POST GRADUATE DEPARTMENT OF COMPUTER SCIENCES

University of Kashmir, Srinagar-190006

NAAC Accredited Grade "A+"

NOTES

Minutes of BOS (Undergraduate) Meeting held on 28-11-2023 at 11:00 a.m in Computer Science Department

A Board of Studies meeting was held on 28-11-2023 at 11:00 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. Prof. M Arif Wani	Member
3. Dr. Manzoor Ahmad Chachoo	Member
4. Dr. Sajad M. Khan	Member
5. Prof. Mohd Ashraf Shah	Consultant U.G CBCS/NEP
6. Dr. Faheem Syeed Masoodi	Member
7. Mr. Gazi Imtiyaz	Member
8. Mr. Zubair Sayeed Masoodi	Member
9. Mr. Audil Hussain	Member
10. Ms. Aasiya Qayoom	Member
11. Mr. Faisal Maqbool	Member (Cluster University, Srinagar.)
12. Mr. Sajad Ahmad Shah	Member
13. Mr. Shabir Ahmad Rather	Member
14. Dr. Tawseef Ahmad Teli	Member

The committee recommends the scheme and structure (Annexure–I) for the entire FYUP. Moreover, the committee recommends the syllabi for the courses of 5th and 8th 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.

[illegible]
Dr. Javaid Iqbal
(Chairman)

[illegible]
Prof. M. Arif Wani
(Member)

[illegible]
Dr. Manzoor A. Chachoo
(Member)

[illegible]
Prof. Mohd Ashraf Shah
Consultant U.G CBCS/NEP

[illegible]
Dr. Sajad M. Khan
(Member)

[illegible]
Dr. Faheem Masoodi
(Member)

[illegible]
Mr. Gazi Imtiyaz
(Member)

[illegible]
Mr. Zubair S. Masoodi
(Member)

[illegible]
Mr. Audil Hussain
(Member)

[illegible]
Ms. Aasiya Qayoom
(Member)

[illegible]
Mr. Faisal Maqbool
Member

[illegible]
Mr. Sajad Ahmad Shah
(Member)

[illegible]
Mr. Shabir Ahmad Rather
(Member)

[illegible]
Dr. Tawseef Ahmad Teli
(Member)

Bachelor with Computer Applications as Major/Minor

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 (Applied Computing)

Third Semester

Course Code: __________	**Course Title: Digital Electronics**
Credits: Theory (4) Practical (2)	**Max. Marks: 100 Min. Marks: 36**

Course Learning Outcomes:

- To introduce concepts of number systems and Boolean algebra.
- To familiarize students with the operation and use of basic digital logic gates as well as the design and minimization of combinational logic circuits.
- To introduce the concept of microprocessors and familiarize them with basic operation of a CPU.

Unit 1:

Introduction to Digital and Analog Quantities, Binary Digits, Logic Levels, Pulse, Waveforms, Clock and Timing Diagrams *(1 Hour)*
Number Systems – Decimal, Binary, Octal, Hexadecimal and their Conversions. *(4 Hours)*
Unsigned Binary Arithmetic, Ones Complement, Twos Complement. Signed Numbers and their arithmetic. Binary Coded Decimal. Error Codes-Parity Code *(4 Hours)*
Logic Gates—AND, OR, NOT, NAND, NOR, XOR and XNOR Gates. *(2 Hours)*
Boolean Algebra: Boolean Operations, Laws and Rules of Boolean Algebra, DeMorgan's Theorems. Constructing a Boolean Expression for a Logic Circuit, Logic Simplification. *(5 Hours)*

Unit 2:

SOP and POS forms, Karnaugh Maps and minimization upto 4 variables, Don't care conditions *(4 Hours)*
Combinational Logic Circuits: AND-OR, AND-OR-INVERT, XOR and XNOR logic, Converting Boolean Expression or Truth Table to a Logic Circuit, NAND and NOR as Universal Gates *(4 Hours)*
Half Adder, Full Adder, 4-bit Parallel Binary Adder, Comparator, Binary Decoder, Encoder, Multiplexer, Demultiplexer *(7 Hours)*

Unit 3:

Latches: SR Latch, D Latch, Gated SR and D Latch *(2 Hours)* Flip Flops: Difference between Flip Flop and Latch, Level vs Edge-Triggered. D Flip Flops, JK Flip Flops and their operation *(4 Hours)*
Characteristics and Applications of Flip Flops (storage, counting), Intro to 555 Timer *(2 Hours)*
Shift Registers – Serial and Parallel (4-bit) *(3 Hours)* Counters: Synchronous and Asynchronous (2/3 bit). Decade Counter, Johnson counter *(4 Hours)*

Unit 4:

von Neumann Architecture: Block Diagram, CPU, Memory, I/O Ports and Buses, Bus Master, Bus Contention: Shared Signal Lines and Tri-State Outputs, Fan-out, Buffers, Device Selection, System Timing.
Microprocessor, ALU, Control/Timing Unit, Decode Unit, Register Set, Instruction Execution Cycle. Memory: Memory Bus, Read / Write operations and Addressing Modes. I/O: Polling, Interrupts and DMA. Intro to Types of CPU Instructions.
Microcontrollers: Architecture, Registers, Functional Units and Peripherals. System on Chip (SoC): Block Diagram, Functional Elements, Difference between Microprocessor, Microcontroller and SoC. *(15 Hours)*

Textbook:

1. Thomas Floyd, Digital Fundamentals, 11th Edition (2015), Pearson.

References:

1. Morris Mano, Michael Ciletti, Digital Design with an Introduction to the Verilog HDL, VHDL, and SystemVerilog, 6th Edition, Pearson (2017)
2. Malvino, Principle of Digital Electronics, McGraw-Hill
3. R.P. Jain - Modern Digital Electronics, McGraw-Hill, 4th ed. 2010
4. LaMerez, Quick Start Guide to Verilog, Springer (2019)
5. M. Rafiquzzaman - Digital Logic, with an Introduction to Verilog and FPGA-Based Design, Wiley (2019)

Practicals:

User a Verilog/SystemVerilog simulator like ModelSim or Icarus Verilog to simulate the following digital circuits:

1. Implement the following logic gates in Verilog:
   a. A 2-input AND Gate
   b. A 3-input OR Gate
2. Implement the following logic gates in Verilog and simulate them using a test bench:
   a. A 2-input NAND Gate
   b. A 2-input NOR Gate
3. Design and simulate 3-input XOR gate in Verilog.
4. Design and simulate a module in verilog that implements the following boolean logic:
   a. $y = \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + A\bar{B}C$
5. Design and simulate a half adder in Verilog
6. Design and simulate a full adder in Verilog
7. Design and simulate a 4-bit binary adder in Verilog
8. Design and simulate a 2-to-1 multiplexer in Verilog
9. Design and simulate a 4-to-1 multiplexer assembled from three 2-to-1 multiplexers in Verilog
10. Design and simulate a 3-to-8 decoder in Verilog
11. Design and simulate a clocked D Flip Flop with reset input in Verilog
12. Design and simulate a simple 4-bit ALU in Verilog that performs addition, subtraction, AND, and OR operations

Bachelor with Computer Applications as Major/Minor

Fourth Semester

Course Code: CAPC1422 **Course Title: DBMS**

Credits: Theory (3) Practical (1) **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 (<u>eno,</u> ename, bdate, title, salary, dno)

PROJECT (<u>pno</u>, pname, budget, dno)

DEPT (<u>dno,</u> dname, mgreno)

WORKSON (<u>eno</u>, <u>pno</u>, 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: ACP322N	**Course Title: OOPs with C++**
Credits: Theory (4) Practical (2)	**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

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:

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

References:

6. Numerical Methods for engineers. S C Chapra and R P Canale , McGrow Hill International Edition.
7. Numerical Methods for Scientific & Engineering Computation, M. K. Jain, S.R.K.
8. Numerical Methods in Science & Engineering Prog. - By Dr. B. S. Grawal, Khanna Pub., New Delhi.
9. R S Aggarwal, Senior Secondary School Mathematics for Class XI/XII (Bharati Bhawan)
10. R.D Sharma – Mathematics for Class 11th / 12th (4 Vols.)
11. 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	**Course Title: Fundamentals of IOT**
Credits: Theory (3) Practical (1)	**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.

Bachelor with Computer Applications as Major/Minor (CT-1)

Fifth Semester

Course Code:	**Course Title: Operating System**
Credits: Theory (3) Practical (1)	**Max. Marks: 100 Min. Marks: 36**

Course Learning Outcomes:

At the end of this course, a student should be able to:

- To learn and understand the concepts of operating system.
- To understand the core structure and functions of operating system.
- Explain process management, processor scheduling, and concurrent programming.
- Understand the concept of deadlocks and synchronization.
- Learn memory management, file management, I/O systems, and disk scheduling.
- Distinguish main memory and virtual memory.
- Learn Operating System Design issues.
- Learn basic Unix commands and shell programming.

UNIT – 1 (15 HOURS)

Introduction: Goals & Structure of Operating System, Basic functions & Modes, System calls, Types of Operating System (Batch, Multiprogramming, Multitasking, Time Sharing, Parallel, Distributed & Real-Time O.S.)

Process: Process Concept, Process states, Threads, Uni-processor Scheduling: Types of scheduling (Preemptive & Non-preemptive), Scheduling algorithms (FCFS, SJF, RR, Priority, Thread Scheduling, Real-Time Scheduling).

UNIT -2 (15 HOURS)

Concurrency: Principles of Concurrency, Mutual Exclusion: Software approaches, Hardware Support, Semaphores, Message Passing, Signals, Monitors, Classical Problems of Synchronization (Readers-Writers, Producer-Consumer, and Dining Philosopher problem).

Deadlock: Principles of Deadlock, Deadlock Prevention & Avoidance, Deadlock Detection.

UNIT – 3 (15 HOURS)

Memory: Memory Management requirements, Memory partitioning (Fixed and Variable Partitioning), Memory Allocation Strategies (First Fit, Best Fit, and Worst Fit), Fragmentation, Swapping. Segmentation, Paging.

Virtual Memory: Virtual Memory Concepts and Implementation. Concept of Demand Paging, Page Replacement Policies (FIFO, LRU, Optimal, Other Strategies), Thrashing.

UNIT- 4 (15 HOURS)

Input/Output: I/O Devices, Organization of I/O functions, I/O Buffering, Disk Scheduling (FCFS, SCAN, C-SCAN, SSTF), RAID, Disk Cache, Operating System Design issues.

Unix: Unix Components/Architecture. Features of Unix. The UNIX Environment and UNIX Structure. Basic Unix commands (echo, printf, ls, who, date, passwd, cal, pwd, cd, mkdir, rmdir, cat, mv, rm, cp, wc and od) Combining commands. Pipe. Basic and Extended regular expressions. The grep, egrep. Typical examples involving different regular expressions, file permissions.

