POST GRADUATE DEPARTMENT OF COMPUTER SCIENCES

University of Kashmir, Srinagar-190006

NAAC Accredited Grade "A+"

NOTES

Minutes of BOS (Undergraduate) Meeting held on 11-08-2023 at 10:30 a.m in Computer Science Department

A Board of Studies meeting was held on 11-08-2023 at 10:30 a.m in the office chamber of Head of the Department regarding finalization of syllabus for four-year undergraduate programme (FYUP) in Computer Applications under NEP-2020.

The following were present:

1. Dr. Javaid Iqbal	in the Chair….
2. Dr. Manzoor Ahmad Chachoo	Member
3. Prof. Mohd Ashraf Shah	Consultant U.G CBCS/NEP
4. Dr. Sajad M. Khan	Member
5. Dr. Faheem Syeed Masoodi	Member
6. Dr. Sajid Yousuf Bhat	Member
7. Mr. Gazi Imtiyaz	Member
8. Mr. Zubair Sayeed Masoodi	Member
9. Mr. Audil Hussain	Member
10. Ms. Aasiya Qayoom	Member
11. Mr. Rashid Ashraf Malik	Member (Cluster University, Srinagar.)
12. Mr. Faisal Maqbool	Member
13. Mr. Tanveer Ahmad Lone	Member (Autonomous College)
14. Dr. Irshad Ahmad Mir	Member
15. Mr. Suhail Ajmal	Member
16. Mr. Sajad Ahmad Shah	Member
17. Mr. Shabir Ahmad Rather	Member

The committee recommends the scheme and structure (Annexure–I) for the entire FYUP. Moreover, the committee recommends the syllabi for the courses of 3rd and 4th semester (Annexure–II) of the four year undergraduate programme in Computer Applications under NEP-2020 effective from the year 2023.

The meeting ended with a vote of thanks to the Chair.

Dr. Javaid Iqbal
(Chairman)

Dr. Manzoor A. Chachoo
(Member)

Prof. Mohd Ashraf Shah
Consultant U.G CBCS/NEP

Dr. Sajad M. Khan
(Member)

Dr. Faheem Masoodi
(Member)

Dr. Sajid Yousuf
(Member)

Mr. Gazi Imtiyaz
(Member)

Mr. Zubair S. Masoodi
(Member)

Mr. Audil Hussain
(Member)

Ms. Aasiya Qayoom
(Member)

Mr. Rashid Ashraf Malik
(Member)

Mr. Faisal Maqbool
Member

Mr. Tanveer Ah Lone
(Member)

Dr. Irshad Ahmad Mir
Member

Mr. Suhail Ajmal
(Member)

Mr. Sajad Ahmad Shah
(Member)

Mr. Shabir Ahmad
Rather
(Member)

Scheme and Structure for FYUP of Computer Application at UG Level effective from 2023

Semester	Major/Minor (CT-1)	Major (CT-2) 4+2	Major (CT-3) 4+2	Minor (Applied Computing)
1	Computer Fundamentals 4+2			Internet basics & Multimedia 4+2
2	Programming Fundamentals with C 4+2			Web Designing 4+2
3	Data Communication & Computer Networks 4+2			OOPs with C++ 4+2
4	DBMS 3+1	Java Programming	Computing Mathematics	IoT Fundamentals 3+1
5	Operating System 3+1	Data Structures using C	Discrete Mathematics	Theory of Computation 3+1
6	Python Programming 3+1	Digital Electronics	Probability and Statistics	Artificial Intelligence 3+1
7	Cyber Security 3+1	Computer Organisation & Architecture	Design & Analysis of Algorithms	Machine Learning 3+1
8 (Honours)	Software Engineering 3+1	Mini Project with Dissertation	Cryptography & Network Security	Mobile Application Development 3+1
8 (Honours with Research)	Research Methodology 3+1	Research Project		Latex 3+1

Skill Enhancement Courses: 1st: 2nd 3rd, From Basket.

