs.
- Explore object-oriented programming concepts such as classes, inheritance, polymorphism, and interfaces in the context of Java.
- Develop proficiency in handling exceptions and errors using Java's exception handling mechanisms.
- Gain practical experience in utilizing Java's standard library classes and packages for tasks like I/O operations, string manipulation, and multithreading.
- Learn to create graphical user interfaces (GUIs) in Java, employing event-driven programming paradigms and integrating various GUI elements.
- Acquire skills in network programming with Java, including socket programming for communication between distributed systems and applications.

Course Content	**TEACHING HOURS**
UNIT 1: Introduction to Java Programming	**12- Hrs**
Introduction to Java Language: Creation of Java. How Java changed the Internet. Features of Java Language. Evolution of Java. Comparison with other languages like C++.Java Virtual Machine (JVM) and Byte-code. Java Language Overview: Lexical issues – Whitespace, Identifiers, Keywords, Literals, Separators, and Comments. Installing JDK.PATH variable. Java program – Structure, Compilation and Execution. Java Class libraries (System Class).main() method. **Data types, Variables and Arrays:** Primitive Data-types and Typed-Literals. Variables – Declaration, Initialization, Scope and Lifetime. Arrays – Single and Multidimensional. Type Conversion and Expression Promotion. **Operators, Expressions and Control statements:** Arithmetic, Bitwise, Relational, Logical, Assignment. Precedence and Associativity. Selection, Iteration and Jump Statements.	
UNIT 2: Object-Oriented Programming in Java	**12- Hrs**
Class Fundamentals: Class Structure (Variable and Method declaration).Modifiers (Access Modifiers and Other Modifiers).Components of Class, Variable and Method declaration. Constructor and finalize(). Garbage Collection. Passing parameters to methods. Variable hiding. Method overloading. Constructor overloading and chaining. Use of this keyword. Code blocks - Static and non-static. **Inheritance:** Mechanism. Role of Access Modifiers. Method Overriding and Shadowing. Use of super keyword. Polymorphism - Early and Late binding. Abstract Class and Interface. Components of Interface declaration. Implementing Interfaces.	

Exception Handling: Mechanism - Exception-Object, Throwing an Exception, and Exception Handler. Catch or Specify policy. Types of Exception - Checked vs Unchecked, Built-in vs Userdefined. Catching an Exception - try-catch-finally. Specifying an Exception - throws. Manually throwing an Exception - throw. Custom Exceptions. Chained Exceptions.	
UNIT 3: Advanced Java Concepts	**12- Hrs**
Packages: Creating and Importing Packages. CLASSPATH variable. static import. **Strings:** Mutable and Immutable Strings. Creating Strings. Operations on Strings. **Threads:** Creating Threads in Java. Java Thread Lifecycle. Multithreading in Java: Synchronization and Inter- process communication (IPC) in Threads. **Applet:** Java Applet class Architecture. Working and Lifecycle of Java Applet. Displaying text and animation, and passing parameters to Applet. Embedding Applets in a web page.	
UNIT 4: Java GUI Programming and Networking	**12-Hrs**
Event-Driven Programming: Java 1.1 Event Delegation Model – Source object, Event object and Listener object. Methods associated with Source, Event and Listener objects. Low-level vs Semantic events. Adapter classes, Inner classes, and Anonymous Inner classes. Adding GUI elements to Applet. **I/O Streams:** Byte, Character, Buffered, Data, and Object Streams. Standard Streams. File I/O Basics, Reading and Writing to Files. Serializing Objects. **Networking Classes and Interfaces:** TCP/IP Server Sockets in Java. Developing simple networking applications in Java like File transfer, Chatting, etc.	

LAB MANUAL

UNIT I

Lab Sheet 1:

- Download latest version of Java Development Kit (JDK), preferably JDK8 or above (Please visit https://java.com/en/download/).
- Follow the instructions that appear during the Installation ofJDK8, and set PATH variable to the appropriate directory location as instructed in the lecture.

- Lab Sheet **2:**
  - Write a Java program that displays "hello world!" on the screen.
  - Write a Java program that receives two integer numbers via keyboard, does their summation, and displays the result. Ensure that only integer values are processed.
  - Write a Java program that prints the season name corresponding to its month number using Ifelse and switch-case statements.
  - Write a Java program that sorts (using bubble sort) an integer array using for loop.
  - Write a Java program that calculates factorial of a number (inputted via keyboard) recursively.
  - Write a Java program that creates a 2D integer array with 5 rows and varying number of columns in each row. Using 'for each' variant of for loop display each element of every row.
- Lab Sheet **3:**
  - Write a Java program that creates a Class, namely Student.
    - Ensure that Age instance variable of the Class is never accessed directly, and its value is never less than 4 and greater than 40 for any Object of the Class (use methods to validate and assign the value).

- Ensure that the constructor always assigns a unique value to Enrollment_No instance variable for every Object of the Class (use a static class variable for counting objects, say Object_Counter).
- Ensure that when an Object is removed, the Object_Counter is automatically decremented (use finalize()), and whenever required the variable can only be accessed using a method even without an Object reference (make the counter private and use a static method to access it).

- Write a Java program in which a Class overloads a method sum(), which takes 2 parameters. The overloaded methods should perform summation of either integer or floating-point values

UNIT II

Lab Sheet 1:

- Write a Java program that creates a Class namely A that has a private instance variable and method, a protected instance variable and method, a default instance variable and method, and a public instance variable and method. Create another Class say B that inherits from A.
  - Show that all except private members are inherited.
  - Show that an inherited instance variable can be shadowed (with the same or weaker access visibility) but can be accessed using super keyword in the sub-class.
  - Show that an inherited method can be overridden (with the same or weaker access visibility) but can be accessed using super keyword in the sub-class.
  - Show that the reference variable of type A or B can't access an overridden method of A in the Object of B.
  - Show that the reference variable of type A can access a shadowed data member of A in the Object of B.