Shell programming: variables, .profile, read and readonly commands, Command line arguments, if, while, for and case control statements. The set and shift commands and handling positional parameters.

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

1. Write a script to implement echo, printf, and ls commands.
2. Write a script to implement who, date, passwd, and cal command.
3. Write a script to implement Directory commands (pwd, cd, mkdir, rmdir)
4. Write a script to implement File related commands (cat, mv, rm, cp, wc and od)
5. Write a script to exhibit use of Pipes in Unix.
6. Write a script to display length of a given string.
7. Write a script to extract a substring from the given string.
8. Write a script to find and replace a string.
9. Write a script to check if a given number is prime.
10. Write a script to generate factorial of a given number.

Reference Books:

- *Peter B. Galvin, Greg Gagne, Abraham Silberschatz: Operating System Concepts, John Wiley & Sons, Inc.*
- *Andrew S. Tanenbaum: Modern Operating Systems, PHI.*
- *William Stallings: Operating Systems, Pearson Education India.*
- *M.G. Venkatesh Murthy: UNIX & Shell Programming, Pearson Education.*
- *Richard Blum , Christine Bresnahan : Linux Command Line and Shell Scripting Bible, Wiley.*

Bachelor with Computer Applications as Major/Minor (CT-1)

Fifth Semester

Course Code:	**Course Title: Operating System**
Credits: Theory (3) Practical (1)	**Max. Marks: 100 Min. Marks: 36**

Course Learning Outcomes:

At the end of this course, a student should be able to:

- To learn and understand the concepts of operating system.
- To understand the core structure and functions of operating system.
- Explain process management, processor scheduling, and concurrent programming.
- Understand the concept of deadlocks and synchronization.
- Learn memory management, file management, I/O systems, and disk scheduling.
- Distinguish main memory and virtual memory.
- Learn Operating System Design issues.
- Learn basic Unix commands and shell programming.

UNIT – 1 (15 HOURS)

Introduction: Goals & Structure of Operating System, Basic functions & Modes, Types of Operating System (Batch, Multiprogramming, Multitasking, Time Sharing, Parallel, Distributed & Real-Time O.S.), System calls.

Process: Process Concept, Process states, Threads, Uni-processor Scheduling: Types of scheduling (Preemptive & Non-preemptive), Scheduling algorithms (FCFS, SJF, RR, Priority, Thread Scheduling).

UNIT -2 (15 HOURS)

Concurrency: Principles of Concurrency, Mutual Exclusion: Software approaches, Hardware Support, Semaphores, Message Passing, Signals, Monitors, Classical Problems of Synchronization (Readers-Writers, Producer-Consumer, and Dining Philosopher problem).

Deadlock: Principles of Deadlock, Deadlock Prevention & Avoidance, Deadlock Detection.

UNIT – 3 (15 HOURS)

Memory: Memory Management requirements, Memory partitioning (Fixed and Variable Partitioning), Memory Allocation Strategies (First Fit, Best Fit, and Worst Fit), Fragmentation, Swapping. Segmentation, Paging.

Virtual Memory: Virtual Memory Concepts and Implementation. Concept of Demand Paging, Page Replacement Policies (FIFO, LRU, Optimal, Other Strategies), Thrashing.

Input/Output: I/O Devices, Organization of I/O functions, I/O Buffering, Disk Scheduling (FCFS, SCAN), RAID, Disk Cache.

PRACTICALS (1 CREDIT: 30 HOURS)

1. Installing and setting up of Linux/Unix Operating system.
2. Using Basic Unix commands like echo, printf, ls, who, date, passwd, cal.
3. Using pwd, cd, mkdir, rmdir, cat commands
4. Using mv, rm, cp, wc and od commands.
5. Demonstrate the use of Pipes.
6. Using Basic and Extended regular expressions.
7. Implementing file permissions, printing commands,
8. Mount devices like usb drives in UNIX/Linux.
9. Write a shell script to demonstrate the use of variable & basic input output on the console.
10. Write a script to demonstrate the use of control statements
11. Write a script to demonstrate the use of looping structures.
12. Write a script to display length of a given string.
13. Write a script to extract a substring from the given string.
14. Write a script to find and replace a string.
15. Write a script to check if a given number is prime.

Text Book:

1. *Peter B. Galvin, Greg Gagne, Abraham Silberschatz: Operating System Concepts, John Wiley & Sons, Inc.*
2. *Shell Programming by Yashwant Kanetkar*

Reference Books:

1. *Peter B. Galvin, Greg Gagne, Abraham Silberschatz: Operating System Concepts, John Wiley & Sons, Inc.*
2. *Andrew S. Tanenbaum: Modern Operating Systems, PHI.*
3. *William Stallings: Operating Systems, Pearson Education India.*
4. *M.G. Venkatesh Murthy: UNIX & Shell Programming, Pearson Education.*
5. *Richard Blum , Christine Bresnahan : Linux Command Line and Shell Scripting Bible, Wiley.*

Bachelor with Computer Applications as Major (CT-2)

Fifth Semester

Course Code: **Course Title: Data Structures using C**
Credits: Theory (4) Practical (2) **Max. Marks: 100** **Min. Marks: 36**

OBJECTIVES:

- *To introduce the fundamentals of Data Structures, Abstract concepts and how these concepts are useful in problem solving.*
- *To learn the linear and non-linear data structures.*
- *To explore the applications of linear and non-linear data structures.*
- *To learn to represent data using tree and graph data structure.*
- *To learn the basic sorting and searching algorithms.*
- *To write programs for different Data Structures and Algorithms*

Unit-I: Linear Data Types – I (15 Lectures)
Introduction to data structure (Linear, Non Linear, Primitive, Non Primitive), Data Structure and Data Operations, Algorithm Complexity
Single dimensional array and its operations (Searching, traversing, inserting, deleting), Two Dimensional array, Addressing, Sparse Matrices, recursion.
Searching Algorithms: Linear Search, Binary Search and their Comparison.

Unit –II: Linear Data Types – II (15 Lectures)
Array Sorting Algorithms: Selection sort. Insertion sort, Bubble Sort, Quick sort.
Stack: Definition & Concepts, Array Representation of Stack, Operations on Stack. Applications of Stack: Expressions and their representation: Infix, Prefix, Postfix and their conversions & evaluation.
Queue: Definition & Concepts, Array Representation of Queue, Operations on Queue, Circular Queue, Applications of Queue.

Unit-III: Linear Data Types – III (15 Lectures)
Review of structures & pointers.
Introduction to Linked Lists and their applications: Singly linked list: Definition & Concepts, representation in memory, operations on singly linked list (insertion, deletion, traversal, reversal). Variations of Linked List (Doubly linked list, circular linked list)
Linked list implementation of Stack and Queue.

Unit-IV: Non- Linear Data Types (15 Lectures)
Trees: Introduction to trees, terminology, Binary Trees, Binary tree representation and traversals (in-order, pre-order, post-order), Binary Search Trees, Applications of trees.
Graphs: Introduction, terminology, linked and matrix representation, Traversing a Graph: BFS, DFS. Dijkstra's Shortest Path Algorithm. Applications of graphs.

TEXT BOOK:

1. Fundamentals of Data Structures in C: Ellis Horowitz, Sartaj Sahni, Susan Anderson-Freed.
2. Data Structures & Algorithms; Concepts, Techniques & Algorithms: G. A. V. Pai Tata McGraw Hill.
3. Systematic approach to data structure using C: A M Padma Reddy Sri Nandi Publications

REFERENCES:

1. Mark Allen Weiss, ―Data Structures and Algorithm Analysis in C, Second Edition, Pearson Education, 1996
2. Alfred V. Aho, John E. Hopcroft and Jeffrey D. Ullman, ―Data Structures and Algorithms, Pearson Education, 1983.
3. Robert Kruse, C.L.Tondo, Bruce Leung, Shashi Mogalla , ― Data Structures and Program Design in C, Second Edition, Pearson Education, 2007
4. Jean-Paul Tremblay and Paul G. Sorenson, ―An Introduction to Data Structures with Applications, Second Edition, Tata McGraw-Hill, 1991.
5. Pradip Dey and Manas Ghosh, ―Programming in C, Second Edition, Oxford University Press, 2011.
6. Data Structures Through C In Depth S.K.Srivastava, Deepali Srivastava BPB Publications
7. Data Structure Through C Yashavant P. Kanetkar BPB Publications

PRACTICALS (02 CREDITS: 60 HOURS)

1. Write a C Program to demonstrate the concept of one-dimensional array.
2. Write a C Program to insert and delete element in an array.
3. Write a C Program to Search an element using sequential search
4. Write a C Program to Search an element using binary search
5. Write a C Program to Add two 2-D Matrices
6. Write a C Program to multiply two 2-D Matrices
7. Write a C Program to read a 2-D Matrix and create its equivalent sparse representation.
8. Write a C Program to add two sparse matrices.
9. Write a C Program to implement recursion (factorial)
10. Write a C Program to arrange the list of numbers using Bubble Sort
11. Write a C Program to arrange the list of numbers using Insertion Sort
12. Write a C Program to arrange the list of numbers using Selection Sort.
13. Write a C Program to arrange the list of numbers using quick sort.
14. Write a C Program to implement stack operations using array.
15. Write a C Program to implement basic operations on queue.
16. Write a C Program to implement a circular queue.
17. Write a C Program to convert infix expression to its postfix form using stack operations.
18. Write a C Program to evaluate the given postfix expression using stack operations.
19. Write a C Program to create a singly linked list and perform operations such as insertions and deletions.
20. Write a C Program to reverse a singly linked list.
21. Write a C Program to implement stack and its operations using linked representation.
22. Write a C Program to implement queue and its operations using linked representation.

23. Write a C Program to implement basic operations on doubly linked list.
24. Write a C Program to implement basic operations creation of Binary Search tree.
25. Write a C Program to implement basic operations insertion, search, find min and find max on Binary Search tree.
26. Write a C Program to implement basic operations deletion on Binary Search tree.
27. Write a C Program to implement tree traversal methods
28. Write a C Program to implement breadth first graph traversal
29. Write a C Program to implement Depth first graph traversal.
30. Write a C Program to implement Dijsktra’s Shortest Path Algorithm on a graph.

Bachelor with Computer Applications as Major (CT-3)

Fifth Semester

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

Course Learning Outcomes:

1. To be able to understand mathematical reasoning in order to read, comprehend, and construct mathematical arguments.
2. To be able to count or enumerate objects, and use basic techniques of counting to solve counting problems.
3. To be able to work with discrete structures such as sets, permutations, relations, graphs, and trees, and use them to represent discrete objects and the relationships between these objects.