Multidisciplinary Course: 1st, 2nd, 3rd Semester: Fundamentals of Computer Applications

Bachelor with Computer Applications as Major

Third Semester

Course Code: CAP322J **Course Title: Data Communication & Computer Networks**
Credits: Theory (4) Practical (2) **Max. Marks: 100** **Min. Marks: 36**

Course Learning Outcomes:

1. To Understand the Rudiments of How computers communicate
2. To understand the operation on the components in a data communication systems and functional relationship of these components
3. To introduce the fundamental concepts of computer Network, topologies, protocols and functioning & significance of networking standards.
4. To provide knowledge of protocols, IP addressing and error detection & correction mechanisms.

Unit 1:

Data communication: Characteristics, Model, Data flow, Data representation, Analog and Digital Data, Analog and Digital Signals, Bit rate, Band width, Nyquest Bit rate, Shanon capacity, Data transmission modes, Parallel transmission, Serial transmission, Transmission impairments, Guided and Unguided transmission media.

Unit 2:

Digital-to-Digital conversion (NRZ, Manchester), Analog-to-Digital conversion (Sampling, PCM, Quantization), Digital to Analog conversion (ASK, FSK, PSK), Multiplexing, Frequency Division Multiplexing, Time Division Multiplexing, De-Multiplexing, Introduction to Modulation and Demodulation.

Unit 3:

Components of Network, Topologies, Categories of Networking: LAN, WAN, MAN. OSI reference model and TCP/IP model, Switching, Circuit switched networks, Datagram Networks, Virtual Circuit Networks, Introduction to Routing.

Unit 4:

Network addressing: Physical & Logical, Subnetting, UDP and TCP, IPV4, Classful addressing, Network Protocols: HTTP, FTP, SMTP, SNMP, DNS, Error Detection & Correction, Hamming Distance, Parity check, Cyclic Redudancy check, Checksum.

Text Book:

1. Data Communications and Networking Book by Behrouz A. Forouzan

Reference Book

1. Computer networks / Andrew S. Tanenbaum, David J. Wetherall.
2. Data and Computer Communications by W Stallings

Unit Wise List of Practicals for Data Communication and Networking.

Unit 1: Data Communication and Transmission Media

1. **Analog vs. Digital Signals:** Demonstrate the differences between analog and digital signals using waveform visualization.
2. **Calculating Bit Rate and Bandwidth:** Calculate the bit rate and bandwidth using the Nyquist formula and Shannon capacity.
3. **Transmission Media Testing:** Crimp a network cable (Cat 5/6) and test its continuity using a cable tester.
4. **Transmission Impairments Simulation:** Use Packet Tracer to simulate different transmission impairments (attenuation, distortion) and observe their effects.
5. **Comparing Guided and Unguided Media:** Compare guided media (coaxial, fiber-optic) and unguided media (wireless) through practical examples.

Unit 2: Data and Signal Conversion, Multiplexing 6. **Digital-to-Analog Conversion:** Use Packet Tracer to demonstrate digital-to-analog conversion using ASK, FSK, and PSK modulation techniques.

7. **Analog-to-Digital Conversion:** Simulate the process of analog-to-digital conversion using PCM and quantization techniques in Packet Tracer.
8. **Frequency Division Multiplexing (FDM):** Set up a simple FDM scenario using Packet Tracer to multiplex multiple signals onto a common medium.
9. **Time Division Multiplexing (TDM):** Create a TDM scenario in Packet Tracer to show how multiple signals are time-shared over a single channel.
10. **Introduction to Modulation and Demodulation:** Demonstrate the process of modulation and demodulation using simple audio signals and software tools.

Unit 3: Networking Components, Models, and Topologies 11. **Building a LAN:** Using Packet Tracer, set up a basic local area network (LAN) with computers, switches, and cables to illustrate a star topology.

12. **Comparing OSI and TCP/IP Models:** Explain the differences between the OSI model and the TCP/IP model through practical examples of layer functions.
13. **Configuring a Router:** Use a physical or virtual router to demonstrate basic router configurations, including IP address assignment.
14. **Virtual LAN (VLAN) Setup:** Configure VLANs on a network switch using Packet Tracer to create separate broadcast domains.
15. **Introduction to Routing:** Set up a small routed network using Packet Tracer to showcase the routing process between subnets.