Lab Sheet 2:

- Write a Java program that creates a Class in which a method asks the user to input 2 integer values, and calls another member function (say div()) to divide the first inputted number by the second number (by passing them as parameters). Handle an exception that can be raised in div() when the denominator equals zero (use try-catch statement).
- Modify the above Java program so that it also creates a Custom Exception that is thrown by div() when the denominator value is 1 (use throw). Handle the exception. c.
- Modify the above Java program so that the exception-handling in not performed by div() rather it only species all the possible exceptions it may throw (use throws). And, the method that calls div() does the exception handling.

Lab Sheet 3:

- Create a Java Package (say pack1) that contains 3 Classes (say A, B and C). Write a Java program that uses this package after setting the CLASSPATH variable. Following scenarios must be considered individually:
  - Importing the whole package (all the 3 classes)
  - Importing only specific class (say Class A only)
- Create another Package (say pack2) that contains same number of classes, and same definition for each class, as that of pack1. Write a Java program that imports all classes from both pack1 and pack2 while ensuring that the name conflicts are not encountered while accessing any of these classes.

UNIT III

Lab Sheet 1:

- Write a Java program to count the number of words in a string that is passed as a command line argument.
- Write a Java program to check whether a string is palindrome or not.
- Write a Java program to count the total number of occurrences of a given character in a string.
- Write a Java program to convert a string to char array.

Lab Sheet 2:

- Write a Java program that creates a Class that extends a Thread class. Create 3 objects of the class, each starting a new thread and each thread displaying "I am Thread: " in an infinite loop. The displayed text must be suffixed by the unique name of the thread.
- Write a Java program that creates a Class that implements interface Runnable, and does the same as the above program.
- Write a Java program to implement a solution for producer-consumer problem using synchronization and inter-process communication in Threads.

Lab Sheet 3:

- Write a Java program that creates a Class that extends an Applet class. The applet is embedded in a web page and is passed 2 numeric parameters. The applet shall display the summation result of the parameters passed.
- Write a Java program that creates a Class that extends an Applet class. The applet simulates a marquee by displaying characters of the message one at a time from right to left across the screen. When the message is fully displayed, the message starts again.
- Write a Java program that creates a Class that extends an Applet class. The applet displays bar chart forthe data passed as parameter. The data includes the number of male and female students enrolled in MCA course.

UNIT IV

Lab Sheet 1:

- Write a Java program that creates a Class that extends an Applet class. Add GUI elements to the applet so as to create a simple 2-player tic-tac-toe game.
- Write a Java program that creates a Class that extends an Applet class. Add GUI elements to the applet so as to create a simple calculator.
- Write a Java program to open and read a file (filename is passed as command line argument), and displays the number of words in the file?
- Write a Java program to copy a file. The source and destination filenames arepassed as command line arguments.

Lab Sheet 2:

- Write a Java program (client) that sends a text message to another Java program (server), which receives and displays it.
- Modify the above Java programs so that each of the two programs is able to send and receive the text messages.

Lab Sheet 3:

- Write a Java program (a client) that opens a connection to https://www.Internic.net

website and displays information about www.google.com.

- Write a Java program (Client) that sends a text message to another Java program (Server), and the Server displays an acknowledgement message on receiving it.
- Write a Java program (Client) that sends a text string to another Java program (Server), which receives it and sends back the reverse string of the received string.

Textbooks

H. Schildt, Java: The Complete Reference, 9th Edition, Tata McGraw Hill, 2014.

Reference Books

- K. Sierra, Sun Certified Programmer For Java 5, Wiley India, 2006
- K. Sierra and B. Bates, Head First Java (Java 5), 2nd Edition, O'Reilly, 2003.
- H.M. Dietel and P.J. Dietel, Java: How to Program, 6th Edition, Pearson Education, 2007.
- C.S. Horstmann and G. Cornell, Java 2 Vol-1 Fundamentals, 7th Indian Reprint, Pearson Education, 2006.
- E. Balagurusamy, Programming with Java: A Primer, 4th Edition, Tata Mcgraw Hill, 2010.

COURSE OUTCOMES (CO):

CO1: Master Java Fundamentals: Students would be able to demonstrate proficiency in Java syntax, data types, variables, operators, and control structures. They would be able to implement object-oriented programming concepts including classes, objects, methods, and constructors.

CO2: Handle Exceptions and Utilize Java Libraries: Students would be able to implement effective exception handling strategies to manage errors and ensure program reliability. Utilize Java's standard library for tasks such as I/O operations, string manipulation, and collections.

CO3: Develop GUI Applications: Students can design and develop graphical user interfaces (GUIs) using Java Swing or JavaFX frameworks. Implement event-driven programming to create responsive user interfaces and handle user interactions.

CO4: Implement Networked Applications: Students can develop client-server applications using Java's networking APIs. Utilize TCP/IP protocols for tasks such as file transfer, real-time communication, or remote procedure calls (RPC).

