SYLLABUS FOR COMPUTER SCIENCE (MINOR)

Under Three Discipline Specific Course (FYUGP)

(To be implemented from Session 2024-25)

SEM. I & II

Proposed Syllabus for Computer Science (Minor) Programme							
Year	**Semester**	**Paper Code**	**Paper**	**Credits**	**Periods/Week**	**Exam. Marks**	**Total Marks**
1st Year	**I**	**COMSMIN101**	**Computer Fundamentals**	**3**	**3**	**60**	**80**
		COMSMIN101T	**Computer Fundamentals (Tutorial)**	**1**	**1**	**20**	
2nd Year	**II**	**COMSMIN102**	**Programming in C**	**3**	**3**	**60**	**80**
		COMSMIN10	**Programming in C (Lab)**	**1**	**1**	**20**	

NOTE:

Tutorials should involve problem solving session/activity related to the subject taught.

1st Year Semester-I			
Course-DSC Paper:	**Paper Code- COMSMIN101** **Computer Fundamentals**	**Credits-3**	**Lectures/Week-3**

Prerequisite(s) and/or Note(s):

(1) High school Physics.

(2) Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Course Objectives:

Knowledge acquired:

(1) Introduce students to the fundamental concepts of system tools, their functionalities, and their role in computer systems.

(2) Familiarize students with various peripheral devices commonly used in computer systems and their functionalities.

(3) Develop students' proficiency in utilizing system tools to optimize computer performance, troubleshoot issues, and maintain system integrity.

(4) Enable students to effectively manage peripheral devices, including printers, scanners, external storage devices, and input/output devices.

Skills gained:

(1) Basic idea programming.

(2) Troubleshooting hardware and software issues.

(3) Efficient internet and file management.

Competency Developed:

(1) Problem-solving in technical contexts.

(2) Adaptability to various software environments.

Effective communication of technical concepts.

Syllabus Overview

Unit 1:	**Basics of Computer**	**15 Lectures**

Generation of Computers; Computer system : Basic Block Diagram, Super Mainframe, Mini & Personal Computer, Nomenclature, Software : Systems and Application; Hardware & Software; Algorithms : Definition, essential features;

Unit 2:	**Peripheral Devices**	**15 Lectures**

Input and Output Devices – Punched Card, Keyboard, Mouse, Joystick, Trackball, Light Pen, Touch Screen, Magnetic Ink Character Recognition (MICR), Optical Character Recognition (OCR), Optical Mark Recognition (OMR), Display units, Printers- Impact and Non-Impact. Primary storage – RAM-SRAM, DRAM, ROM-PROM, EPROM, EEPROM, Secondary storage – Hard drive, Magnetic drive, Compact Disk, Cache memory, components of motherboard.

Unit 3:	**Programming Fundamentals**	**10 Lectures**

Complexity: notation, time & space; Computability & correctness concepts; Structured programming concepts; Process of problem solving, Flowcharts and Pseudo codes.

Unit 4:	Software and Data	5 Lectures

Concept of software, Firmware, Types of Software: (System software, Application software, Utility software), Concept of Operating System, Loaders, Linkers, Debuggers, Translators: (Compilers, Assemblers, Interpreters), Data and Information, Types of data, Units of data measurement, Qualities of Information, Concept of database, Languages, Generation of Languages.

Suggested Readings

1. Sinha P.K., "Computer Fundamentals", 6 th Edition, BPB Publication, 2012.
2. Rajaraman,V.,"Computer Fundamentals", 6 th Edition, PHI,2012.
3. Thareja R., "Fundamentals of Computers", Oxford University Press, 2014.
4. Stallings W., "Operating systems", 8th Edition, Pearson, 2014.

Course-DSC	Paper Code- COMSMIN101T	Credits-1	Tut./Week-1
Paper:	Computer Fundamentals (Tutorial)		

Computer Fundamentals Tutorial as assigned and advised by teacher(s).

1st Year
Semester-II

Course-MINOR Paper Code- COMSMIN202 Credits-3 Lectures/Week-3
Paper: Programming Fundamentals Using C

Prerequisite(s) and/or Note(s):
(1) High school mathematics.
(2) Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Course Objectives:

Knowledge acquired:
(1) Knowledge about program development and implementation
(2) Syntax of C programming language
(3) Knowledge about how humans interact with computers through a language.