Unit 4: Network Addressing, Protocols, and Error Detection 16. **Subnetting Practice:** Given an IP address range, guide students through subnetting to create multiple subnets with varying host ranges.

17. **Configuring IP Addresses:** In Packet Tracer, configure IP addresses for devices in a network, ensuring proper subnetting.

18. **Testing UDP and TCP:** Use Packet Tracer to demonstrate the differences between UDP and TCP by sending data between devices.

19. **Exploring Network Protocols:** Set up a variety of services (HTTP, FTP, SMTP, DNS) in Packet Tracer to showcase their functions within a network.

20. **Error Detection Techniques:** Implement error detection techniques like parity check and CRC in Packet Tracer to show data integrity maintenance.

Bachelor with Computer Applications as Minor
Third Semester

Course Code: ACP322N
Credits: Theory (4) Practical (2)

Course Title: OOPs with C++
Max. Marks: 100 Min. Marks: 36

Course Learning Outcome:

The student should understand object oriented programming and C++ concepts, improve his/her problem solving skills and should:

- Be able to explain the difference between object oriented programming and procedural programming.
- Be able to program using C++ features such as composition of objects, operator overloading, inheritance and polymorphism, file I/O, etc.
- Be able to build C++ classes using appropriate encapsulation and design principles.
- Be able to apply object oriented techniques to solve bigger computing problems.

UNIT-1 (15 HOURS)

Introduction to Object Oriented Programming

Comparison of Procedural Programming and OOP, Benefits of OOP, Abstraction, Encapsulation, Inheritance, Polymorphism, Difference between C and C++.

Elements of C++ Language:

Tokens and identifiers, Variables and Constants, Reference variables, Basic data types in C++, Streams in C++. Types of operators in C++.

Decision and Control Structures:

if statement, if-else statement, switch statement, Loop: while, do-while, for. Break and continue.

Pointers and structures.

UNIT-2 (15 HOURS)

Functions:

Inline function, function overloading

Introduction to Classes and Objects:

Classes in C++, class declaration, declaring objects, Defining Member functions, Inline member function, Array of objects, Objects as function argument, Static data member and member function, Friend function and friend class.

Constructors and Destructors:

Constructors, Instantiation of objects, Default constructor, Parameterized constructor, Constructor overloading, Copy constructor and its use, Destructors.

UNIT-3 (15 HOURS)

Operator Overloading:

Overloading unary and binary operators.

Inheritance:

Derived class and base class, access specifiers, type conversion, accessing the base class member, Types of Inheritance, Virtual base class, Abstract class

UNIT-4 (15 HOURS)

Virtual Functions and Polymorphism:

Virtual functions, pure virtual functions; Polymorphism. Compile time polymorphism, Run time polymorphism

File Handling:

Opening and Closing a file, File modes, Functions for I/O operations.

Introduction to standard template library.

Components of STL, Containers : Vector and lists.

Text Book:

i. E.Balagurusamy: *Object oriented programming with C++*

Reference:

i. Bjarne Stroustrup: *The C++ programming language.*
ii. Robert Lafore: *Object oriented programming in C++*
iii. Yashwant Kanathker: *Let's C++*
iv. Schildt. H: *C++ The Complete Reference*

OBJECT ORIENTED PROGRAMMING WITH C++

PRACTICALS (2 CREDITS: 30 HOURS) MAXIMUM MARKS 50, MINIMUM MARKS: 18

UNIT -1

1. Write a C++ program using decision making and loop structure for the given situation.
2. Use the structure in C++ program for solving the given problem.
3. Create C++ programs to perform the given arithmetic operations using pointers.
4. Write a program to Demonstrate the use of Streams in C++
5. Write a menu driven program in C++ using switch statement.