COURSE TITLE: Operating System

Course Code: MCA21303CR						**Examination Scheme**	
Total number of Lecture Hours: 48						**External**	**80**
						Internal	**20**
Lecture (L):	**4**	**Practicals(P):**	**-**	**Tutorial (T):**	**-**	**Total Credits**	**4**

Course Objectives

1. Understand Fundamental Concepts of Operating Systems
2. Develop Skills in Process Management and Synchronization
3. Explore Distributed Operating Systems
4. Gain Expertise in Deadlocks Management
5. Explore Real Time Operating System
6. Acquire Skills in Real-Time Task Scheduling

Course Content	**TEACHING HOURS**
UNIT 1:	**12 Hrs**
Types of Operating Systems; Operating System Structures – Processes, Scheduling criteria, Scheduling Algorithms. Processor allocation and scheduling in distributed systems - System Models, Load balancing and sharing approach, fault tolerance; Real time distributed systems.	
UNIT 2:	**12 Hrs**
Interprocess Communication and Synchronization, Classical problems, Critical section, Semaphores, Monitors. Synchronization in Distributed Systems - Clock Synchronization and related algorithms, Logical Clocks. Mutual Exclusion: Centralized & Distributed (Contention & Token) Algorithms. Election Algorithms: Bully Algorithm, Invitation Algorithm.	
UNIT 3:	**12 Hrs**
Memory Management: Address Spaces, Virtual Memory. Page Replacement Algorithms, Design and Implementation Issues for Paging Systems, Segmentation. General architecture of Distributed Shared Memory systems; Design and implementation issues of DSM; granularity - Structure of shared memory space, consistency models, replacement strategy, thrashing.	
UNIT 4: Deadlocks	**12 Hrs**
Deadlocks characterization, Methods for handling deadlocks. Deadlock - Prevention, Avoidance, Detection, Recovery. Deadlock Detection - Distributed Algorithms Threads - Characteristics, Advantages & Disadvantages, Design Issues & Usage. Client Server model; Remote procedure call and implementation issues.	

LAB MANUAL

UNIT I

Lab Sheet 1

1. Write a program to implement process systemcalls.
2. Write a program to implement I/O systemcalls

Lab Sheet 2:

1. Write a program to simulate the SJF scheduling algorithm. The program should read the following inputs: • Number of processes • Burst time requirement of each process The program should generate the following outputs: • Process statistics after each context switch • Average Turn around time • Average Waiting time
2. Write a program to simulate the Round Robin scheduling algorithm. The program should read the following inputs: • Number of processes • Burst time requirement of each process • Length of the Time Slice The program should generate the following outputs: • Process statistics after each context switch Average Turn around time • Average Waiting time

Lab Sheet 3:

1. Write a program to simulate FCFS scheduling algorithm.
2. Write a program to simulate priority schedulingalgorithm.

Unit II

Lab Sheet 1:

1. Write a program to implement the producer – consumer problem using semaphores.
2. Write a program to implement IPC using shared memory.
3. Write a program to simulate the concept of dining philosophers problem.

Lab Sheet 2:

1. Create client server programs using RPC wherein the server accepts a number from the client and returns the square of the number which is then displayed by the client. Use rpcgen to generate the stubs automatically.
2. Write a program to simulate Clock Synchronization in Distributed Systems using Lamport's Algorithm.

Lab Sheet 3:

1. Write a program to simulate the Bully Election algorithm.

UNIT III

Lab Sheet 1

1. Write a program to implement and simulate MFT (Memory management with fixed partitioning technique) algorithm.
2. Write a program to implement and simulate MFT (Memory management with variable partitioning technique) algorithm
3. Write a program to simulate the following contiguous memory allocation techniques a) Worst-fit b) Best-fit c)First-fit

Lab Sheet 2:

1. Write a program to simulate the LRU page replacement algorithm. The program should read the following inputs: • Length of the reference string • Reference string • Number of page frames The program should generate the following outputs: • Page replacement sequence after each reference • Number of page faults 2. Write a program to simulate the LFU page replacement algorithm. The program should read the following inputs: • Length of the reference string • Reference string • Number of page frames The program should generate the following outputs: • Page replacement sequence after each reference • Number of page faults 3. Write a program to simulate the FIFO page replacement algorithm.

Lab Sheet 3:

1. Write a set of programs to use the concept of shared memory through LINUX system calls. • One process creates a shared memory segment and writes a message into it. • Another process opens the segment, reads the message and outputs the message to standard output. Some of the important system calls to be used include: shmget(), shmat(), shmctl() etc.

Unit IV

Lab Sheet 1:

1. Write a program to simulate the Banker's Algorithm for Deadlock Avoidance. The program should

read the following inputs: • Number of Processes • Number of resource types • Current allocation and Maximum allocation of resources to each process • Currently Available Resources • New request details The program should generate the following outputs: • Determine whether the system is in the safe state or not

2. Modify the previous program to determine the safe sequence if the system is in safe state.

Lab Sheet 2:

1. Write a program to implement deadlock detection (resource allocation graph)algorithm.
2. Write a program to simulate deadlock prevention.

Lab Sheet 3:

1. Write a program to implement mutual exclusion of threads on LINUX using the pthread.h library Some of the important system calls to be used include: pthread_mutex_lock, pthread_self, pthread_create, pthread_exit

Textbooks:

Abraham Silberchatz, Peter B. Galvin, Greg Gagne, "Operating System Principles", John Wiley.
Pradeep K. Sinha , "Distributed Operating Systems : Concepts and Design", PHI
Rajib Mall, Real-Time Systems: Theory and Practice (Second Edition), Pearson Education.