Unit 1: (15 Hours)

Sets, Types, Set Operations, Venn Diagrams, Cardinality, Set Properties and Identities.
Relations, Cartesian Product, Properties of Relations, Types of Relations, Representation of Relations, Closures, Equivalence Relations.
Functions, Domain, Codomain, Range, Types of Functions (one-to-one, many-to-one, onto), Composition of Functions. Algebra of Functions.

Unit 2: (15 Hours)

Logic: Propositions, Variables and Logical Operators with Truth Tables. Conditionals and Biconditionals. Converse, Contrapositive, and Inverse of a Proposition.
Truth Tables of Compound Propositions and their equivalence. Tautology, contradiction and contingency. De Morgan's Laws and other Logical Equivalences (laws). Satisfiability.
Predicates. Universal and Existential Quantifiers. Logical Equivalences Involving Quantifiers, Negation and De Morgan’s Laws for Quantifiers. Converting English into Logical Expressions. Nested Quantifiers, Negation and Order of Nested Quantifiers. Rules of Inference.

Unit 3: (15 Hours)

Introduction to Matrices, operations on matrices (addition, multiplication, transpose). Adjacency Lists, Adjacency Matrices, Incidence Matrices. Counting: Pigeonhole Principle and its Applications.
Permutations, Combinations, Permutations and Combinations with Repetition, Binomial Theorem. Summations, Introduction to Recurrence Relations.

Unit 4: (15 Hours)

Graphs, Terminology, Types: Simple, Multigraphs, Directed, Undirected, Complete, Bipartite, Subgraphs. Graph isomorphism. Graph Connectivity, Euler and Hamilton Paths and Circuits, Konigsberg Bridge Problem. Cliques. Graph Coloring: Four Color Theorem.
Trees: Terminology, Arity, Ordered Rooted Trees, Unbalanced and Balanced Trees, Minimal Spanning Trees (Kruskal’s & Prim’s Algorithm).

TUTORIAL: 2 CREDITS/ 60 HOURS **MAXIMUM MARKS: 50, MINIMUM MARKS: 18**

Discrete Mathematics (Tutorial)

1. Find whether $(p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)$ is a tautology or a contradiction?
2. Prove the following De-Morgan's law using truth table:
   $\neg (p \wedge q)$ is logically equivalent to $(\neg p \vee \neg q)$
3. State which rule of inference is used in the argument:
   If it rains today, then we will not have a barbecue today. If we do not have a barbecue today, then we will have a barbecue tomorrow. Therefore, if it rains today, then we will have a barbecue tomorrow.
4. Prove by Mathematical Induction that:
   $2 + 6 + 10 + \ldots + (4n - 2) = 2n^2$
5. Give direct proof of the theorem "If n is an odd integer, n^2 is odd."
6. Prove by contraposition that if n is an integer and 3n+2 is odd, then n is odd.
7. Prove that if n=ab, where a and b are positive integers, then $a \leq \sqrt{n}$ or $b \leq \sqrt{n}$.
8. Prove that $\sqrt{2}$ is irrational using proof by contradiction.
9. Let f and g be the functions from the set of integers to a set of integers defined by $f(x) = 2x+3$ and $g(x) = 3x+2$. What is the composition of f and g? What is the composition of g and f?
10. a). Show that $f(x) = x^2+2x+1$ is $O(x^2)$.

    b). Show that n^2 is not $O(n)$.

11. Determine whether the function $f(x) = x^2$ from the set of integers to a set of integers is one-to-one.
12. How many reflexive relations are there on a set with n elements?
13. Let m be a positive integer with m>1. Show that the relation
    a. $R = \{(a,b) | a \equiv b (\text{mod } m)\}$
    b. is an equivalence relation on a set of integers.
14. Draw the Hasse diagram of (D (75), divides), where the set D (75) represents the set of all divisors of 75. Check whether (D (75), divides) forms a lattice or not.
15. Find the transitive closure for the relation represented by the following matrix:

$$\begin{vmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{vmatrix}$$

16. How many cards must be selected from a standard deck of 52 cards to guarantee that at least three cards of the same suit are chosen?
17. How many bit strings of length eight either start with a 1 bit or end with two bits 00.
18. Suppose that a department contains 10 men and 15 women. How many ways are there to form a committee with six members if it must have the same number of men and women?
19. 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.
20. Let H_n denote the number of moves needed to solve the Tower of Hanoi problem with n disks. Set up a recurrence relation for the sequence H_n.

21. What is the solution of the recurrence relation

$$a_n - a_{n-1} + 2a_{n-2}$$

with $a_0 = 2$ and $a_1 = 7$?

22. Find all solutions to the recurrence relation

$$a_n = 5a_{n-1} - 6a_{n-2} + 7^n$$

23. Find all solutions to the recurrence relation

$$a_n = 3a_{n-1} + 2n$$

What is the solution with a_1=3?

24. Draw a graph from following adjacency matrix with respect to ordering of vertices a, b, c, d:

$$\begin{vmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{vmatrix}$$

25. Use Huffman coding to encode the following symbols with the frequencies listed:
    a. A: 0.08 B: 0.10 C: 0.12 D: 0.15 E: 0.20 F: 0.35
    b. What is the average number of bits used to encode the character?
26. Find a binary search tree for the words mathematics physics geography zoology meteorology geology psychology chemistry using alphabetical order.

27. Use Prim's algorithm to find a minimum spanning tree of any graph.
28. What is the chromatic number of the complete bipartite graph $K_{m,n}$, where m and n are positive integers?
29. Show that C_6 is bipartite. Also show that K_3 is not bipartite.
30. Use Dijkstra's algorithm to find the length of a shortest path between any two vertices in some weighted connected graph.

Textbook:

1. Discrete Mathematics, Seymour Lipschutz, marc Lipson, McGrawHill Publications.
2. Kenneth Rosen, Discrete Mathematics and its Applications (8e, 2019, McGraw-Hill)

References:

1. Susanna Epp, Discrete Mathematics with Applications (5e, 2019, Cengage Learning)
2. Eric Lehman, F. Thomson Leighton, Mathematics for Computer Science (1e, 2018)
3. Sarah-Marie Belcastro, Discrete mathematics (2e, 2019, CRC Press)
4. Lipschutz, Schaum's outline of Discrete Mathematics-MGH (2007)
5. GSN Murthy, UM Swami, Mathematics for JEE (Main and Advanced) (Wiley)
6. R S Aggarwal, Senior Secondary School Mathematics for Class XI (Bharati Bhawan)

Bachelor with Computer Applications as Minor (Applied Computing)

Fifth Semester

Course Code: **Course Title: Theory of Computation**
Credits: Theory (3) Practical (1) **Max. Marks: 100** **Min. Marks: 36**

Course Learning Outcomes:
Upon completion of this course, students will be able to:

1. Understand the basic concepts of complexity theory, computability theory, and automata theory.
2. Apply different types of proof techniques, such as proof by construction, proof by contradiction, and proof by induction.
3. Design and analyze finite automata and regular expressions to recognize and generate regular languages.
4. Design and analyze pushdown automata and context-free grammars to recognize and generate context-free languages.
5. Understand the Church-Turing thesis and its implications for computability.
6. Define and analyze polynomial-time and NP problems.
7. Understand the P vs NP problem and its implications for complexity theory.

Unit 1:
Introduction to Automata Theory: Automata and Languages. Regular Languages: Finite Automata (formal definition of computation, designing finite automata, the regular operations).
Deterministic and non-deterministic automata, formal definition of NFA, equivalence of NFAs and DFAs. Closure under the regular operations.
Regular Expressions: Formal definition, equivalence with finite automata.

Unit 2:
Introduction to Grammar and its hierarchy. Context Free Languages: Formal definition, examples, designing of CFGs, Ambiguity. Pushdown Automata: Formal definition, equivalence with CFGs.
Non-Context Free Languages. Deterministic Context-Free Languages: properties, Deterministic context-free grammars, relationship of DPDAs and DCFGs.
Recursive and Recursively enumerable languages.

Unit 3:
Computability Theory. The Church-Turing Thesis: Turing Machines (formal definition, examples). Introduction to Decidability, Reducibility, Recursion Theorem. Introduction to Undecidability: Halting problem.
Complexity Theory. Time Complexity: Asymptotic notations, Class P (polynomial time, examples), The class NP, P vs NP, NP-completeness. Introduction to Space complexity.

Tutorial:
Examples and problems based on Unit 1st to 3rd as specified by the course teacher.

Text Book:

1. Introduction to the Theory of Computation, Third Edition Michael Sipser, Cengage Learning, ISBN-13: 978-1-133-18779-0

Reference Book

1. Padma Reddy
2. Introduction to Automata Theory, Languages, and Computation by Jeffrey D. Hopcroft, John E. Hopcroft, and Jeffrey D. Ullman (3rd edition, 2006)
2. Formal Languages and Automata Theory by Peter Linz (3rd edition, 2011)

Bachelor with Computer Applications as Major/Minor (CT-1)

Sixth Semester

Course Code: **Course Title: Python Programming**
Credits: Theory (3) Practical (1) **Max. Marks: 100** **Min. Marks: 36**

LEARNING OUTCOMES:

After completing this course, the learner shall be able to:

1. *Understand the fundamentals of programming through Python*
2. *Transform a solution from a subjective world into an objective world.*
3. *Work with and manipulate different basic data structures available in Python*
4. *Use the List comprehensions and generators in their programs*
5. *Apply basic object-oriented concepts to design classes and objects in Python*
6. *Use concepts of Inheritance and Polymorphism in their programs*
7. *Perform basic file operations for text and CSV files*
8. *Use existing inbuilt Python modules*
9. *Develop custom modules*
10. *Use basic functionalities provided by packages Numpy and Pandas*
11. *Develop visualizations using different plotting functions available in matplotlib*

Unit-I : (15 Lectures)
Origins and History of Python, Structure of a Python Program, Interpreter shell, Indentation, Comments, Identifiers and keywords, Literals, Basic operators (Arithmetic operator, Relational operator, Logical or Boolean operator, Assignment Operator, Bit wise operator)
Entering Expressions into the Interactive Shell, The Integer, Floating-Point, and String Data Types, String Concatenation and Replication, Storing Values in Variables. Creating Python Programs: Input and Output Statements, Control statements: -branching, looping, range function, Exit function, break, continue and pass.

Unit II (15 Lectures)
Functions: Defining a Function, Calling a Function, def Statements with Parameters, Formal and Actual Arguments, Positional Arguments, Keyword Arguments, Default Arguments, Variable Length Arguments, Return Values and return Statements, The None Value, Local and Global Scope, The global Statement.
The List Data Type: Creating Lists, Basic List Operations, Indexing and Slicing in Lists, Built-In Functions Used on Lists, List Methods. List comprehension.