UNIT -2

1. Develop programs that implements a class and use it with objects.
2. Develop relevant friend functions to solve the given problem.
3. Use function overloading to solve the given problem.
4. Write a program to implement all types of constructors (constructor overloading).
5. Write program to delete the given object using destructor in C++ program.

UNIT -3

1. Write a program to demonstrate operator overloading for Unary operator.
2. Write a program to demonstrate operator overloading for Binary operator.'
3. Write a program for implementing single, multi-level, and multiple inheritance.
4. Write a C++ program using virtual base class.
5. Develop a program to demonstrate use of Pointer to derived class

UNIT -4

1. Write a C++ program demonstrating the use of pure virtual function.
2. Implement run time polymorphism using virtual functions in the given C++ program.
3. Develop C++ program to perform read/write operation from/to the given file.
4. Write a C++ program to illustrate the iterators in vector
5. Develop C++ program to demonstrate the use of list containers.

Bachelor with Computer Applications as Major/Minor
Fourth Semester

Course Code: CAPC1422
Credits: Theory (3) Practical (1)

Course Title: DBMS
Max. Marks: 100 **Min. Marks: 36**

Course Learning Outcomes:

- To introduce the core concept of Relational Database.
- To enable students to design the databases for a wide variety of Real World problems
- To introduce the concept and process of Database Normalization
- To enable the student to learn DML, DDL, DCL commands using SQL

Unit I:

Introduction to Databases, Database Users, Characteristics of Database approach, Applications of DBMS, Advantages and Disadvantages, Database System Concepts and Architecture, Data Models, Schemas, Instances, Three-Schema Architecture and Data Independence, Database System, Centralized and Client/Server Architecture for DBMS,

Unit II:

Data Modelling using Entity-Relationship Mode, Entity Types, Entity Sets, Attributes, and Keys, Relationship Types, Relationship Sets, ER Diagrams, Relational Data Model and Relational Model Constraints , Functional Dependencies , Normalization : 1NF, 2NF, 3NF

Unit III:

Relational Algebra and Relational Calculus, Introduction to DDL,DML,DCL. Introduction to transactions, Transaction states ACID properties, Concurrency Control Techniques: 2-phase locking, time stamp ordering.

Textbook:

1. R. Elmasri, S.B. Navathe, Fundamentals of Database Systems 6th Edition, Pearson Education, 2010

References:

1. An introduction to database systems by Desai, Bipin C
2. PL/SQL by Ivan Bayross

DBMS- LAB

Creation of Tables in SQL (DDL)

1. Overview of using SQL tool, Data types in SQL,
2. Create Schema and insert at least 5 records for each table.
3. Queries for Altering Tables (with add and modify clause) , Dropping Tables
4. Queries for DROP RENAME TRUNCATE and Describe a Table
5. SQL Constraints: NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK, DEFAULT

Practicing DML Commands

6. Insert, Select, Update, Delete
7. Select Command with WHERE, IN, BETWEEN AND LIKE clause
8. SELECT with AND, OR, NOT
9. Practice Queries using COUNT, SUM, AVG, MAX, MIN, GROUP BY, HAVING.

Joining Tables

10. Practice the Inner Join, Left Join,
11. Practice the Right Join, Cross Join

Create the following Database Schema and Practice the queries

EMP (eno. ename, bdate, title, salary, dno)

PROJECT (pno, pname, budget, dno)

DEPT (dno. dname, mgreno)

WORKSON (eno, pno, resp, hours)

12. Write an SQL query that returns the project number and name for projects with a budget greater than 100,000.
13. Write an SQL query that returns all works on records where hours worked is less than 10 and the responsibility is 'Manager
14. Write an SQL query that returns the employees (number and name only) who have a title of 'EE' or 'SA' and make more than 35,000.
15. Write an SQL query that returns the employees (name only) in department 'D1' ordered by decreasing salary.
16. Write an SQL query that returns the departments (all fields) ordered by ascending department name.
17. Write an SQL query that returns the employee name, department name, and employee title.
18. Write an SQL query that returns the project name, hours worked, and project number for all works on records where hours > 10.
19. Write an SQL query that returns the project name, department name, and budget for all projects with a budget < 50,000.
20. Write an SQL query that returns the employee numbers and salaries of all employees in the 'Consulting' department ordered by descending salary.
21. Write an SQL query that returns the employee name, project name, employee title, and hours for all works on records.