Skills gained:
(1) Problem solving skills
(2) Logical thinking to approach a problem
(3) Building programs for different problems at hand.

Competency Developed:
(1) Applying the skills learnt to model real world problems
(2) Facility in solving real life problems by thinking logically and outside of box.
(3) Ease of switching to any other programming language

Syllabus Overview

Unit 1: Introduction to C programming 10 Lectures

History of C, Overview of Procedural Programming, Introduction to Algorithm & Flowcharts. Using main() function Compiling and Executing Simple Programs in C. Declaring, Defining and Initializing Variables, Scope of Variables, Using Named Constants, Keywords, Data Types, Casting of Data Types, Operators (Arithmetic, Logical and Bitwise), Using Comments in programs, Character I/O (getc, getchar, putc, putchar etc), Formatted and Console I/O (printf(), scanf()) , Using Basic Header Files (stdio.h, conio.h, stdlib.h etc).

Unit 2: Expression and Control Flow 10 Lectures

Simple Expressions in C (including Unary Operator Expressions, Binary Operator Expressions), Understanding Operators Precedence in Expressions, Conditional Statements (if construct, switch-case construct), Understanding syntax and utility of Iterative Statements (while, do-while, and for loops), Use of break and continue in Loops, Using Nested Statements (Conditional as well as Iterative)

Unit 3: Functions 7 Lectures

Utility of functions, Call by Value, Call by Reference, Functions returning value, Void functions, Return type of functions, Functions parameters, Differentiating between Declaration and Definition of Functions, Functions with variable number of Arguments.

Unit 4: Arrays and Strings in C 10 Lectures

Creating and Using One Dimensional Arrays (Declaring and Defining an Array, Initializing an Array, Accessing individual elements in an Array, Manipulating array elements using loops), Use Various types of arrays (integer, float and character arrays / Strings) Two-dimensional Arrays (Declaring, Defining and Initializing Two Dimensional Array, Working with Rows and Columns).

Unit 5: User-defined Data Types and Pointers Basics 8 Lectures

Understanding utility of structures and unions, Declaring, initializing and using simple structures and unions, Manipulating individual members of structures and unions, Array of Structures, Pointers, integer, float and string pointers, working of pointers.

Suggested Readings

1. Rajaraman V. & Radhakrishnan, An Introduction To Digital Computer Design, PHI.
2. Malvino & Leach, Digital Principles & Applications, TMH
3. S. Salivahanan, S. Arivazhagan, Digital Circuits and Design, Oxford University Press

Course-MINOR Paper Code- COMSMIN202L Credits-1 Lab hours/Week-2
Paper: Programming Fundamentals Using C (Lab)

Students are advised to do laboratory/practical practice not limited to, but including the following types of problems:

1. WAP to perform input/output of all basic data types.
2. WAP to enter two numbers and find their sum.
3. WAP to reverse a number.
4. WAP to Swap Two Numbers (using and without using a third variable).
5. WAP to check whether a number is even or odd
6. WAP to compute the factors of a given number.
7. WAP to enter marks of five subjects and calculate total, average and percentage.
8. WAP to print the sum and product of digits of an integer.
9. WAP to check whether a character is vowel or consonant
10. WAP to find the largest among three numbers

SYLLABUS FOR COMPUTER SCIENCE (MINOR)

Under Three Discipline Specific Course (FYUGP)

(To be implemented from Session 2024-25)

SEM. III & IV

Proposed Syllabus for Computer Science (Minor) Programme							
Year	**Semester**	**Paper Code**	**Paper**	**Credits**	**Periods/Week**	**Exam. Marks**	**Total Marks**
2nd Year	**III**	**COMSMIN303**	**Basic Digital Electronics**	**3**	**3**	**60**	**80**
		COMSMIN303L	**Basic Digital Electronics (Lab)**	**1**	**1**	**20**	
	IV	**COMSMIN404**	**Programming in Java**	**3**	**3**	**60**	**80**
		COMSMIN404T	**Programming in Java (Lab)**	**1**	**1**	**20**	

NOTE:

Tutorials should involve problem solving session/activity related to the subject taught.