Reference Books:

Andrew.S. Tanenbaum, "Modern Operating Systems", PHI. Andrew. S. Tanenbaum,"Distributed Operating System", PHI.
Andrew S. Tanenbaum, Modern Operating Systems (Third Edition), Pearson Education.
David E. Simon, An Embedded Software Primer, Pearson Education.
Laplante, P., Real-Time Systems Design and Analysis (Third Edition), IEEE/Wiley Interscience.
Jane W.S. Liu, Real-Time Systems (Sixth Edition), Pearson Education.
Raj Kamal, Embedded Systems: Architecture, Programming and Design (Third Edition), Tata McGraw-Hill Education

COURSE OUTCOMES (CO):

- **CO1: Students will understand the fundamental concepts and functions of an operating system.**
- **CO2: Students will develop skills in process management and CPU scheduling techniques.**
- **CO3: Students will acquire comprehensive knowledge of memory management methods and their practical applications.**
- **CO4: Students will achieve proficiency in the principles and design of distributed systems.**
- **CO5: Students will gain expertise in identifying, preventing, and resolving deadlocks.**

CO6: Students will acquire expertise in real-time systems.

Machine Learning

Course Code:				**MCA21304CR**		**Examination Scheme**	
Total number of Lecture Hours:24 Hrs						**External**	**40**
						Internal	**10**
Lecture (L):2	**-**	**Practicals(P):**	**-**	**Tutorial (T):**	**-**	**Total Credits**	**2**

Course Objectives

- *To present an overview of Machine Learning(ML) principles and approaches .*
- *To understand pattern clustering and classification algorithms to classify data*
- *To understand the Implementation of Support Vector Machine algorithm*
- *To create new machine learning techniques.*

Course Content	TEACHING HOURS
UNIT 1:	**12-Hrs**
Linear regression, Classification Algorithms: KNN and effect of various distance measures (Euclidean, Manhattan, Mahalanobis Distances, etc.) Clustering Algorithms: Fuzzy C-means, Hierarchical clustering, Density-based spatial clustering of applications with noise (DBSCAN) Cluster Validity index. Compactness Cluster Measure, Distinctness Cluster Measure, Validity Index Using Standard Deviation, Point Density Based Validity Index, Validity index using Local and Global Data Spread,	
UNIT 2:	**12Hrs**
Logistic Regression, Support Vector Machines: Binary Linear Support Vector Machines, Optimal Hyperplane, Kernel Functions, Solving Non-linear Classification problems with Linear Classifier. Applications of Support Vector Machines. Dimensionality Reduction, Principal Component Analysis, Fisher Linear Discriminant, Quadratic Discrminant Analysis, Multiple Discriminant Analysis..	

Textbooks

1. Tom M. Mitchell, "Machine Learning", McGraw-Hill, 2010
2. Bishop, Christopher. Neural Networks for Pattern Recognition. New York, NY: Oxford University Press, 1995

Reference Books

1. Introduction to Machine Learning by Ethem Alpaydin, MIT Press
2. Pattern Classification by Duda and Hart. John Wiley publication
3. The Elements of Statistical Learning by Trevor Hastie, Robert Tibshirani, Jerome Friedman, Springer.
4. Pattern Recognition and Machine Learning, Christopher M. Bishop, Springer
5. Machine Learning: A probabilistic Perspective, by Kevin P. Murphy, MIT Press

COURSE OUTCOMES (CO):

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

CO1: Develop and apply pattern classification algorithms to classify multivariate data.
CO2: Develop and apply regression algorithms for finding relationships between data variables.
CO3: Apply a variety of learning algorithms to data.
CO4: Perform evaluation of learning algorithms and model selection?
CO5: Write scientific reports on computational machine learning methods, results and conclusions.

COURSE TITLE: Theory of Computation

Course Code: MCA21305DCE						**Examination Scheme**	
Total number of Lecture Hours: 48						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practicals(P):**	**-**	**Tutorial (T):**	**1**	**Total Credits**	**4**

Course Objectives:

- ✓ To Understand computational models and finite automata in formal language theory and computational complexity.
- ✓ To Design and analyze DFA and NFA, understand regular languages, and their equivalence with regular expressions.
- ✓ To Study context-free languages (CFLs), grammars (CFGs), parse trees, and pushdown automata (PDA).
- ✓ To Explore context-sensitive languages (CSL), linear bounded automata (LBA), recursive languages (REL), and Turing machines (TM).
- ✓ To Learn about decidability, undecidability, reduction techniques, and complexity theory foundations.

COURSE CONTENT	TEACHING HOURS
UNIT 1: Introduction to Computation	**12 Hrs**
Introduction to computation, Finite Automata, DFA, Kleene's theorem, Non-determinism, Finite Automata with output. Regular Languages: introduction to formal languages, regular operations, closure property Regular Expression; Equivalence of DFA, NFA, and RE. Non-Regular Languages and Pumping Lemma.	
UNIT 2: Context-Free Languages	**12 Hrs**
Context-Free Languages: introduction to CFL, context free grammars, Chomsky normal form, parse trees, derivation and ambiguity, closure and non-closure properties. Pushdown Automata (PDA), Deterministic vs Non-deterministic PDAs. Non-CFL and pumping Lemma for CFLs.	
UNIT 3: Context-Sensitive Languages and Turing Machine	**12 Hrs**
Context-Sensitive Languages: introduction to CSL, context sensitive grammars, Linear Bounded Automata (LBA) Recursive and Recursively Enumerable Languages: introduction to REL and Chomsky hierarchy, Hilbert's algorithm and Church-Turing Thesis. Turing Machines, equivalence of Deterministic, Non-deterministic, and multi-tape TMs. Universal TMs.	
UNIT 4: Undecidability and Computational Complexity	**12 Hrs**

Decidable Languages: Decidability, and Undecidability, Reductions and its applications. A Halting Problem, Complexity: Asymptotic Notation and properties thereof. Deterministic and Nondeterministic Turing Machine cost models (space and time).	