Bachelor with Computer Applications as Major
Fourth Semester

Course Code:CAPC2422 **Course Title: Java Programming**
Credits: Theory (4) Practical (2) **Max. Marks: 100** **Min. Marks: 36**

COURSE LEARNING OUTCOMES:

- *To understand Object Oriented Concepts using Java Language.*
- *To develop, debug and document programs in Java using OOP paradigms.*

UNIT-I (15 HOURS)

Introduction to Java Language its history, features and evolution. 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.

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-II (15 HOURS)

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.

UNIT-III (15 HOURS)

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 User- defined. Catching an Exception - try-catch-finally. Specifying an Exception - throws. Manually throwing an Exception - throw. Custom Exceptions. Chained Exceptions

UNIT-IV (15 HOURS)

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. Introduction to JDBC:Connecting JAVA applications with MYSQL Database for performing CRUD Operations.

REFERENCES:

1. Java: The Complete Reference by Herbert Schildt
2. Core Java Fundamentals 11th Edidition (John Wiley & Sons, Inc.) by Cay Horstmann.
3. Java, How to Program (Prentice-Hall) by Deitel and Deitel
4. Java in a Nutshell (O'Reilly) by David Flanagan

PRACTICAL (2 CREDITS: 30 HORS) **MAX. MARKS: 50 MIN. MARKS: 18**

1. Download latest version of Java Development Kit (JDK), preferably JDK8 or above
2. Follow the instructions that appear during the Installation of JDK8, and set PATH variable to the appropriate directory location as instructed in the lecture.
3. Write a Java program that displays "hello world!" on the screen
4. 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
5. Write a Java program that prints the season name corresponding to its month number using If-else and switch-case statements.
6. Write a Java program that sorts (using bubble sort) an integer array using for loop.
7. Write a Java program that calculates factorial of a number (inputted via keyboard) recursively.
8. 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.
9. Write a Java program that reads an integer from keyboard and displays it.
10. Write a Java program that reads a floating-point number from keyboard, converts it to integer and displays it.
11. Write a Java program that reads a string from keyboard, converts it to a floating-point number and displays it.
12. Write a Java program that populates all the 10 elements of an integer array using keyboard input, increments every element by 1, and displays every element.
13. Write a Java program that iteratively calculates factorial of a number.
14. 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.
15. 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*.
16. 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).
17. 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.
18. 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.
19. 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)
20. 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.
21. Write a Java program to count the total number of occurrences of a given character in a string.
22. Write a Java program to convert a string to char array.

23. 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.
24. Write a Java program that creates a Class that implements interface *Runnable*, and does the same as the above program.
25. Write a Java program to demonstrate applets by extending an *Applet* class.

Bachelor with Computer Applications as Major

Fourth Semester

Course Code: CAPC3422 **Course Title: Computing Mathematics**
Credits: Theory (4) Practical (2) **Max. Marks: 100** **Min. Marks: 36**

Course Learning Outcomes:

- To introduce elements of 10+2 level mathematics to students of Computer Applications who are from a medical or arts background
- To cover fundamental concepts of matrices and determinants
- To cover fundamental concepts of calculus.
- To acquire fundamental knowledge regarding the problems of approximation and errors in Computer based numerical problems solving.

Unit 1: Matrices and Determinants:

Matrix, Types of Matrices, Matrix Operations: Add, Sub, Multiply, Divide, Transpose. Determinants, Minors and Cofactors, Properties of Determinants. Elementary Transformations. Solutions of Linear Equations by Matrix Method and Determinants.