Unit-III (15 Lectures)
The Dictionary Data Type: Creating Dictionary, Accessing and Modifying key: value Pairs in Dictionaries, Built-In Functions used on Dictionaries, Dictionary Methods
Tuples and Sets, Creating Tuples, Basic Tuple Operations, Indexing and Slicing in Tuples, Built-In Functions Used on Tuples, Set Methods, Traversing of Sets
Strings, Creating and Storing Strings, Basic String Operations, Accessing Characters in String by Index Number, String Slicing and Joining, String Methods, Formatting Strings.
Files: Types of Files, Creating and Reading Text Data

TEXT BOOKS

1. Gowrishankar S, Veena A, "Introduction to Python Programming", 1st Edition, CRC Press/Taylor & Francis, 2018. ISBN-13: 978-0815394372
2. Downey, A.B., (2015), Think Python How to think like a Computer Scientist, 3rd edition.
3. Taneja, S. & Kumar, N., (2017), Python Programming- A Modular Approach. Pearson Education.

REFERENCE BOOKS

1. Jake VanderPlas, "Python Data Science Handbook: Essential Tools for Working with Data", 1st Edition, O'Reilly Media, 2016. ISBN-13: 978-1491912058
2. Aurelien Geron, "Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems", 2nd Edition, O'Reilly Media, 2019. ISBN – 13: 978-9352139057.
3. Wesley J Chun, "Core Python Applications Programming", 3rd Edition, Pearson Education India, 2015. ISBN-13: 978-9332555365
4. Miguel Grinberg, "Flask Web Development: Developing Web Applications with Python", 2nd Edition, O'Reilly Media, 2018. ISBN-13: 978-1491991732.
5. Brown, M. C. (2001). The Complete Reference: Python, McGraw Hill Education.
6. Dromey, R. G. (2006), How to Solve it by Computer, Pearson Education.
7. Guttag, J.V.(2016), Introduction to computation and programming using Python. MIT Press.
8. Liang, Y.D. (2013), Introduction to programming using Python. Pearson Education.

Lab Work: Python Programming (Credits=1, Lectures=30)

1. Install and configure Python IDE
2. Write simple Python program to display message on screen
3. Write a program to demonstrate different number data types & arithmetic operations in Python.
4. Write a Python program to demonstrate use of conditional statements:
5. Write Python program to demonstrate use of looping statements
6. Demonstrate the following control transfer statements in Python with suitable examples.
    i) break ii) continue iii) pass
7. Write a python program to demonstrate the working of following functions in Python.
    i) id( ) ii) type( ) iii) range( )
8. Write Python programs to demonstrate the following:
    i) input( ) ii) print( ) iii) 'sep' attribute iv) 'end' attribute
9. Write a program to create a menu driven program with the following options using functions:
    Addition 2. Subtraction 3. Multiplication 4. Division
10. Write Python program to perform following operations on Lists:
    a) Create list b) Access list c) Update list (Add item, Remove item) d) Delete list
11. Write Python program to perform following operations on Tuples:
    a) Create Tuple b) Access Tuple c) Update Tuple d) Delete Tuple
12. Write Python program to perform following operations on Sets:
    a) Create Set b) Access Set elements c) Update Set d) Delete Set
13. Write Python program to perform following operations on Dictionaries:
    a) Create Dictionary b) Access Dictionary elements c) Update Dictionary d) Delete Set e) Looping through Dictionary
14. Write a Python program to demonstrate slicing operations on lists and strings.
15. Write a Python program to perform read and write operations on a file.

Bachelor with Computer Applications as Major (CT-2)

Sixth Semester

Course Code: **Course Title: Computer Organization &Architecture**
Credits: Theory (4) Practical (2) **Max. Marks: 100** **Min. Marks: 36**

Course Learning Outcomes:

After the course the students are expected to be able to:

- Explain design of the various functional units and components of computers.
- Explain the basics of organizational and architectural issues of a digital computer and Classify and compute the performance of machines.
- Explain principles of computer organization and the basic architectural concepts.
- Demonstrate memory management and I/O techniques.
- To have better idea on how to write assemble language programs.
- Explain key aspects of Computer Organization & Architecture by enabling them to perform the experiments with support of a design and simulation.

Unit 1: Basic Computer Organisation (15 Hours)

Digital Computers, Von Neumann Computer, Basic organisation of a Computer, Computer Generations, Instruction Codes, Stored Program Organization, Computer registers, Computer instructions, Instruction cycle, Fetch and Decode, Common Bus System, Bus Interconnection Structures

Unit 2: Arithmetic Logic Unit (15 Hours)

General Register Organisation, Control Word, Stack Organisation, Instruction Formats, Addressing Modes (Direct, Indirect, Register, Indexed), Reduced Instruction Set Computer (RISC), Complex Instruction Set Computer (CISC), Instruction Pipelining,
Control Unit Implementation: Hardwired and Microprogrammed, Control Memory.

Unit 3: Memory organization (15 Hours)

Memory Hierarchy, Memory Technology, Main memory, RAM and ROM Chips, Memory Address Map, Memory Connection to CPU. RAM Technologies, SRAM, DRAM, Read Only Memory(ROM), PROM, EEPROM, Flash Memory, Auxiliary memory, Magnetic Disks, Magnetic Tape, Cache memory, Cache Memory Mappings (Direct, Associative, Set-Associative)

Unit 4: Input-Output Organization (15 Hours)

Peripheral Devices, Input-Output Interface, I/O Bus and Interface Modules, Isolated and Memory Mapped I/O, Modes of Transfer, Programmed I/O, Interrupt-Initiated I/O, Direct Memory Access (DMA), Bus Arbitration, DMA Controller, DMA Transfer, Input-Output Processor (IOP). Introduction to parallel Processing, Multi-processing systems. Multicore architecture.

Text Book:

1. M.Morris Mano-Computer System Architecture, Revised Third Edition, Pearson Education

Reference Book

1. William Stallings-Computer Organization and Architecture, Tenth Edition, Pearson Education
2. Carl Hamacher-Computer Organization, Fifth Edition, Tata McGraw Hill
3. Ramesh S Gaonkar-Microprocessor Architecture, Programming, and Applications with the 8085, Prentice Hall

<u>List of Practicals :</u>

Computer Organization and architecture lab consist of performing various experiments in GNU Sim (An open source and platform independent simulator for 8085 microprocessor).

1. Write the working of 8085 simulator GNUsim8085.
2. Write the working of 8086 simulator EMU8086.
3. Discuss basic architecture of 8085
4. Discuss basic architecture of 8086
5. Discuss the various registers available in 8085.
6. Study the complete instruction set of 8085 and write the instructions in the instruction set of 8085 along with examples.
7. Write an assembly language code in GNUsim8085 to demonstrate immediate addressing.
8. Write an assembly language code in GNUsim8085 to demonstrate indirect addressing.
9. Write an assembly language code in GNUsim8085 to demonstrate indexed addressing.
10. Write an assembly language code in GNUsim8085 to implement data transfer instruction.
11. Write an assembly language code in GNUsim8085 to store numbers in reverse order in memory location.
12. Write an assembly language code in GNUsim8085 to implement arithmetic instruction.
13. Write an assembly language code in GNUsim8085 to perform addition of two 8 bit numbers.
14. Write an assembly language code in GNUsim8085 to perform addition of two 16 bit numbers.
15. Write an assembly language code in GNUsim8085 to find 1’s & 2’s complement of a 8 bit number.
16. Write an assembly language code in GNUsim8085 to add two 8 bit numbers stored in memory and also storing the carry.
17. Write an assembly language code in GNUsim8085 to find the factorial of a number.
18. Write an assembly language code in GNUsim8085 to implement logical instructions.
19. Write an assembly language code in GNUsim8085 to implement stack and branch instructions.
20. Write a program using in GNUsim8085 for Decimal addition and subtraction of two Numbers.
21. Write a program using in GNUsim8085 for Hexadecimal addition and subtraction of two Numbers.
22. Write a program using 8085 Microprocessor for addition and subtraction of two BCD numbers.
23. Write a program to perform multiplication of two 8 bit numbers using 8085.
24. Write a program to perform division of two 8 bit numbers using 8085.
25. Write a program to find the largest and smallest number in an array of data using 8085 instruction set.
26. Write a program to arrange an array of data in ascending and descending order.
27. Write an assembly language code in EMU8086 to store numbers in reverse order in memory location.
28. Write an assembly language code in EMU8086 to implement arithmetic instruction.
29. Write an assembly language code in EMU8086 to perform addition of two 8 bit numbers.
30. Write an assembly language code in EMU8086 to perform addition of two 16 bit numbers.

Bachelor with Computer Applications as Major (CT-3)

Sixth Semester

Course Code: **Course Title: Probability and Statistics**
Credits: Theory (4) Practical (2) **Max. Marks: 100** **Min. Marks: 36**

Course Learning Outcomes:

1. Learn the language and core concepts of probability theory.
2. Understand basic principles of statistical inference.
3. Become an informed consumer of statistical information and have a good knowledge of what expectation and variance mean and be able to compute them.

Prerequisite: Fundamental Mathematics

Unit 1: (15 Lectures)

Introduction to Probability: Random experiment, sample space, trial, event. Simple probability, Compound Probability, mutually exclusive events, Addition theorem, independent events, Multiplication theorem, Dependent events, Conditional probability, Bayes' Theorem, Partitions and Total probability law. Exploring Univariate Data: Types of data, Mean, Mode and Median.

Unit 2: (15 Lectures)
Standard Deviation and Variance, Range and Finding Outliers. Counting, Random variables, probability mass function, probability density function. distributions, quantiles, mean-variance, Joint distributions, covariance, correlation, independence, and Central limit theorem.
Discrete Distributions, Random Variables, Binomial Distributions, Geometric Distributions Continuous Distributions, Density Curves, The Normal Distribution

Unit 3: (15 Lectures)
Multivariate Data, Scatter Plots, Correlation, The Least Squares Regression Line, Residuals, Non-Linear Models, Relations in Categorical Data, Margins of Error and Estimates, Confidence Interval for a Proportion, Confidence Interval for the Difference of Two Proportions, Confidence Interval for a Mean, Confidence Interval for the Difference of Two Means.

Unit 4: (15 Lectures)
Tests of Significance, Inference for the Mean of a Population, Sample Proportions, Inference for a Population Proportion, Comparing Two Means, Comparing Two Proportions, Goodness of Fit Test, and Two-way Tables.
Simple correlation (Pearson's correlation coefficient), Simple linear regression, Prediction, error in prediction, principle of least square.