Tutorials

Unit I

Week 1: Kleene Closure

a. Let L= {ab, aa, baa}.Which of the following strings are in L*= { abaabaaabaa, aaaaabaaaa, baaaaaabaaaab, baaaaabaa}.

b. Let L= {ab, cd}. Write down first ten strings in L* using Lexicographic ordering

c. Show that (L*)* = L* for all languages?

Week 2. Finite Automata with output

a.What is the difference between Moore & Mealy machines?

b.Construct a Mealy machine to accept all strings ending with aa or bb over the Alphabet {a,b}.

c.How are transducers different from other automata?

Week 3. Finite Automaton and Regular Expression

a. Give a deterministic finite automaton over the alphabet {a, b} which accepts all strings containing no more than two consecutive occurrences of the same input letter. (For example, abba should be accepted but not abaaab.)

b. Give regular expressions for the following languages on

1. all strings containing exactly one a

2. all strings containing no more than three a's

3. all strings that contain no run of a's of length greater than two

4. all strings in which all runs of a's have lengths that are multiples of three.

Unit II

Week 4.Pushdown Automata

a. Demonstrate the Construction of a Pushdown Automata with example

b. Construct pushdown automata for the following languages. Acceptance either by empty stack or by final state.

Week 5. Context Free Grammar.

a. Find a Context-Free Grammar for the following language

b. Find a CFG that generates the language

c. Which language generates the grammar G given by the productions

1. S → aSa | aBa B → bB | b

2. S → abScB | λ B → bB | b What language does it generate?

Week 6. Pumping Lemma

a. Demonstrate the use of Pumping Lemma for context free languages with the help of an example

b. Using Pumping Lemma, Prove {0n 1 n 2 n | n ≥ 0} is not a context free language.

Unit III

Week 7. Turing Machines

a. Discuss Turing machine with the help of an example

b. Give a detailed description of a total Turing machine accepting the palindromes over {a, b}: that is, all strings x ∈ {a, b} ∗ such that x = rev x.

Week 8.Multi Tape Turing Machine

a.Demonstrate use of Multitape TMs with the help of an example

b.Suppose we try to construct a Turing machine to solve a particular problem, but we are not successful. Does it mean that no Turing machine exists that can solve that problem? Explain and justify your answer.

Week 9.Universal Turing Machine.

a. Demonstrate encoding of a Universal ™ with help of an example

b. Draw the state diagram for a Turing machine that increments a binary number. Assume that the input tape contains at least one non-blank symbol.

Unit IV

Week 10. Decidability

a. Discuss Decidability of a langua ge with help of an example

b. Let L be a decidable language. Prove that the complement L' is decidable

c. Prove that L ∪ L' is decidable, when L is decidable. d.

Week 11. Complexity Theory

a. Discuss the concept of Complexity theory in terms of DTIME, DSPACE,NTIME, NSPACE

b. Compute computation complexity of language L= {0^n1^n| n>0}

Week 12. The Halting Problem

a. Explain the concept of Reduction.

b. Is there an explicit program P so that for a given y it is decidable whether P terminates on input y?

Textbooks

1. New York: Wiley. Linz, Peter. An introduction to formal languages and automata. Jones & Bartlett Learning. Seventh Edition.
2. "Introduction to the Theory of Computation" by Michael Sipser, Third Edition.

Reference Books

1. Cohen, Daniel IA, Introduction to computer theory, 2nd Edition.
2. Parkes, Alan P. Introduction to languages, machines and logic: computable languages, abstract machines and formal logic. Springer Science & Business Media, 2012., 2nd Edition

COURSE OUTCOMES (CO):

Upon successful completion of this course, learners will be able to:

1. **Interpret the role of computational models and finite automata in computer science**, recognizing their significance in formal language theory and computational complexity.
2. **Design and analyze deterministic and non-deterministic finite automata (DFA and NFA)**, demonstrating an understanding of regular languages and their equivalence with regular expressions.
3. **Understand and work with context-free languages (CFLs), context-free grammars (CFGs), parse trees, and pushdown automata (PDA)**, including identifying their properties and limitations.
4. **Explore and analyze advanced automata and language classes**, including context-sensitive languages (CSL), linear bounded automata (LBA), recursive and recursively enumerable languages (REL), and Turing machines (TM), and understand their roles in computational theory.
5. **Analyze decidability and complexity concepts**, including applying reduction techniques and understanding the basics of complexity theory, such as asymptotic notation and the models of deterministic and non-deterministic Turing machines.

COURSE TITLE: Wireless and Mobile Communication

Course Code			**: MCA21306DCE**			**Examination Scheme**	
Total number of Lecture Hours: 48						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practical(P):**	**0**	**Tutorial (T):**	**1**	**Total Credits**	**4**

Course Objectives:

- Learn the classification and types of wireless telephones including Cordless, Fixed Wireless (WLL), Wireless with limited mobility (WLL-M), and Fully Mobile Wireless phones.
- Understand the concept of cells, sectorization, coverage area, frequency reuse, and cellular networks.
- Learn the purpose and functionality of various channels such as Pilot, Sync, Paging, Forward Traffic Channels, Access Channels, and Reverse Traffic Channels.
- Understand the GSM reference architecture and the components of mobile networks including Mobile Stations (MS), Base Transceiver Stations (BTS), Base Station Controllers (BSC), and Mobile Switching Centers (MSC).
- Explore different handoff scenarios within the GSM network.