Unit 2: Functions and Limits

Functions, Concept, Domain and Range, Types of Functions
Limits and Continuity: Limit of a Function, Algebra of Limits, Simplification and Evaluation of Limits. Standard Limits, Continuity, Geometric Meaning

Unit 3: Differentiation

Ab-Initio Method, Derivatives of some Common Functions, Rules to find Derivatives, Second Order Derivatives, Anti-Derivative of a Function.

Unit 4: Approximation and Errors:

Mathematical Modelling and Engineering Problem-Solving, Role of Computers and Software. Significant Figures, Accuracy and Precision. Numerical Error. Floating Point Representation. Computer Arithmetic. Round-Off, Truncation Errors and Error Propagation.

Textbook:

1. Wiley's Problems in Mathematics for JEE Main and Advanced (Vol 1 + 2)
2. Introduction to Linear algebra by Gilbert Strang, 6th Edition

References:

1. Numerical Methods for engineers. S C Chapra and R P Canale , McGrow Hill International Edition.
2. Numerical Methods for Scientific & Engineering Computation, M. K. Jain, S.R.K.
3. Numerical Methods in Science & Engineering Prog. - By Dr. B. S. Grawal, Khanna Pub., New Delhi.
4. R S Aggarwal, Senior Secondary School Mathematics for Class XI/XII (Bharati Bhawan)
5. R.D Sharma – Mathematics for Class 11th / 12th (4 Vols.)
6. Computer Oriented Numerical Methods, R. S. Salaria., Khanna Publisher.

Tutorials:

1. Show how to use determinants to calculate the area of a triangle from its coordinates.
2. Show how to use determinants to determine that a set of given points are collinear.
3. Show how to use determinants to solve a non-homogenous system of linear equations
4. Show how to use determinants to solve a homogenous system of linear equations
5. Show that the value of a determinant remains unchanged if its rows and columns are interchanged
6. Show that if each element of a row or a column of a determinant is multiplied by a constant k then the value of the new determinant is k times the value of the original determinant
7. Show the different ways of representing a mathematical function. Also, given a mathematical function, show how to plot its graph.
8. Use limits to determine the tangent to the curve at a given point
9. Use limits to determine the instantaneous velocity of a ball dropped from a height at a given time.
10. Use differentiation to compute the rate of change of one quantity from the rate of change of another, more easily measurable quantity (e.g. the rate of increase of the radius of a balloon from the rate of change of its volume)
11. Show how to use the derivative of a function as a measure of rate of change
12. Show how to use the derivative to calculate the maxima and minima of a function
13. Write an algorithm for finding the roots of a quadratic equation
14. Develop a simple mathematical model to determine the terminal velocity of a free-falling body (a parachutist) near the earth's surface. Present a numerical solution for this problem.

References:

- Stewart, Calculus Concepts and Contexts (2nd Edition)
- R.S. Aggarwal, Mathematics XI/XII
- Numerical Methods for engineers. S C Chapra and R P Canale

Bachelor with Computer Applications as Minor (Applied Computing)
Fourth Semester

Course Code: ACP422N
Credits: Theory (3) Practical (1)

Course Title: Fundamentals of IOT
Max. Marks: 100 Min. Marks: 36

Course Learning Outcomes:

- Understand the fundamental characteristics of IoT, including its physical design, basic components, and the concepts of things, sensing, and actuators.
- Explore various application areas of IoT such as home automation, smart cities, medical, logistics, environment, analytics, and smart grids.
- Gain insights into IoT protocols used for communication and data exchange within IoT ecosystems.
- Develop hands-on skills in working with hardware platforms like Raspberry Pi and Arduino, and learn how to implement basic sensors for monitoring temperature, humidity, proximity, gas, air quality, and ultrasonic sensors.

UNIT - 1 (15 Hours)

Introduction: Definition & Characteristics of Iot, Physical Design of Iot, Basic Components of IoT, Thing, Sensing & Actuators, Vision, Physical Parameters. Iot Protocols.

Application Areas of IoT: Home Automation, Smart Cities, Medical, Logistics, Environment, Analytics. Smart Grids.