Tutorials:

1. Two dice are rolled. Find the sample space for this experiment.
2. What is the probability of drawing a red card from a standard deck of 52 cards?
3. If two events are mutually exclusive, and the probability of either event A or event B occurring is 0.4, what is P(A) + P(B)?
4. If two events are independent, and the probability of event A is 0.3 and the probability of event B is 0.6, what is P(A and B)?
5. If A and B are mutually exclusive events, and P(A) = 0.2, what is P(B)?
6. Calculate the conditional probability of event A given that event B has occurred if P(A) = 0.4 and P(B) = 0.3.
7. Using Bayes' Theorem, find the probability of event A if P(B/A) = 0.6, P(A) = 0.4, and P(B)= 0.5.
8. Define a random variable and provide an example of one.
9. If X is a discrete random variable with the following probability mass function: P(X = 1) = 0.2, P(X = 2) = 0.4, and P(X = 3) = 0.4, find E(X), the expected value of X.
10. In a deck of cards, what is the probability of drawing a face card (jack, queen, or king)?
11. If you roll a fair six-sided die three times, what is the probability of getting three 6s in a row?
12. If there are 25 students in a class, and 5 of them wear glasses, what is the probability that a randomly selected student does not wear glasses?
13. If a bag contains 8 white balls and 6 black balls, what is the probability of drawing a white ball and then a black ball (without replacement)?
14. A box contains 12 chocolates, of which 3 are dark chocolate. What is the probability of randomly selecting a dark chocolate?
15. If you have a deck of cards and draw 2 cards with replacement, what is the probability that both cards are aces?
16. If you flip a coin three times, what is the probability of getting exactly two heads?
17. If a jar contains 30 red marbles and 20 blue marbles, what is the probability of drawing a red marble on the first try and a blue marble on the second try (without replacement)?
18. If you roll a fair six-sided die four times, what is the probability of getting at least one 1?
19. Imagine you are flipping a fair coin. How can you use probability distributions to represent the outcomes of this coin toss experiment?
20. Suppose you have a standard six-sided die. What is the probability mass function for this die, and what is the probability of rolling a 3?

21. You are conducting a survey where each respondent can either say "Yes" or "No" to a question. If you expect 70% of people to say "Yes," can you use the binomial distribution to calculate the probability of getting exactly 4 "Yes" responses out of 10 respondents?
22. You are modeling the outcome of a single basketball free throw attempt, where a player either makes it (success) or misses it (failure). Is this scenario best represented by a Bernoulli distribution? Why or why not?
23. Imagine you are tracking the number of customer arrivals at a small coffee shop per hour. Can you explain when and why you might use a Poisson distribution to model this situation?

24. Consider the heights of adult males in a population. Why is the normal distribution often used to describe this data, and what are the defining characteristics of the normal distribution?
25. If a student's test score has a z-score of -1.5, what does this mean in terms of their performance compared to the rest of the students? How can you use percentiles to describe their ranking within the class?
26. Suppose you have a population of 100 test scores with a mean of 75 and a standard deviation of
10. If you take random samples of 30 test scores each and calculate the sample means, what is the expected mean of the sample means, and what is the standard error of the sample means? You're conducting a survey to estimate the average income of a population.
27. From a sample of 50 individuals, you find a sample mean income of 800000 and a sample standard deviation of 70000. Calculate a 95% confidence interval for the population mean income.
28. In a chi-square goodness-of-fit test, you expect a uniform distribution of colors in a bag of marbles, but you observe the following counts: Red (25), Blue (30), Green (20), and Yellow (25). Calculate the chi-square test statistic and determine if the observed distribution significantly differs from the expected uniform distribution at a 5% significance level
29. Suppose you have a dataset of 50 test scores, and the scores are normally distributed with an unknown mean (μ) and a known standard deviation (σ) of 15. If the maximum likelihood estimation gives you a mean of 65, what is the likelihood function for this dataset?
30. In a survey of 200 people, 120 said they prefer tea over coffee. Calculate a 95% confidence interval for the proportion of people in the entire population who prefer tea.

Textbook:

1. Probability and Statistics in Engineering (4th Edition) - W. Hines, D. Montgomery, D. Goldsman, C. Borror- Wiley Publication.

2. Introduction to Probability and Statistics for Engineers and Scientists (3rd Edition) - Sheldon M. Ross, Elsevier Academic Press.

References:

1. Mood A.M. Graybill F.A and Boes D.C. (1974): Introduction to the Theory of Statistics. McGraw Hill.
2. Snedecor G.W and Cochran W.G. (1967); Statistical Methods. Lowa State University Press.
3. Cooke, Cramer and Clarke (1996): Basis Statistical Computing, Chapman and Hall. 4. David S. (1996): Elementary Probability, Oxford House.
4. Meyer P.L (1970): Introductory Probability and Statistical application, Addison Wesley.

Bachelor with Computer Applications as Minor (Applied Computing)

Sixth Semester

Course Code: **Course Title: Artificial Intelligence**
Credits: Theory (3) Practical (1) **Max. Marks: 100** **Min. Marks: 36**

Course Learning Outcomes:

1. Study the concepts of Artificial Intelligence.
2. Learn the methods of solving problems using Artificial Intelligence.
3. Learn the knowledge representation techniques, reasoning techniques and planning.

Prerequisite: Data Structures and Analysis of Algorithm

Unit 1: AI and Turing Test: (15 Lectures)
Introduction to Artificial Intelligence, Background and Applications, Turing Test, Rational Agent approaches to AI, Introduction to Intelligent Agents, structure, behaviour and environment.

Unit 2: Searching Problem Space: (15 Lectures)
Problem Characteristics, Production Systems, Informed vs Uninformed Strategies, Hill Climbing and its Variations, Heuristics Search Techniques: Best First Search, A* algorithm.
The Wumpus World Environment, Acting and reasoning in the Wumpus world, Representation, Reasoning, Logic, Prepositional Logic and First-Order Logic, Rules of Inferencing: Resolution Principle.

Unit 3: Uncertain knowledge and reasoning: (15 Lectures)
Acting under Uncertainty, handling uncertain knowledge, Uncertainty and rational decisions, Basic Probability Notation, Prior probability, Conditional probability, joint probability distribution, Bayes' Rule and Its Use, Applying Bayes' rule: The simple case, Normalization, Dampster's Shafer's Theorem.

Tutorial:

1. Given a real life situation
   - identify the characteristics of the environment
   - identify the percepts available to the agent
   - identify the actions that the agent can execute
   - suggest the performance measures to evaluate the agent
   - recommend the architecture of the desired intelligent agent
2. Undertake a comparative study on popular programming languages used in AI.
3. Translate commonly used English sentences into their equivalent logical expressions.
4. Given a problem description, formulate it in terms of a state space search problem. Analyze the given problem and identify the most suitable search strategy for the problem. Make an analysis of the properties of proposed algorithms in terms of
   - time complexity
   - space complexity
   - optimality

5. Write a Program to Implement Breadth First Search.
6. Write a Program to Implement Depth First Search.
7. Write a program to implement Hill Climbing Algorithm
8. Write a program to implement A* Algorithm.
9. Write a program to implement Tic-Tac-Toe game.
10. Write a Program to Implement 8-Puzzle problem.
11. Write a Program to Implement Water-Jug problem.
12. Write a Program to Implement Alpha-Beta Pruning using Python.
13. Write a Program to Implement 8-Queens Problem.
14. Build a Fuzzy inference system for the 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?
15. Solve 2-input 1-output project risk prediction problem using Mamdani Inference. Make necessary assumptions.

Textbook:

1. DAN.W. Patterson, Introduction to A.I and Expert Systems - PHI, 2007.

References:

1. Russell & Norvig. Artificial Intelligence-A Modern Approach, LPE, Pearson Prentice Hall, 2nd edition. 2005.
2. Ivan Bratko,"Prolog Programming for Artificial Intelligence", 3 rd Edition, Pearson Education.
3. Rich & Knight, Artificial Intelligence - Tata McGraw Hill. 2nd edition, 1991.
4. W.F. Clocksin and Mellish, Programming in PROLOG, Narosa Publishing House, 3" edition, 2001.

Bachelor with Computer Applications as Major/Minor (CT-1)

Seventh Semester

Course Code: **Course Title: Cyber Security**
Credits: Theory (3) Practical (1) **Max. Marks: 100** **Min. Marks: 36**
Learning Outcomes:

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

- Define and explain the key concepts of cybersecurity, including the CIA triad (Confidentiality, Integrity, and Availability), cyber threats and vulnerabilities, cybercrime, and cyberattack methods.
- Comprehend the importance of reporting cybercrime incidents and understand the procedures for reporting to law enforcement agencies and cybercrime reporting platforms.
- Identify the legal framework for cybersecurity, including relevant cybercrime laws, regulations, and data privacy laws. Implement data security measures to protect personal data from unauthorized access, use, or disclosure.
- Understand the security considerations for e-commerce transactions, digital payments, and social media platforms. Apply best practices to protect financial information and personal data in these domains.
- Implement cybersecurity best practices in daily life, such as creating strong passwords, protecting online accounts, avoiding suspicious links and attachments, and keeping software up to date. Understand the importance of user awareness and training in cybersecurity.

Syllabus:

THEORY (3 CREDITS, 45 hours)

UNIT-I (15 hours)

Introduction to Cybersecurity: Defining cybersecurity and its importance, The CIA triad: Confidentiality, Integrity, and Availability, Types of cyber threats and vulnerabilities, **Cybercrime and Its Types:** Understanding cybercrime and its motivations, Types of Cyber Attackers- White, grey and black hat, Common types of cybercrime: hacking, phishing, malware, ransomware, **Methods of Infiltration**: Social Engineering, Denial-of-Service, distributed DoS, Botnet, password attacks, The impact of cybercrime on individuals, organizations, and society, **Security Vulnerability & Exploits**: Hardware vulnerability, software vulnerability. **Remedial and Mitigation Measures:** Implementing security controls to prevent cyberattacks, Network security: firewalls, intrusion detection systems, access control, Endpoint security: antivirus, anti-malware, software updates, User awareness and training: educating users about cybersecurity threats and best practices.

UNIT-II (15 hours)

Reporting Cybercrime: The importance of reporting cybercrime incidents, Procedures for reporting cybercrime to law enforcement agencies, Role of cybercrime reporting platforms, Cyber Law: Legal framework for cybersecurity, Relevant cybercrime laws and regulations, Data privacy laws and their implications for cybersecurity practices. **Data Privacy and Security:** Understanding data privacy principles, Implementing data security measures: encryption, access controls, data retention policies, Protecting personal data from unauthorized access, use, or disclosure

UNIT-III (15 hours)

E-Commerce, Digital Payments and Its Security: Security considerations for e-commerce transactions, Protecting financial information and digital payments, Secure online payment methods and protocols,

Overview of Social Media and Its Security: Understanding social media platforms and their security challenges, Protecting personal information on social media, Avoiding social media scams and phishing attacks. **Cybersecurity Best Practices and Do's and Don'ts:** Implementing cybersecurity best practices in daily life, creating strong passwords and protecting online accounts, Avoiding suspicious links and attachments, Keeping software up to date and installing patches promptly.