Course Content	**TEACHING HOURS**
UNIT 1:	**12 Hrs**
Classification and types of Wireless telephones. Introduction to Cordless, Fixed Wireless (WLL), Wireless with limited mobility (WLL-M) and (Fully)Mobile Wireless phones. Introduction to various generations of mobile phone technologies and future trends. Wireline vs. Wireless portion of mobile communication networks. Mobile-Originated vs. Mobile-Terminated calls. Mobile Phone numbers vs. Fixed-Phone numbers	
UNIT 2:	**12 Hrs**
Concept of cells, sectorization, coverage area, frequency reuse, cellular networks & handoffs. Wireless Transmission concepts; types of antennas; concepts of signal propagation, blocking, reflection, scattering & multipath propagation. Comparison of multiple access techniques FDM, TDM and CDM. Concept of Spread Spectrum(SS) techniques; Frequency Hopping SS . Direct Sequence SS and concept of chip-sequence.	
UNIT 3:	**12 Hrs**
Concept of Forward and Reverse CDMA channel for a cell/sector. Concept/derivation of Walsh codes & Code Channels within a CDMA Channel. Simplified illustration of IS-95 CDMA using chip sequences. Purpose of Pilot, Sync, Paging, Forward Traffic Channels. Purpose of Access & Reverse TCs.	
UNIT 4:	**12 Hrs**
GSM reference architecture and components of Mobile Networks: MS, BTS, BSC, MSC; their basic functions and characteristics. Use of HLR and VLR in mobile networks. Handoff scenarios in GSM.	

Tutorial

Unit I

Tutorial 1
Q1. Describe the evolution of wireless and mobile communication technologies by writing concise notes on: (a) Fixed Wireless (b) Cordless Phones (c) WLL / WLL-M technologies (d) Fully-Mobile Wireless
Q2. Name and briefly describe three technologies used by second-generation mobile networks and indicate the bandwidth of the channel used by each one.
Q3. Explain the concept of a cell, coverage area and sectorization.
Tutorial 2
Q1 Draw a diagram showing the positioning of wireless networks vis – a - vis wired network.
Q2 Why are wired +network usually part of the wireless infrastructure?
Q3 Differentiate between Portability, nomadicity and mobility
Tutorial 3
Q1 Name three channel sounding techniques, Give the advantages and disadvantages of each.
Q2 What are the three important radio propagation phenomena at high frequencies? Which of them is predominant indoors

Unit II

Tutorial 1
Q1. Using diagrams, explain the idea of Frequency Reuse in the context of AMPS and CDMA.
Q2. Using a diagram and text explain the concept of handoff/handover in mobile networks.
Q3. Write short notes on: (a) types of antennas; (b)concepts of signal propagation, blocking, reflection, scattering & multipath propagation.

Tutorial 2
Q1 Name the two most popular techniques used in digital cellular modems and give one example standard that uses each of them
Q2 For a 64-QAM modem give the SNR at which the error rate over a telephone line is 10.
Q3 Why is PPM used with infrared communication instead of PAM?

Tutorial 3
Q1 Name a cellular telephony standard that employs FDMA
Q2 What are the popular access schemes for data networks? Classify them.
Q3 Name two duplexing methods and one example standard that uses each of these technologies.

Unit III

Tutorial 1

Q1. Using diagrams and text explain the Concepts of Spread Spectrum(SS) techniques; Frequency Hopping SS & Direct Sequence SS.
Q2. Explain using diagrams the Concept of Forward and Reverse CDMA channel for a cell/sector.
Q3. Explain the Concept/Derivation of Walsh codes & Code Channels within a CDMA Channel.

Tutorial 2
Q1 What is the difficulty of implementing CSMA/CD in a wireless environment
Q2 What is the capture effect and how does it impact the performance of the random access methods?
Q3 Name three standard using TDMA/TDD as their access method.

Tutorial 3
Q1 Assume that you have a six secyor cells in a hexagonal geometry. Draw the hexagonal grid corresponding to this case, Compute S, for reuse factors of 7,4 and 3. Comment on your results
Q2 Compare peer to peer and multihop ad hoc topologies

Unit IV

Tutorial 1

Q1. Explain the Purpose of Pilot, Sync, Paging, Forward Traffic Channels in CDMA networks.
Q2. Using diagrams and text explain briefly GSM reference architecture and components of Mobile Networks: MS, BSC, NSS; their subsystem functions and characteristics.
Q3. Draw diagrams with associated text to explain various Handoff Scenarios supported in GSM.

Tutorial 2
Q1 Give three reasone why it is difficult to dwtect collusions at the transmitter in wireless networks.
Q2 What are the new elements added to the GSM infrastructure to support GPRS?
Q3 What are the new elements added to the AMPS infrastructure to support CDPD?

Tutorial 3
Q1 Draw the protocol stack of CDPD to the M-ES at the MDMS and at thr ND-IS. Show the communication between different peer layers.
Q2 Of the design goals of CDPD which three do you consider important? Why?
Q3 Explain with diagram MTP, PTP ?

Textbooks

3. K.Pahlavan, P.Krishnamurthy, “Principles of Wireless Networks”, PHI.

Reference Books

8. T. Rappaport, “Wireless Communications, Principles and Practice(2nd Edition)”,Pearson.Andy
9. 2. Dornan, “The Essential Guide to Wireless Communications Applications”,Pearson.
10. Jochen Schiller, “Mobile Communications”, Pearson.

COURSE OUTCOMES (CO):

CO1: Students will be able to classify and describe different types of wireless telephones and their uses.
CO2: Students will gain insights into the evolution of mobile technologies and the differences between various network types and call scenarios.
CO3: Students will understand and explain key concepts in cellular networks including cell structure, sectorization, and handoffs.
CO4: Students will be able to describe and derive the functionality of Forward and Reverse CDMA channels.
CO5 : Students will understand and apply Walsh codes, chip sequences, and the different CDMA channel types within a network.