SEMESTER - III

Course-MINOR Paper:	Paper Code- COMSMIN303 Basic Digital Electronics	Credits-3	Lectures/Week-3

Prerequisite(s) and/or Note(s):

1. High school Physics.
2. Note(s): Syllabus changes yearly and may be modified during the term itself, depending on the circumstances. However, students will be evaluated only on the basis of topics covered in the course.

Course Objectives:

Knowledge acquired:

1. Introduce students to the fundamental concepts of system tools, their functionalities, and their role in computer systems.
2. Familiarize students with various peripheral devices commonly used in computer systems and their functionalities.
3. Develop students' proficiency in utilizing system tools to optimize computer performance, troubleshoot issues, and maintain system integrity.
4. Enable students to effectively manage peripheral devices, including printers, scanners, external storage devices, and input/output devices.

Skills gained:

1. Basic idea programming.
2. Troubleshooting hardware and software issues.
3. Efficient internet and file management.

Competency Developed:

1. Problem-solving in technical contexts.
2. Adaptability to various software environments.
3. Effective communication of technical concepts.

Syllabus Overview

Unit 1:	Number Systems and Codes	10 Lectures

Binary, octal, hexadecimal and decimal number systems and their inter conversion, BCD numbers (8421-2421), Gray code, excess–3 code, code conversion, ASCII, EBCDIC codes, their advantages and disadvantage, Binary addition and subtraction, Negative number representation: Sign magnitude, 1's, 2's Complement. signed and unsigned binary numbers, Fixed and floating-point representation.

Unit 2:	Logic Gates	13 Lectures

AND, OR, NOT Gates and their Truth Tables, NOR, NAND & XOR gates, Boolean algebra, Basic Boolean Laws, De-morgan’s theorem, Boolean function and their truth tables, Minimization techniques, K-Map for 2, 3 and 4 variables, Sum of Product & Product of Sum, Don’t care conditions

Unit 3:	Combinational Logic	11 Lectures

Half adder, Full adder, parallel adder, half subtractor, full subtractor, 4-bit binary adder cum subtractor, Multiplexer, Demultiplexer, Decoder, BCD to seven segment Decoder, Encoders**.**

Unit 4:	Sequential Logic	11 Lectures

Set-reset latches, D-flip-flop, R-S flip-flop, J-K flip-flop, Master slave flip-flop, T flip-flop, Synchronous/Asynchronous counter, Applications of counter, Registers, Applications of registers.

Suggested Readings

1. Sinha P.K., "Computer Fundamentals", 6 th Edition, BPB Publication, 2012.
2. Rajaraman,V.,"Computer Fundamentals", 6 th Edition, PHI,2012.
3. Thareja R., "Fundamentals of Computers", Oxford University Press, 2014.
4. Stallings W., "Operating systems", 8th Edition, Pearson, 2014.

Course-MINOR Paper:	Paper Code- COMSMIN303L Basic Digital Electronics (Lab)	Credits-1	Lab Hours/Week-2

Students are advised to do laboratory/practical practice not limited to, but including the following types of problems:

1. Implementation of all the basic gates.
2. Design all gates using NAND gates.
3. Design all gates using NOR gates.
4. Design of a XOR gate using basic gates.
5. Design of an 8x1 MUX using basic gates.
6. Design of an 1x8 DEMUX using basic gates.
7. Design of a Decoder (different variants) using basic gates.
8. Design of an Encoder (different variants) using basic gates.
9. Simple Boolean Expression using Basic gates and Universal gates: (Examples)

   $A \cdot (\overline{B}+A) + B.\overline{A}$

   $XZ + X'YZ$

   $A + B [AC + (B + C') D]$

10. Design Half-Adder, Full-Adder, Half-Subtractor, Full-Subtractor Circuit

SEMESTER – IV

Course-MAJOR Paper:	Paper Code-COMSMIN404 Programming in Java	Credits-3	Lectures/Week-3

Prerequisite(s) and/or Note(s):

1. Familiarity with basic programming.
2. **Note(s):**
   - Course content may be revised during the term to align with academic priorities.
   - Students will be evaluated only on the topics covered in class.