REFERENCES:

- Raef Meeuwisse, "Cybersecurity for Beginners," Cyber Simplicity Limited, ISBN: 9781911452034, 1911452037.
- William Stallings, "Cryptography and Network Security: Principles and Practice," Pearson. Eigth edition. 2023.
- Jeetendra Pande, "Introduction to Cyber Security" Uttarakhand Open University, Year: 2017, ISBN: 9789384813963.
- Mayank Bhushan, Raj Kumar Singh, Aatif Jamshed, " Fundamental of Cyber Security (Principles, Theory & Practices," BPB Publications.
- Mark Stamp, "Information Security – Principles and Practice," Wiley Publication, Second edition, 2011.

PRACTICAL (1 CREDIT, 15 hours)

1. Discuss and define terms such as Confidentiality, Integrity, and Availability. Provide real world examples to illustrate these concepts.
2. Classify and discuss examples of cyber threats and vulnerabilities. Use case studies to analyze real-world scenarios.
3. Simulate a phishing email and discuss common signs of phishing. Teach students how to identify and avoid falling victim to social engineering.
4. Install and configure firewall, antivirus and anti-malware software on student machines. Discuss the significance of regular software updates.
5. Walk through the steps of reporting a simulated cybercrime incident on Cybercrime Portal (https://www.cybercrime.gov.in/). Discuss the role of law enforcement and reporting platforms.
6. Demonstrate the use of encryption tools and set up access controls on sample data. Discuss the importance of protecting sensitive information.
7. Simulate an e-commerce transaction, focusing on securing financial information. Discuss secure online payment methods and protocols.
8. Discuss and demonstrate security settings and features on popular social media platforms (Facebook, Twitter) that enhance security. Practice avoiding scams and phishing attacks.
9. Teach students to check if their email accounts have been compromised on the website https://haveibeenpwned.com
10. Conduct a workshop to evaluate the strength of passwords using the password strength checker at https://asecuritysite.com/encryption/passes, creating strong passwords, recognizing suspicious links, and installing software updates. Provide hands-on experience with these practices.

Bachelor with Computer Applications as Major (CT-2)

Seventh Semester

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

Learning Outcomes:

1. ***Basic Syntax:*** Understand the fundamental syntax of Java, including variables, data types, operators, and control structures.
2. ***Object-Oriented Programming:*** Comprehend the principles of object-oriented programming, such as classes, objects, inheritance, encapsulation, and polymorphism.
3. ***Exception Handling:*** Master the concept of exception handling to deal with runtime errors effectively.
4. ***File I/O:*** Understand how to read from and write to files using Java's input/output capabilities.
5. ***Multithreading:*** Gain knowledge of concurrent programming by learning how to create and manage multiple threads.
6. ***Networking:*** Understand the concept of TCP/IP server sockets and use them to create networking applications for chatting or file transfer.

Unit I: (15 Lectures)
Introduction and Background of Java, Features of Java, Java vs C++, Java Virtual Machine (JVM) and Byte-code, Java program – Structure, Compilation and Execution. Java Class libraries (System Class).main () method. Data types, Variables, Constants and Literals, Strings: Creating Strings. Operations on Strings. Basic input/Output.
Operators, Expressions and Control statements, Looping structures, 1-D and 2-D arrays.

Unit II: (15 Lectures)

Class, Object superclass, Instance, Reference, Components of Class, Variable and Method declaration, Access Specifiers, Object creation.
Constructor and its types, object cloning. Passing parameters to methods. Variable hiding.
Method overloading. Constructor overloading, Use of “this”, “volatile”, “transient”, “native” keywords, Static and non-static code blocks.

Unit III: (15 Lectures)
Inheritance, types, 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, Introduction to threads: Creating Threads in Java. Java Thread Lifecycle. Multithreading in Java, Creating and Importing Packages.

Unit IV: (15 Lectures)

Exception Handling, Exception-Object, throwing an Exception, and Exception Handler, 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. Streams: Byte, Character, Buffered, Data, and Object Streams. Standard Streams. File I/O Basics, Reading and Writing to Files.

Textbook:

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

Reference Books:

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

Java Programming (Lab Manual)

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.
2. 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.
3. Write a Java program that iteratively calculates factorial of a number.
4. Write a Java program that sorts (using bubble sort) an integer array using for loop.
5. Write a Java program that creates a Class, namely Student.
   (a) 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).
   (b). 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).
   (c). 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).
6. 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.
7. 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.
   (a).Show that all except private members are inherited.
   (b).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.
   (c). Show that an inherited method can be overridden (with the same or weaker access visibility) but canbe accessed using super keyword in the sub-class.
   (d).Show that the reference variable of type A or B can't access an overridden method of A in the Object of B.
   (e). Show that the reference variable of type A can access a shadowed data member of A in the Object of B.

8. 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).
9. 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.
10. 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.

11. Write a Java program to implement a solution for producer-consumer problem using synchronization and inter-process communication in Threads.

12. 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.
13. 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?
14. Write a Java program (client) that sends a text message to another Java program (server), which receives and displays it.

Bachelor with Computer Applications as Major (CT-3)

Seventh Semester

Course Code: **Course Title: Design & Analysis of Algorithms**
Credits: Theory (4) Practical (2) **Max. Marks: 100** **Min. Marks: 36**

COURSE OBJECTIVES:
1. Identify various Time and Space complexities of various algorithms.
2. Understand Tree Traversal method and Greedy Algorithms
3. To explain the concepts greedy method and dynamic programming. Applying for several applications like knapsack problem, job sequencing with deadlines, and optimal binary search tree, TSP and so on respectively.
4. To illustrate the methods of backtracking and branch bound techniques to solve the problems.
5. To familiarize the concepts of deterministic and non-deterministic algorithms.

UNIT - I: (15 Lectures)

Introduction to Algorithm, definition, characteristics, pseudo code for expressing algorithms, performance analysis-space complexity, time complexity, Growth of functions: asymptotic notation- big (O) notation, omega notation, theta notation, recurrences, solving recurrence relations: Substitution and master method.

UNIT - II: (15 Lectures)

Divide and Conquer: General method, binary search, min-max, quick sort, merge sort. Greedy method: General method, job sequencing with deadlines, Knapsack problem, Single source shortest path problem.

UNIT - III (15 Lectures)

Dynamic programming: General method, 0/1 knapsack problem, All pairs shortest path problem, Travelling sales person problem.Backtracking: General method, n-queen problem (4-queen), Sum of subsets problem, Travelling sales person problem

UNIT - IV (15 Lectures)

Branch and bound: General method, LC branch and bound solution, FIFO branch and bound solution. Computational Complexity, Non-deterministic algorithms, Reductions, NP-hard and NP-complete classes, Cook‘s theorem.

TEXT BOOKS: 1. Ellis Horowitz, Satraj Sahni, Rajasekharam (2007), Fundamentals of Computer Algorithms, 2nd edition, University Press, New Delhi.
REFERENCE BOOKS: 1. R. C. T. Lee, S. S. Tseng, R.C. Chang and T. Tsai (2006), Introduction to Design and Analysis of Algorithms A strategic approach, McGraw Hill, India.
2. Allen Weiss (2009), Data structures and Algorithm Analysis in C++, 2nd edition, Pearson education, New Delhi.
3. Aho, Ullman, Hopcroft (2009), Design and Analysis of algorithms, 2nd edition, Pearson education, New Delhi.

1. Implement factorial of a number program using iterative and recursive methods.
2. Write a program in C to calculate the sum of numbers from 1 to n using recursion.
3. Write a program in C to print the Fibonacci Series using recursion.
4. Write a program in C to get the largest element of an array using recursion.
5. Write a program in C to find the GCD of two numbers using recursion.
6. Write a program in C to reverse a string using recursion.
7. Write a program in C to check if a number is a prime number or not using recursion.
8. Write a program in C to find the LCM of two numbers using recursion.
9. Write a program in C to print even or odd numbers in a given range using recursion.
10. Write a C program to multiply two matrices using recursion.
11. Write a program in C to calculate the power of any number using recursion
12. Write a C program to check whether a given string is a palindrome or not using recursion.
13. Implement Linear Search.
14. Implement Binary Search using iterative and recursive methods.
15. Implement Tower of Hanoi Problem.
16. Implement BFS.
17. Implement DFS.
18. Implement Selection sort.
19. Implement Bubble Sort and analyse the time complexity.
20. Implement Merge Sort
21. Sort a given set of elements using the quick sort method.
22. Write a program to check whether a given graph is connected or not using the DFS method.
23. Implement fractional knapsack problem using Greedy Strategy.
24. Implement minimum spanning tree using Prim's algorithm and analyse its time complexity.
25. Implement minimum spanning tree using Kruskal's algorithm and analyse its time complexity.
26. Apply dynamic programming methodology to implement 0/1 Knapsack problem.
27. Apply dynamic programming methodology to find all pairs shortest path of a directed graph using Floyd's algorithm.
28. 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}. A suitable message is to be displayed if the given problem instance doesn't have a solution.
29. Implement N-Queens problem using backtracking.
30. Find the solution to the Travelling Salesman Problem. Repeat the experiment for a graph having total number of nodes (n) = 4, 8, 12, 16, 20 and note the time required to find the solution. Plot the graph taking n on the x-axis and time on y-axis and analyze the graph to determine whether it is exponential or not.

Bachelor with Computer Applications as Minor (Applied Computing)

Seventh Semester

Course Code:	**Course Title: Machine Learning**
Credits: Theory (3) Practical (1)	**Max. Marks: 100 Min. Marks: 36**

Unit I

Machine Learning, Definitions – Applications - Advantages, Disadvantages & Challenges -Types of Machine Learning- Supervised, Unsupervised, Reinforcement, Designing a learning System, Bias, Variance and Hypothesis, Hypothesis Evaluation. Dataset, features, feature vector, skewed datasets.

Unit II

Feature extraction, Feature selection, Feature Handling, Normalization, Missing data, hyper parameters, overfitting, under fitting. Dimension reduction. Discriminative and Generative Models, Linear Regression, Logistic Regression

Unit III

Classification algorithms, Decision Tree, SVM for linear data, Naïve bayes.
Clustering Algorithms: distance measures: Euclidean, Minkowski's, Manhattan and Mahalanobis, K-Means, Hierarchical Clustering (Agglomerative and Divisive clustering), Cluster Validity, Perceptron, ANN and back-propagation.

Text Books

1. Introduction to Machine learning, Ethem Alpaydin, Third Edition, MIT Press, 2009.
2. Machine learning for dummies, John Paul Muller, Luca Massaron, Weily, 2 nd Edition, 2021.