COURSE TITLE: Organizational Behaviour

Course Code: MCA21307DCE						**Examination Scheme**	
Total number of Lecture Hours: 48						**External**	**80**
						Internal	**20**
Lecture (L):	**3**	**Practicals(P):**	**-**	**Tutorial (T):**	**1**	**Total Credits**	**4**

Course Objectives

- The main objective of Organizational Behavior is to understand the human interactions in an organization find what is driving it and influence it for getting better results for attaining business goals.
- The organizations in which people work have an effect on their thoughts, feelings, and actions. These thoughts, feelings, and actions, in turn, affect the organization itself.
- Organizational behavior studies the mechanisms governing these interactions, seeking to identify and foster behaviors conducive to the survival and effectiveness of the organization.

Course Content	TEACHING HOURS
UNIT 1:	**12 Hrs**
Definition, need and importance of organizational behaviour, Nature and scope, Frame work, Organizational behaviour models. Personality – types – Factors influencing personality – Theories – Learning – Types of learners – The learning process – Learning theories – Organizational behaviour modification.	
UNIT 2:	**12 Hrs**
Misbehaviour – Types – Management Intervention. Emotions - Emotional Labour – Emotional Intelligence – Theories. Attitudes – Characteristics – Components – Formation – Measurement- Values. Perceptions – Importance – Factors influencing perception – Interpersonal perception- Impression Management. Motivation – importance – Types – Effects on work behaviour	
UNIT 3:	**12 Hrs**
Organization structure – Formation – Groups in organizations Influence – Group dynamics – Emergence of informal leaders and working norms Group decision making techniques – Team building - Interpersonal relations Communication – Control. Meaning – Importance – Leadership styles – Theories – Leaders Vs Managers – Sources of power – Power centers – Power and Politics.	
UNIT 4:	**12 Hrs**

Oganizational culture and climate, Factors affecting organizational climate, Job satisfaction – Determinants – Measurements – Influence on behaviour. Organizational change – Importance – Stability Vs Change – Proactive Vs Reaction change – the change process – Resistance to change – Managing change. Stress, Work Stressors, Prevention and Management of stress, Balancing work and Life. Organizational development, Characteristics, objectives, Organizational effectiveness	

Tutorial

Unit 1

Tutorial 1

Q1 Define Organisational Behaviour. State its importance and scope.

Q2 Define planning. Explain the steps involved in planning and state the limitations in planning

Q3 Explain the importance of planning as the beginning of the process of management. State how decision making plays a vital role in the exercise of planning.

Tutorial 2

Q1 Distinguish clearly between intrapersonal and interpersonal conflicts. Quote an example. How does it deteriorate teamwork in the organisation?

Q2 State how systems Approach and contingency Approach have played the role of integrating various fragmented approaches of management

Q3 Explain the theory of transactional analysis. Discuss ego states as its link

Tutorial 3

Q1 Which leadership style is suitable to HR Manager of I.T. industry in the present era. Give justification

Q2 Discuss the merits and demerits of formal and informal group formation in industrial organisation functioning at the national level

Q3 Elaborate on the evolution of management thought & its relevance in today's scenario

UNIT 2

Tutorial 1

Q1 Define motivation. Elaborate A.H.Maslow' s hierarchy theory of motivation.

Q2 "Controlling techniques are very effective in an organisation". Elaborate

Q3 Write short notes on Formation of the team. b) Principles of decision making. c) Dimensions of attitude d) MBO. e) Stress management.

Tutorial 2

Q1 Elaborate on the SOBC model of O.B. Give Examples

Q2 Explain the concept of conflict management with its Process.

Q3 Compare A.H. Maslow's theory with Herzberg's theory of Motivation

Tutorial 3

Q1 Explain the meaning of personality. What are the determinants of personality? Give relevant examples.

Q2 Distinguish between formal organizations & informal organizations. Explain the importance of the formation of teams Q3 Write short notes on a) Functions of management. b) Morale Indicators. c) Dimensions of attitude. d) Planning premises.

e) Job satisfaction.

UNIT 3

Tutorial 1

Q1 "Nothing is constant, the only change is constant". Explain the statement w.r.t. factor responsible for the change.

Q2 What is departmentalization? Explain the various types of departmentalization?
Q3 Write short notes on 1) Decision-making process. 2) Leadership styles. 3) Models of OB. 4) Functions of Management. 5) Line and staff authority.

Tutorial 2
Q1 What are the different types of motives? Explain A.H.Maslow's hierarchy need a theory of motivation
Q2 "Its is remarked that attitudes shape the personality of an individual". Comment.
Q3 Explain nature 7 purposes of planning with its steps, in detail.

Tutorial 3
Q1 what do you understand by 'Motives' and explain the Herzberg theory of motivation, with Relevant examples.
Q2 Define stress. Explain ill effects of stress on human beings. How do people manage stress
Q3 Enumerate various factors responsible for the change

UNIT 4
Tutorial 1
Q1 What is conflict? What are the sources of conflict?
Q2 What can be the consequences of conflict on an organisation?
Q3 How can grievance affect an organisation and its employees? Describe the process of handling grievance

Tutorial 2
Q1 What are the Factors affecting organizational climate
Q2 How can an employee balance his work and personal life in an organization
Q3 What do you mean by Organisational Culture? State its elements. Also discuss how organisational culture can be created and sustained.