Course Objectives:

Knowledge Acquired:

1. Write Java programs effectively.
2. Understand and apply object-oriented programming (OOP) principles.
3. Design and implement simple graphical user interfaces (GUIs).
4. Work with files and databases using Java.
5. Gain a foundational understanding of Java web development concepts.

Skills Gained:

1. Understanding and applying Object-Oriented Programming (OOP) principles
2. Implementing and using data structures
3. Handling exceptions effectively
4. Working with multithreading for concurrent programming
5. Engaging in application development (desktop, web, and mobile)
6. Implementing networking features and protocols

Competency Developed:

1. Proficiency in Object-Oriented Programming (OOP)
2. Implementation of data structures using Java
3. Design and development of Graphical User Interfaces (GUIs)
4. Familiarity with version control systems

Syllabus Overview

Unit 1: Introduction to Java **10 Lectures**

Java Architecture and Features, Understanding the semantic and syntax differences between C and Java, Compiling and Executing a Java Program, Variables, Constants, Keywords, Data Types, Operators (Arithmetic, Logical and Bitwise) and Expressions, Comments, Basic Program Output, Decision Making Constructs (conditional statements and loops) and Nesting, Java Methods (Defining, Scope, Passing and Returning Arguments, Type Conversion and Checking, Built-in Java Class Methods).

Unit 2: Arrays, Strings, and I/O **12 Lectures**

Creating & Using Arrays (One Dimension and Multi-dimensional), Referencing Arrays Dynamically, Java Strings: The Java String class, Creating & Using String Objects, Manipulating Strings, String Immutability& Equality, Passing Strings To & From Methods, String Buffer Classes. Simple I/O using System.out and the Scanner class, Byte and Character streams, Reading/Writing from console and files.

Unit 3: Object-Oriented Programming Overview **10 Lectures**

Principles of Object-Oriented Programming, Defining & Using Classes, Controlling Access to Class Members, Class Constructors, Method Overloading, Class Variables & Methods, Objects as parameters, final classes, Object class, Garbage Collection.

Unit 4: Inheritance, Interfaces, Packages, Enumerations, Autoboxing & Metadata **13 Lectures**

Inheritance:(Single Level and Multilevel, Method Overriding, Dynamic Method Dispatch, Abstract Classes), Interfaces and Packages, extending interfaces and packages, Package and Class Visibility, Using Standard Java Packages (util, lang, io, net), Wrapper Classes, Autoboxing /Unboxing, Enumerations and Metadata.

Suggested Readings

1. Ken Arnold, James Gosling, David Homes, "The Java Programming Language", 4thEdition,2005.
2. James Gosling, Bill Joy, GuyL Steele Jr, Gilad Bracha, Alex Buckley "The Java LanguageSpecification, Java SE 8 Edition (Java Series)", Published by Addison Wesley, 2014.
3. Joshua Bloch, "Effective Java" 2nd Edition, Publisher: Addison-Wesley,2008.
4. Cay S. Horstmann, Gary Cornell, "Core Java 2 Volume1, 9thEdition, PrinticeHall.2012
5. Cay S. Horstmann, Gary Cornell, "Core Java 2 Volume 2 - Advanced Features", 9th Edition,PrinticeHall.2013
6. Bruce Eckel, Thinking in Java", 3rd Edition, PHI,2002.
7. E. Balaguruswamy,"Programming with Java", 4th Edition, McGrawHill.2009.
8. Paul Deitel, Harvey Deitel, "Java: How to Program", 10th Edition, Prentice Hall,2011.
9. Bert Bates, Kathy Sierra, "Head First Java", Orielly Media Inc. 2nd Edition,2005. 10.Object Oriented Programming through JAVA, Pradha Krishna, University Press 11.David J. Eck, "Introduction to Programming Using Java", Published by Create Space Independent Publishing Platform, 2009.
10. John R. Hubbard, "Programming with JAVA", Schaum's Series, 2ndEdition, 2004.