References

- Christopher M. Bishop, "Pattern Recognition and Machine Learning", Springer.
- Tom Mitchell, "Machine Learning", McGraw Hill.
- Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar, "Foundations of Machine Learning", Second Edition, MIT Press.
- Sebastain Raschka, Vahid Mirjalili , "Python Machine Learning", Packt publishing.

Lab Manual

1. **Basics of Python**

- Python Installation
- Vectorization:
  DOT product: using for loops, using numpy library dot Function
  Array: one dimensional array; multi-dimensional array; dimension and shape of array, zero, ones and empty array; number generation; indexing and slicing; Boolean indexing;
  fast element wise functions on array, array operations.
  Statistical functions
  - Set operations
  - Linear algebra.

2. **Perform the following operations using Python on any open source dataset (e.g., data.csv)**

   - Import all the required Python Libraries.
   - Locate open source data from the web (e.g., https://www.kaggle.com). Provide a clear description of the data and its source (i.e., URL of the web site).
   - Load the Dataset into pandas dataframe.
   - Data Preprocessing: check for missing values in the data using pandas isnull(), describe() function to get some initial statistics. Provide variable descriptions. Types of variables etc. Check the dimensions of the data frame.
   - Data Formatting and Data Normalization: Summarize the types of variables by checking the data types (i.e., character, numeric, integer, factor, and logical) of the variables in the data set. If variables are not in the correct data type, apply proper type conversions.
   - Turn categorical variables into quantitative variables in Python.

3. Create a Linear Regression Model using Python/R to predict home prices using Boston Housing Dataset (https://www.kaggle.com/c/boston-housing). The Boston Housing dataset contains information about various houses in Boston through different parameters. There are 506 samples and 14 feature variables in this dataset.
   The objective is to predict the value of prices of the house using the given features.
4. Implement logistic regression using Python/R to perform classification on the dataset of your choice
   - Compute Confusion matrix to find TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall on the given dataset.

5. Write a Python program to plot a few activation functions that are being used in neural networks.
6. Generate ANDNOT function using McCulloch-Pitts neural net by a python program.
7. With a suitable example demonstrate the perceptron learning law with its decision regions using python. Give the output in graphical form

8. Implement Artificial Neural Network training process in Python by using Forward Propagation, Back Propagation.
9. Write a python program to show Back Propagation Network for XOR function with Binary Input and Output

Bachelor with Computer Applications as Major/Minor (CT-1)

Eighth Semester

Course Code: **Course Title: Software Engineering**
Credits: Theory (3) Practical (1) **Max. Marks: 100** **Min. Marks: 36**

UNIT I

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.

UNIT II

Introduction to Requirements Engineering - Requirements Types: functional and nonfunctional requirements. Requirement Engineering Framework. Requirement Elicitation Process and Techniques.
Software Cost Estimation: Software Process and Project Measurement: Measures, Metrics and Indicators, Size -Oriented Metrics vs. Function - Oriented Metrics, Capability Maturity Model Integration (CMMI). COCOMO Model

UNIT III

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.
Software Testing – Basics of Software Testing, Different types of testing techniques.

Textbook:

1. Pankaj Jalote - An Integrated approach to Software Engineering, 3rd edition
2. Software Engineering: A Practitioner's Approach Seventh edition. Roger.S.Pressman

Reference Books:

1. Software Engineering: Principles and practice, 3rd Edition, Hans Van Vliet, Wiley.
2. Sommmerville - Software Engineering. Pearson, 7/e, 2006.
3. James F. Peters Software Engineering – An Engineering Approach, Wiley& Sons.

Software Engineering (Tutorials)

1. Study and compare different software process models and prepare comparison based analysis report on it. Also identify the suitable applications for the individual process model.
2. Suppose that a project was estimated to be 400 KLOC. Calculate effort & time for each of 3 modes of development. Using the following basic model equations

$$E = a(KLOC)^b$$

$$time = c(Effort)^d$$

$$Personrequired = Effort/time$$

Software Product Type	**a**	**b**	**c**	**d**
Organic	2.4	1.05	2.5	0.38
Semi-detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

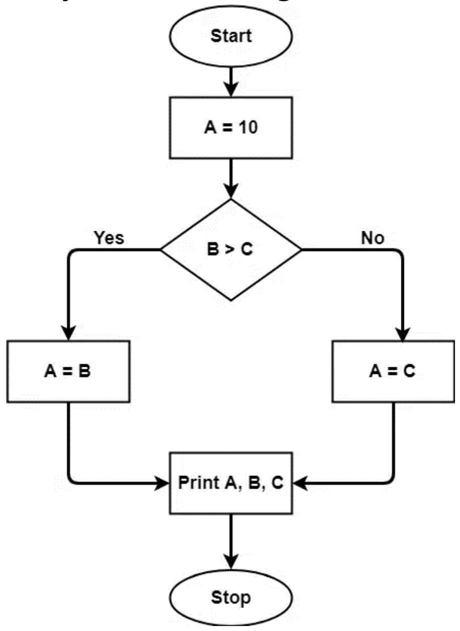
3. Calculate the effort, duration and average persons required using the above details for basic CoCoMo model. Given
   Number of user inputs=15
   Number of user outputs=14
   Number of user inquiries=8
   Number of files =6
   Number of external interfaces=13
   The weighing factors have following values (inputs=3, outputs=5, inquiries=3, files=15, interfaces=7).
   Assume that all complexity adjustment values are average and 1 function point = 25 LOC. The Values of Constant used in basic CoCoMo model. a=2.4, b=1.05, c=2.5, d=0.38.

4. Given the following values, compute function point when all complexity adjustment factor (CAF) and weighting factors are average.
   User Input = 50
   User Output = 40
   User Inquiries = 35
   User Files = 6
   External Interface = 4
5. For a Library Management System or University Admission system do the following.

   a. Identify the different requirements of the application.
   b. Identify the various elicitation techniques and their usage for the identified requirements.
   c. Classify the requirement into functional and non-functional requirements.
   d. Use various validation techniques to validate your requirement document.

6. Design the system using structured as per the instructions and applicability.
   a. Implement the design using DFD, ER diagrams and structure chart whichever applicable.
   b. Identify various processes, data store, input, output etc. of the system.
   c. Use processes at various levels to draw the DFDs.
   d. Identify various modules, input, output etc. of the system and ask students to analyze.

7. Identify the appropriate and necessary testing approach and design detailed test cases according to IEEE format for test cases.
   a. Identify various modules of the system so that they can be tested stand-alone.
   b. Identify the groups of the module that can be tested together in integration.
8. **Case Study:** You are the technical lead in your company for a new product aimed at university research scientists: an electronic lab notebook (ELN). The marketing team has given you the following information. The scientists who will use the ELN traditionally record their experiments in paper notebooks, but increasingly they are looking for digital solutions which allow them to share their information with other scientists they are working with, such as Wikis and Dropbox, which make frequent functionality changes. A small amount of market research indicates that the scientists vary considerably in what the capabilities they would like in an ELN, and the ELN solution will need to change over time as the scientists' needs change. However the scientists all say an ELN should be easy to use and quick to use. The scientists work in labs headed by a senior academic (the Lab Head). Provision of computing resources for the lab (hardware and software) varies between labs. Some have all their needs met by the central university IT department; others manage all their own hardware and software and depend on central IT only for network connections; most are somewhere in between.
   a. Which software development process is likely to be most appropriate for developing the software system?
   b. Explain why, contrasting your choice with one other software development process.
9. Discuss in detail various testing techniques.
10. Calculate the cyclometic complexity of the following control flow graph.



Bachelor with Computer Applications as Major (CT-2)

Eighth Semester

Course Code:	**Course Title: Mini Project with Dissertation**
Credits: Theory (4) Practical (2)	**Max. Marks: 100 Min. Marks: 36**

Bachelor with Computer Applications as Major (CT-3)

Eighth Semester

Course Code: **Course Title: Cryptography & Network Security**
Credits: Theory (4) Practical (2) **Max. Marks: 100** **Min. Marks: 36**

Learning Outcomes

- Understand the fundamentals of cryptography, including classical ciphers, symmetric key cryptography, and public key cryptography.
- Apply cryptographic techniques to ensure message authentication and integrity.
- Comprehend the principles and applications of digital signatures.
- Understand the fundamental principles of network security and implement network security mechanisms.
- Analyse and mitigate network security threats and employ network security tools and techniques.
- Use Cryptool 2 to implement, analyse and understand various cryptographic procedures.

Syllabus:

THEORY (4 CREDITS, 60 hours)

UNIT-I (15 hours)
Introduction to Cryptography: History of cryptography, Basics of cryptography, Types of cryptographic algorithms, Security implications of cryptography. Classical Ciphers: Caesar Cipher, Mono-alphabetic cipher, Hill cipher, Poly-alphabetic cipher (Vegnere Cipher), One time pad, Transposition Cipher (Rail-fence Cipher). Introduction to Number Theory: Prime Number Generation and Testing for Primality, Fermat's and Euler's Theorems. Symmetric Key Cryptography: Principles of symmetric key cryptography, Encryption and decryption algorithms, Claude Shannon's Theory of Diffusion and Confusion, Avalanche Effect, Feistel Cipher, Common symmetric key algorithms (e.g., DES, Triple DES, AES),

UNIT-II (15 hours)
Public Key Cryptography: Principles of public key cryptography, Key pair generation and Diffie-Hellman key exchange, Common public key algorithms (e.g., RSA)
Authentication Schemes and Hash Functions: Message Authentication, Introduction to digital signatures, Digital signature algorithms. (RSA). Introduction to hash functions, Properties of hash functions, Common hash algorithms (e.g., SHA1), Cryptographic Hash Applications: Password hashing and storage, Message Authentication Code (MAC)

UNIT-III (15 hours)
Network Security Principles and Protocols: Defining network security, understanding the CIA triad (Confidentiality, Integrity, Availability) in network security, identifying common network security threats and vulnerabilities. Introduction to network security protocols, including IPsec (Internet Protocol Security), SSL (Secure Sockets Layer). Exploring network security mechanisms such as firewalls, intrusion detection systems (IDS. Investigating the application of network security principles and protocols in various scenarios, including email security.

UNIT-IV (15 hours)
Network Security Threats and Mitigation Strategies: Understanding various types of network attacks, including denial-of-service (DoS) attacks, distributed denial-of-service (DDoS) attacks, man-in-the-middle (MitM) attacks, spoofing attacks, and malware attacks. Implementing network security mitigation strategies, including network hardening, vulnerability scanning and patching. Exploring network security tools and techniques, such as network sniffers for network traffic analysis.

REFERENCES:

- William Stallings, "Cryptography and Network Security: Principles and Practice," Pearson. Eigth edition. 2023.
- Sarhan M. Musa, "Network Security and Cryptography," Mercury Learning and Information, 2nd edition. 2022.
- Andreas Antonopoulos, "Mastering Bitcoin: Unlocking Digital Cryptocurrencies" O'reilly, Second Eduction.
- Jan L. Harrington, "Network Security: A practical Approach," Morgan Kaufmann Publishers.
- Roberta Bragg, Mark Phodes-Ousley, Keith Strassberg, "Network Security The Complete Reference," McGraw Hill Publications.