Tutorial 3
Q1 Explain in details the various types of culture?
Q2 How to create a positive organsational culture?
Q3 Write short notes on: Strong Vs. Weak Culture II. Soft Vs. Hard Culture III. Formal Vs Informal Culture IV. Concept of Workplace Spirituality

Textbooks

7. Stephen P. Robins, Organisational Behavior, PHI Learning / Pearson Education, 11th edition.
8. Fred Luthans, Organisational Behavior, McGraw Hill, 11th Edition

Reference Books

1. Schermerhorn, Hunt and Osborn, Organisational behaviour, John Wiley
2. Udai Pareek, Understanding Organisational Behaviour, 2nd Edition, Oxford Higher Education.
3. Mc Shane & Von Glinov, Organisational Behaviour, 4th Edition, Tata Mc Graw Hill.
4. Hellrigal, Slocum and Woodman, Organisational Behavior, Cengage Learning, 11th Edition.
5. Ivancevich, Konopaske & Maheson, Organisational Behaviour & Management, 7th edition, Tata McGraw Hill.

COURSE OUTCOMES (CO):

CO1: Students will be able to define organizational behavior and explain its importance within organizations.

CO2: Students will be able to describe different personality types and the factors influencing personality development.

CO3: Students will identify various types of misbehavior in organizations and propose management interventions to address these issues.

CO4: Students will analyze group dynamics, including the emergence of informal leaders and working norms.

CO5: Students will compare and contrast leadership styles, theories, and the distinctions between leaders and managers. They will also identify sources of power, power centers, and the impact of power and politics in organizations.

COURSE TITLE: Fundamentals of Programming with C

Course Code: MCA21301OE						**Examination Scheme**	
Total number of Lecture Hours:24						**External**	**40**
						Internal	**10**
Lecture (L):	**2**	**Practicals(P):**	**-**	**Tutorial (T):**	**1**	**Total Credits**	**2**

Course Objectives

- Understand the principles of problem-solving and algorithm development.
- Gain proficiency in the C programming language.
- Develop and implement solutions to computational problems.
- Learn to debug, test, and optimize C programs.

Course Content	TEACHING HOURS
UNIT 1:	**12-Hrs**
Introduction to C language- Background, C programs, Identifiers, Data Types, Variables, Constants, Input/Output Statements, Arthimetic Opeartors, and Expressions: Evaluating Expressions, Precedence and Associativity of operators, Type Conversions.	
UNIT 2:	**12Hrs**
Conditional Control Statements: Relational and Logical operators, If, If-Else, Switch Statement and Examples, Loop Control Statements: For, While, Do While, and Examples , Function: Function Basics, User Defined Functions, Arrays , One and Two Dimensional Arrays	

Textbooks

1. Programming in ANSI C" by E. Balagurusamy, 8th Edition (2021), McGraw Hill Education
2. Let Us C by Yashavant Kanetkar, 18th Edition (2023), BPB Publications
3. Data Structures Using C; by Reema Thareja, 3rd Edition (2019), Oxford University Press

Reference Books

5. C Programming and Data Structures; by P. S. Deshpande and O. G. Kakde, 1st Edition (2012), Dreamtech Press
6. C Programming; by K. R. Venugopal and S. R. Prasad, 2nd Edition (2015), McGraw Hill Education

CO1: Understand basic C programming concepts: Master data types, control structures, and functions in C.
CO2: Design and implement algorithms: Develop solutions to computational problems using C.
CO3: Improve problem-solving skills: Apply systematic approaches to solve complex problems.
CO4: Apply C programming to real-world tasks: Use C for practical applications like file handling and data management**.**

SEMESTER IV

Effective from year-2021

PROJECT WORK

Effective from year-2021

COURSE TITLE:Management Information Systems

Course Code: MCA21401OE						**Examination Scheme**	
Total number of Lecture Hours: 20						**External**	**40**
						Internal	**10**
Lecture (L):	**2**	**Practicals(P):**	**-**	**Tutorial (T):**	**-**	**Total Credits**	**2**

Course Objectives

- Information Systems (IS) enables new approaches to improve efficiency and efficacy of business models. This course will equip the students with understanding of role, advantages and components of an Information System.
- The objective of the course is to help students integrate their learning from functional areas, decision making process in an organization and role of Information Systems to have a vintage point in this competitive world.

Course Content	**TEACHING HOURS**
UNIT 1:	**10 Hrs**
MIS Basics, System View of Business, Process of MIS, Development of MIS within the organization, Management Process, Information Needs, System Approach in Planning Organizing and Controlling MIS. Planning, Implementation and Controlling of Management Information System.	
UNIT 2:	**10 Hrs**
Fundamentals of Data Processing, Computer Operation of Manual Information System, Components of Computer Systems, Flow Chart, Conversion of Manual to Computer Based Systems, Computer Systems Software, Application Software. Managerial Decision Making, characteristics and components of Decision Support System.	

Text Book: Software Project Management, Bob Hughes and Mike Cotterell, McGraw Hill

Reference Books: -

1. Software Project Management A Unified Framework, Walker Royce, Addison-Wesley
2. A practitioner's Guide to Software Engineering, Roger Pressman, Tata McGraw Hill 2014 8th edition.
3. Basics of Software Project Management, NIIT,
Prentice-Hall India, Latest Edition

COURSE OUTCOMES (CO):

CO1: Define the principles of project management for developing software.
CO2: Explain various project management scheduling techniques.
CO3: Apply different techniques of project monitoring, control and review.

CO4: Classify various project management tools and estimate the risks involved in project activities.
CO5: Assess issues related to project quality and staffing.
CO6: Discuss the effect of project management practices in an organization