Course-MINOR Paper: **Paper Code- COMSMIN404T Programming in Java (Lab)** **Credits-1** **Tutorial hours/Week-2**

Students are advised to do laboratory/practical practice not limited to, but including the following types ofproblems:

1. Java Program to Get Input from the User
2. Java Program to Swap Two Numbers
3. Java Program to Find the Factorial of a Number
4. Java Program to Check if a number is even or odd

5. Java Program to Find Largest Among 3 Numbers
6. Java Program to Find LCM and HCF of 2 numbers
7. Java Program to Display All Prime Numbers from 1 to N
8. Java Program to Check Leap Year
9. Java Program to Check Armstrong Number between Two Integers
10. Java Program to Check whether input character is vowel or consonant
11. Java Program to Find Factorial of a number
12. Java Program to Display Fibonacci Series
13. Java Program to Calculate Simple Interest
14. Java Program to Print Pattern
15. Java Program for Conversions
16. Java Program to Create a Class and Object
17. Java Program to Create Abstract Class
18. Java Program to Show Inheritance in Class
19. Java Program to Show Overloading of Methods in Class
20. Java Program to Show Overriding of Methods in Classes
21. Java Program to Show Use of Super Keyword in Class
22. Java Program to Show Use of This Keyword in Class
23. Java Program to Show Usage of Static keyword in Class
24. Java Program to Show Use of Static and Non-static Methods
25. Java Program to Show Usage of toString() Method
26. Java Program to Show Usage of compare() Method
27. Java Program to Show Usage of equals() Method
28. Java Program to implement a thread.
29. Java Program to Search an Element in an Array
30. Java Program to Find the Largest and smallest Element in an Array
31. Java Program to Sort an Array
32. Java Program to Merge Two Arrays
33. Java Program to Add, subtract, multiply Two Matrices
34. Java Program to Check Whether Two Matrices Are Equal or Not
35. Java Program to Find the Transpose
36. Java Program to Compute the Sum of Diagonals of a Matrix
37. Java Program to find the sum of upper and lower triangular
38. Java Program to Get a Character From the Given String
39. Java Program to Replace a Character at a Specific Index
40. Java Program to Reverse a String
41. Java Program to Compare two strings
42. Java Program to implement other string functions
43. Java Program to find the sum of any number of integers entered as command line arguments
    Java Program to find the sum of any number of integers interactively, i.e., entering every number from the keyboard, where as the total number of integers is given as a command line argument

SYLLABUS FOR COMPUTER SCIENCE (MINOR)

Under Three Discipline Specific Courses (FYUGP)

(To be implemented from Session 2024-25)

SEM. V, VI, VII & VIII

Proposed Syllabus for Computer Science (Minor)Programme							
Year	**Semester**	**Paper Code**	**Paper**	**Credits**	**Periods/Week**	**Exam. Marks**	**Total Marks**
3rd Year	**V**	**COMSMIN505**	**Computer Networks**	**3**	**3**	**60**	**80**
		COMSMIN505T	**Computer Network (Tutorial)**	**1**	**1**	**20**	
	VI	**COMSMIN606**	**Python Programming**	**3**	**3**	**60**	80
		COMSMIN606 L	**Python Programming (Lab)**	**1**	**1**	**20**	
4th Year	**VII**	**COMSMIN707**	**R Programming**	**3**	**3**	**60**	**80**
		COMSMIN70L	**R Programming (Lab)**	**1**	**1**	**20**	
	VIII	**COMSMIN808**	**Artificial Intelligence**	**3**	**3**	**60**	80
		COMSMIN808T	**Artificial Intelligence(Tutorial)**	**1**	**1**	**20**	

NOTE:

Tutorials should involve problem solving session/activity related to the subject taught.

SEMESTER-V

Course-MINOR Paper:	**Paper Code-COMSMIN505 Computer Network**	**Credits-3**	**Lectures/Week-3**

Computer Networks

Brief Course Description

This course introduces the fundamental concepts of computer networks, including network architectures, communication protocols, transmission media, network devices, routing, switching, and Internet technologies. It provides students with an understanding of how data is transmitted, managed, and secured across interconnected systems and networks.