UNIT - 2 (15 Hours)

Iot Communication Models, APIs, IoT Architecture: Basic Architecture: 3 layer and 5 layer Architecture, ITU-IoT Reference Model, Machine to Machine Communication, IoT Gateways, Wireless Sensor Networks. Technologies: Bluetooth Low Energy(BLE), ZigBee: Architecture, Comparison with other wireless standards. LoraWAN.

UNIT - 3 (15 Hours)

Electronic Product Code (EPC), Near Field Comm.(NFC), 6LoWPAN, End to End Reliability: MQTT, SCADA. Hardware and Software Platforms: Hardware: Raspberry Pi, ESP8266 Wifi Module, Arduino. Implementation of Basic Sensors (temperature, humidity, proximity, gas, air quality, Ultrasonic sensors)

Internet of Things Privacy and Security Issues, Steps towards a security platform in IoT

Text Books:

1. Vijay Madisetti and Arshdeep Bahga, "Internet of Things (A Hands-on-Approach)", 1stEdition, VPT, 2014. (ISBN-13: 978-8173719547)
2. Internet of Things (IoT), Dr. Kamlesh Lakhwani, Dr. Hemant Kumar Gianey, Joseph Kofi Wireko, Kamal Kant Hiran

References:

1. Schwartz, Marco. "Internet of Things with Arduino Cookbook". Packt Publishing Ltd, 2016.
2. Jan Holler, Vlasios Tsiatsis, Catherine Mulligan, Stefan Avesand, Stamatis Karnouskos, David Boyle, "From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence", 1st Edition, Academic Press, 2014. (ISBN-13: 978-0124076846)
3. Hakima Chaouchi "The Internet of Things Connecting Objects to the Web" by Wiley publications

PRACTICAL (2 CREDITS: 30 HORS) **MAX. MARKS: 50 MIN. MARKS: 18**

Learning Outcome: These practical's cover a range of fundamental concepts in IoT, including hardware interfacing, sensor integration, data communication, cloud integration, and real-world applications. Students can gain hands-on experience and a solid understanding of the core principles of IoT through these practical exercises.

1. Study the fundamental of IOT software's and components. Install Arduino IDE development platform.
2. Connect a microcontroller chip on a breadboard, establish power connections, and verify basic functionality. Gather the necessary components: microcontroller chip (e.g., Arduino Uno), breadboard, jumper wires, USB cable for power, and an LED.
3. To practice connecting analog input and output components to the microcontroller using the breadboard.
4. Write a program to Read sensor data using analog or digital pins.
5. LED Blinking Using Arduino: To interface LED/Buzzer with Arduino/Raspberry Pi/ ESP8266
6. and write a program to turn ON & OFF.
7. To interface Push button/Digital sensor (IR/LDR) with Arduino/ ESP8266/Raspberry Pi and write a program to turn ON LED when push button is pressed or at sensor detection. Upload the program to the board and observe the blinking.
8. To interface DHT11 sensor with Arduino/Raspberry Pi/ ESP8266 and write a program to print temperature and humidity readings and Display sensor readings on the serial monitor.
9. IoT Cloud Platform Integration: Create an account on a popular IoT cloud platform ThingSpeak.
10. Write a program to Send temperature and humidity sensor data to the cloud and visualize it on a dashboard.
11. Write a Program to interface motor using relay with Arduino/Raspberry Pi/ ESP8266 and write a program to turn ON motor when push button is pressed.
12. Build a web server using Arduino and Ethernet/Wi-Fi shield using ESP8266 Wifi module. Develop a web page to control LEDs remotely.
13. Use buttons on the webpage to toggle LED states.
14. Log data at regular intervals to an SD card or cloud platform. Create graphs to visualize the logged data over time.
15. Set up an MQTT broker (e.g., Mosquitto) on a computer or cloud server. Program an Arduino to publish sensor data and subscribe to commands via MQTT.
16. Create a soil moisture sensing system using Arduino and moisture sensors. Send data to the cloud to monitor soil conditions remotely.
17. Design a basic home automation system with Arduino and relays.