PRACTICAL (2 CREDIT)

1. Overview of Cryptool 2 and its features, Hashing (SHA1), (SHA2), (MD5)
2. Installation, and setup of Cryptool 2 environment. Download link https://www.cryptool.org/en/ct2/downloads.
3. Use Cryptool 2 to demonstrate the principles of classical ciphers such as Caesar Cipher,
4. Use Cryptool 2 to demonstrate the principles of classical ciphers such as Hill cipher
5. Use Cryptool 2 to demonstrate the principles of classical ciphers such as Vigenere Cipher.
6. Use Cryptool 2 to encrypt and decrypt messages using Feistel Cipher (FEAL).
7. Explore the processes of encryption and decryption in DES using Cryptool 2.
8. Explore the processes of encryption and decryption in AES using Cryptool 2.
9. Engage in experiments with public key algorithms (RSA) using Cryptool 2.
10. Use the "check password strength" tool in Cryptool 2 to assess and analyse the strength of a password.
11. Conduct experiments on hashing techniques (SHA-1) and the verification of hash values using Cryptool 2.
12. Conduct experiments on hashing techniques (SHA-2) and the verification of hash values using Cryptool 2
13. Setup and configure Windows Firewall for filtering. Explain its role in network security management on Windows-based systems.
14. Overview of wireshark tool,
15. Installation, and setup of Wireshark
16. Use Wireshark to capture and analyse network traffic. Identify patterns and anomalies, and discuss the role of network forensics in incident response.
17. Use Wireshark to capture and analyse network traffic on a test network. Identify different protocols, analyse packet headers, and discuss the significance of each.

Bachelor with Computer Applications as Minor (Applied Computing)

Eighth Semester

Course Code: **Course Title: Mobile Application Development**
Credits: Theory (3) Practical (1) **Max. Marks: 100** **Min. Marks: 36**

Course Learning Outcomes:

1. Understand key Android programming concepts
2. Install and configure Android Studio
3. Understand the Underlying architecture of Android Operating System
4. Create basic user interfaces for Android apps using industry standard practices.
5. Design and Develop Android apps using Kotlin/Java and the Android SDK.
6. Connect databases and server functions into Android apps
7. Practice testing and fixing issues in Android apps to make them work well.
8. Deploy Android apps on the Google Play Store

Unit 1:
Introduction to Android Operating System, Android: History, versions, basics, and tool. Overview of Android Development Framework, Android SDK Features, Android Virtual Devices. Creating AVDs. The basic structure of an Android app. Android application components: Android Manifest file. Best Practices in Android Programming. Introduction to Various Mobile Applications Development Platforms.

Unit 2:
Android Application Lifecycle, Activities, Activity Lifecycle, Activity States, Monitoring State Changes. Android user interface design, Measurement in Android UI. Fundamental Android UI Design. Introducing Views and Layouts. Types of Layout. Linear, Relative, Grid and Table Layouts Creating and Using Menus. User Interface (UI) Components: Text Views, Buttons, Radio and Toggle Buttons, Checkboxes, Spinners, Dialog and pickers. Event Handling. Handling clicks or changes of various Ul Components

Unit 3:
Fragments, Creating Fragments. Lifecycle of Fragments. Fragment States, Understand the different states fragments can exist in during their lifecycle. Adding Fragments to Activity. Managing Fragments with Transactions. Intents and Broadcasts, Intent Basics, Using Intents to Launch Activities, Implicit Intents Passing Data to Intents.

Textbook:

1. Android Programming for Beginners - Second Edition by John Horton
2. Head First Android Development: A Brain-Friendly Guide by Dawn Griffiths (Author), David Griffiths (Author)
3. Professional Android Application Development by Reto Meier, Wiley India.
4. Android Application Development with Kotlin Hardik Trivedi, BPB Publications

Online References:

1. https://www.udemy.com/course/become-an-android-developer-from-scratch/
2. https://developer.android.com/courses/android-basics-kotlin/course
3. https://www.coursera.org/specializations/android-app-development
4. Explore Google Code Labs : https://codelabs.developers.google.com/?product=android

Tutorials

Bachelor with Computer Applications as Major/Minor (CT-1)

Eighth Semester

Course Code: **Course Title: Research Methodology**
Credits: Theory (3) Practical (1) **Max. Marks: 100** **Min. Marks: 36**

Learning Outcomes:

1. To understand and comprehend the basics in research methodology and applying them in research/ project work.
2. Identify the research gaps by reviewing the existing literatures on the concerned topic.
3. To select an appropriate research design.
4. To collect data and develop skills in qualitative and quantitative data analysis and presentation.
5. Ability to choose methods appropriate to research objectives.

UNIT I

Introduction to Research Methodology: Meaning of Research, Objectives of Research, Motivations in Research, Types of Research: Pure and Applied Research, Qualitative and Quantitative. Research Approaches, Significance of Research, Research Methods v/s Methodology, Fundamentals of Research and Scientific Methods, Research Process, Criteria of Good Research.

UNIT II

Defining the Research Problem: Identification of Research problem. Collecting and reviewing the literature, conceptualization and Formulation of a research problem. Identifying variables. Constructing hypothesis. Writing a Research Proposal.

Research Design: Need for research design, Features of a good research design, Different research designs (exploratory, descriptive experimental and diagnostic research). Sampling: Meaning and types of sampling; Probability and Non-Probability.

UNIT III

Data Collection and analysis: Collection of Primary and Secondary Data via questionnaire and Schedules, other Observation Interview Methods, Case Study, Focus Group Discussion. Reliability and Validity of Measurement Scales. Exploratory data analysis, parametric and nonparametric tests, correlation and regression analysis, Analysis of Variance (ANOVA). Writing Research Report, Format of Report.

Text Books:

1. Ranjit Kumar., "Research Methodology, A step by step guide for beginners", Sixth Edition, 2009, Pearson Education.
2. Kothari C.R., "Research Methodology, Methods and Techniques, Second edition, 2008, New Age International Publication.
3. Krishna Swamy K.N., Siva Kumar A.I., Mathirajan M., "Management Research Methodology 2006, Pearson Education, New Delhi.
4. Cooper D., Schindler P., Business research methods", 2003, Tata Mc-Graw Hill, New Delhi.

Reference Books:

1. Alan Bryman., "Social Research Methods", 2018, London.
2. Robert O. Kuehl Brooks/cole., "Design of Experience: Statistical Principles of Research Design and Analysis".
3. Alan Bryman & Emma Bell., "Business Research Methods", Oxford University Press.

TUTORIAL / LAB

Installing and Configuring Android Studio

1. Discuss the various steps in conducting a research study.
2. Describe the contents of research proposal.
3. Illustrate various levels of measurements popularly used in research with their properties.
4. Distinguish between random and systematic errors.
5. What are various sources of error in measurements?
6. State the characteristics of a good Questionnaire.
7. What points should be taken into consideration by a researcher in developing a sample design for his/her research project?
8. Role of Computers in Research.
9. Maintenance of data using software such as Mendeley, Endnote.
10. Tabulation and graphical presentation of research data and software tools.
11. Creating, Saving, Opening, Printing and Formatting a Document in LaTex/MS Office.
12. Use of Encyclopedias, Research Guides, Handbook etc.
13. Professional Written Communication: Students prepare E-mails, Letters, memos, proposals, formal and informal reports.
14. Oral Presentations using usual aids such as handouts and presentation software such as PowerPoint.
15. Software for detection of Plagiarism.

Bachelor with Computer Applications as Major/Minor (CT-1)

Eighth Semester (Honours with Research)

Course Code:	**Course Title: Research Project**
Credits: 12 Credits	**Max. Marks: 300 Min. Marks: 36**

Bachelor with Computer Applications as Major/Minor (CT-1)

Eighth Semester (Honours with Research)

Course Code:	**Course Title: Latex**
Credits: Theory (3) Practical (1)	**Max. Marks: 100 Min. Marks: 36**

Course Learning Outcome

This course aims to equip students with essential skills in using LaTeX for document preparation. This includes understanding LaTeX syntax, creating various document types, formatting equations, customizing styles, integrating media, troubleshooting errors, and fostering efficiency and collaboration in producing professional/ Scientific documents

Unit-1: Overview of LaTeX

Overview of LaTeX: Introduction to LaTeX as scientific writing
Installation and setup of LaTeX distribution, Installation of Software -TeXStudio/MikeTex in Linux/Windows)
Basics for document structuring, preamble preparation, typesetting a simple document, Adding basic information to a document, Environments, Footnotes, Sectioning and displayed material.
Basics of LaTeX syntax: Document structure, commands, and environments

Unit 2: Document Formatting

Writing simple documents: Creating text, sections, lists, and basic formatting
Customizing document layout: Page styles, margins, headers, and footers
Working with mathematical equations: Math mode, symbols, and equations
Handling figures and tables: Inserting figures, captions, Creating Tables, enumeration list, itemized list, font effects.
Reference and Citation Inserting references, Manual reference, Reference using BibTex, citing reference

Unit 3: Advanced Topics in LaTeX

Creating Professional Scientific Documents:
Managing large documents: Using chapters, sections, and managing content
Advanced mathematical typesetting: Matrices, aligning equations, and theorem environments.
Customizing styles: Packages, templates, and custom commands.
Unit 4: Collaborative Writing and Final Project

Graphics in LaTeX, Simple pictures using PS Tricks, Plotting of functions, Beamer presentation.
Collaborative writing with LaTeX: Using version control systems (e.g., Git) and online platforms (e.g., Overleaf)

Tutorial

1. Create a new LaTeX document using the 'article' class. Add sections (at least 3) with appropriate titles.
2. Include a title, author, and date in the document.
3. Apply formatting (e.g., bold, italics) to specific text within the document.
4. Write an equation for a quadratic formula (inline and displayed).
5. Create a matrix with specific elements.
6. Use mathematical symbols and operators to create complex equations.
7. Create a table showcasing student grades (at least 5 rows and 3 columns).
8. Insert an image into the document with a caption.
9. Insert a PNG or JPEG image into a LaTeX document using the \includegraphics command. Experiment with resizing the image, changing its position, and exploring different alignment options.
10. Use the graphicx package to create a figure environment and add a caption to the image. Practice referencing the figure within the text using \ref and \label commands.
11. Add citations within the document text.
12. Create a bibliography with at least 3 references.
13. Customize headers and footers with specific content.
14. Adjust margins and font styles in the document.
15. Share and collaborate on LaTeX documents using overleaf
16. Draft a CV using LaTeX templates