Prerequisite(s) and/or Note(s)

Students should have basic knowledge of computer systems and operating systems. Familiarity with fundamental computer concepts is desirable. Practical sessions involving network configuration, simulation tools, and troubleshooting exercises may accompany the theoretical components of the course.

Course Objective

The course aims to provide students with a comprehensive understanding of computer networking principles, network models, communication protocols, and network services. It enables learners to analyze network operations, configure network devices, and understand modern networking technologies and security mechanisms.

Course Outcome

Upon successful completion of this course, students will be able to explain networking concepts and protocols, analyze network architectures, configure basic network components, troubleshoot network-related issues, and apply networking principles to design and manage communication systems effectively.

Knowledge Acquired

Students will acquire knowledge of network fundamentals, OSI and TCP/IP models, transmission media, network topologies, switching and routing concepts, IP addressing, wireless networks, network security, and Internet-based communication technologies.

Skills Gained

Students will develop skills in network configuration, IP addressing, subnetting, protocol analysis, network troubleshooting, simulation using networking tools, and implementing basic network security measures in various networking environments.

Competency Developed

Upon completion of the course, students will be competent in designing, configuring, managing, and maintaining computer networks. They will be able to apply networking concepts to real-world scenarios, ensure efficient data communication, and support organizational networking and connectivity requirements.

Detailed Syllabus
3rd Year: Semester 5

COMSMIN505	COMPTER NETWORKS	[Credits: 3, Lectures 45]

Unit 1: Introduction to Computer Networks **(6 Lectures)**
Network definition; network topologies; network classifications; network protocol; layered network architecture; overview of OSI reference model; overview of TCP/IP protocol suite.

Unit 2: Data Communication Fundamentals and Techniques **(6 Lectures)**
Analog and digital signal; data-rate limits; digital to digital line encoding schemes; pulse code modulation; parallel and serial transmission; digital to analog modulation-; multiplexing techniques- FDM, TDM; transmission media.

Unit 3: Networks Switching Techniques and Access mechanisms **(6 Lectures)**
Circuit switching; packet switching- connectionless datagram switching, connection-oriented virtual circuit switching; dial-up modems; digital subscriber line; cable TV for data transfer.

Unit 4: Data Link Layer Functions and Protocol **(7 Lectures)**
Error detection and error correction techniques; data-link control- framing and flow control; error recovery protocols-stop and wait ARQ, go-back-n ARQ; Point to Point Protocol on Internet.

Unit 5: Multiple Access Protocol and Networks **(5 Lectures)**
CSMA/CD protocols; Ethernet LANS; connecting LAN and back-bone networks- repeaters, hubs, switches, bridges, router and gateways;

Unit 6: Networks Layer Functions and Protocols **(5 Lectures)**
Routing; routing algorithms; network layer protocol of Internet- IP protocol, Internet control protocols

Unit 7: Transport Layer Functions and Protocols **(5 Lectures)**
Transport services- error and flow control, Connection establishment and release-three way handshaking

Unit 8: Overview of Application layer protocol **(5 Lectures)**
Overview of DNS protocol; overview of WWW & HTTP protocol

Suggested Readings

1. B. A. Forouzan: Data Communications and Networking, Fourth edition, THM, 2007.

2. A. S. Tanenbaum: Computer Networks**,** Fourth edition**,** PHI , 2002

3. Dr. Rakesh Kumar Mandal: Computer Networks for Students, First Edition, SPD, 2018

COMSMIN505T	COMPTER NETWORKS	[Credits: 1, Lectures 15]

Computer Networks tutorial as assigned and advised by teacher(s).

SEMESTER-VI

Course-MINOR Paper:	**PaperCode-COMSMIN606 Python Programming**	**Credits-3**	**Lectures/Week-3**

Course Name: PYTHON PROGRAMMING

Brief Course Description
This course introduces the fundamentals of Python programming, covering basic syntax, data types, control structures, functions, and data structures. Students learn to develop, test, and debug Python programs while building problem-solving and logical thinking skills. The course provides a strong foundation for advanced studies in software development, data science, artificial intelligence, and other computing-related fields.

Course Objectives
This course aims to introduce students to the fundamentals of Python programming and develop their ability to design, write, test, and debug programs. It focuses on building a strong foundation in programming concepts, problem-solving techniques, data structures, and algorithmic thinking while preparing students for advanced studies and applications in computing and software development.

Prerequisite(s) and/or Note(s)

A basic understanding of high school mathematics is recommended for this course. The syllabus may be revised or updated during the academic term based on institutional or academic requirements; however, students will be assessed only on the topics covered during the course.

Knowledge Acquired

Students will gain a solid understanding of Python programming fundamentals, including variables, data types, operators, control structures, functions, and essential data structures such as lists, tuples, dictionaries, and sets. They will learn how these concepts are used to develop efficient and reliable programs.

Skills Gained

Through practical exercises, assignments, and programming projects, students will develop proficiency in writing, testing, and debugging Python code. They will strengthen their problem-solving abilities, learn to implement algorithms effectively, and acquire the skills needed to troubleshoot and resolve programming errors.

Competency Developed

The course will enhance students' logical and analytical thinking, attention to detail, and ability to design structured solutions to computational problems. Students will also develop good coding practices, documentation skills, and the ability to collaborate effectively on software development projects.

Detailed Syllabus
3rd Year: Semester 6
COMSMIN606 **PYTHON PROGRAMMING** **[Credits: 3, Lectures 45]**

Unit 1: Introduction to Programming **(6 Lectures)**

Concept of problem solving, Problem definition, Program design, Debugging, Types of errors in programming, Documentation. Flowcharts, algorithms, Types of programming paradigms – unstructured, procedure oriented, object oriented. Programming methodologies - top-down and bottom-up programming.

Unit 2: Introduction to Python **(12 Lectures)**

Structure of a Python Program, Elements of Python, Entering Expressions into the Interactive Shell, The Integer, Floating-Point, and String Data Types, String Concatenation and Replication, Storing Values in Variables**.**

Unit 3: Flow control and Functions **(12 Lectures)**

Boolean Values, Comparison Operators, Boolean Operators, Mixing Boolean and Comparison Operators, Elements of Flow Control, Program Execution, Flow Control Statements, Importing Modules, Ending a Program Early with sys.exit(), def Statements with Parameters, Return Values and return Statements, The None Value, Keyword Arguments and print(), Local and Global Scope, The global Statement, Exception Handling.

Unit 4: List, Dictionary, String and Tuples **(15 Lectures)**

String, String functions, Manipulating Strings, Lists: Creating Lists; Operations on Lists; Built-in Functions on Lists; Implementation of Stacks and Queues using Lists; Nested Lists. Dictionaries: Creating Dictionaries; Operations on Dictionaries; Built-in Functions on Dictionaries; Dictionary Methods; Populating and Traversing Dictionaries. Tuples and Sets: Creating Tuples; Operations on Tuples; Built-in Functions on Tuples; Tuple Methods; Creating Sets; Operations on Sets; Built-in Functions on Sets; Set Methods.

Suggested Readings:

1. Yashavant Kanetkar and Aditya Kanetkar , Let Us Python,BPB,3rd Edition.
2. Sheetal Taneja and Naveen Kumar, Python Programming- A modular approach with Graphics, Database,
   Mobile and Web applications, Pearson, Sixteenth Impression-2023,
3. T. Budd, Exploring Python, TMH, 1st Ed, 2011
4. Python Tutorial/Documentation www.python.or 2015
5. Allen Downey, Jeffrey Elkner, Chris Meyers, How to think like a computer scientist: learning with Python , Freely available online.2012

COMSMIN606 L PYTHON PROGRAMMING LAB [Credits:1, Tutorial Hours: 30]

Students are advised to do laboratory/practical practice not limited to, but including the following types of problems:

1. Write a menu driven program to convert the given temperature from Fahrenheit to Celsius and vice versa depending upon users' choice.

2. WAP to calculate total marks, percentage and grade of a student. Marks obtained in each of the three subjects are to be input by the user. Assign grades according to the following criteria:

Grade A: Percentage >=80
Grade B: Percentage>=70 and <80
Grade C: Percentage>=60 and <70
Grade D: Percentage>=40 and <60
Grade E: Percentage<40

3. Write a menu-driven program, using user-defined functions to find the area of rectangle, square, circle and triangle by accepting suitable input parameters from user.

4. WAP to display the first n terms of Fibonacci series.

5. WAP to find factorial of the given number.

6. WAP to implement the use of arrays in Python.

7. WAP to implement String Manipulation in python in Python.

8. WAP to find sum of the following series for n terms: 1 – 2/2! + 3/3! - - - - - n/n!

9. WAP to calculate the sum and product of two compatible matrices.

10. WAP to create Class and Objects in Python.

12. WAP to implement Data Hiding in Python.

13. WAP to implement constructor and destructor for a class in Python.

14. WAP to implement constructor and destructor in Python.

15. WAP to implement different types of inheritance in Python.

16. WAP to implement concept of Overriding in Python.

17. Write programs to create mathematical 3D objects using class.
    a. curve b. sphere c. cone d. arrow e. ring f. cylinder

18. WAP to Check if a Number is Positive, Negative or 0

19. WAP to Check leap year or not.

20. WAP to display calendar of the given month and year.

SEMESTER-VII

Course-MINOR Paper:	**Paper Code-COMSMIN707 R Programming**	**Credits-3**	**Lectures/Week-3**

Course Name: **R PROGRAMMING**

Brief Course Description

This course introduces the fundamentals of R programming for data analysis, statistical computing, and visualization. Students will learn R syntax, data structures, control statements, data manipulation, graphical representation of data, and basic statistical analysis techniques using R. The course emphasizes practical applications of R in data science and analytics.

Prerequisite(s) and/or Note(s)

Basic knowledge of computer fundamentals and elementary mathematics/statistics is desirable. No prior programming experience is required.

Course Objectives

The course aims to provide students with a strong foundation in R programming, data handling, statistical analysis, and data visualization. It enables learners to develop analytical skills and apply R tools for solving real-world data-driven problems.

Course Outcomes

Upon successful completion of this course, students will be able to write and execute R programs, manage and analyze datasets, create meaningful data visualizations, perform statistical and probabilistic analyses, and apply R programming techniques to solve real-world data-driven problems. They will develop the ability to use R as an effective tool for data analysis, decision-making, research, and advanced studies in data science and analytics.

Knowledge Acquired

Students will gain knowledge of R programming concepts, data structures, control structures, data manipulation techniques, visualization methods, statistical analysis, probability distributions, and hypothesis testing using R.

Skills Gained

Students will develop skills in writing R programs, managing and analyzing datasets, creating informative visualizations, performing statistical computations, and importing/exporting data from various sources.

Competency Developed

Upon completion of the course, students will be able to use R effectively for data analysis, statistical modeling, visualization, and decision-making tasks, preparing them for advanced studies and careers in data science, analytics, and research.

Detailed Syllabus
4th Year: Semester 7
COMSMIN707 R PROGRAMMING [Credits: 3, Lectures 45]

Unit 1: Introduction to R Programming (6 Lectures)

Overview of R programming: Evolution of R language, features of R programming language, what is R? , Why R? , Advantages of R over Other Programming Languages, Downloading and installing of R language, setting up the R environment - Interactivity and Code Execution: R basic syntax - R Studio - R command Prompt, R script file, comments

Unit 2: Data Structures in R (10 Lectures)

Vectors: Creating vectors using c(), Indexing and subsetting vectors in R, Operations on Vectors, Understanding vectorized operations and efficiency – *Matrices* : Creating matrices using matrix(), *Basic matrix operations* : addition, subtraction, multiplication, Transposing matrices - *Lists* : Creating lists with list(), Nested lists and their components, Indexing and extracting values from lists, Manipulating list components – *Arrays* : Creating arrays using array(), Manipulating and transforming array data - *Data Frames:* Introduction to Data Frames, Manipulating Data Frames, Indexing, filtering, and subsetting data frames, Applying functions to columns, Aggregating and summarizing data- *Factors:* Categorical data representation with factors, Modifying and Working with Ordinal Factors in R, Merging and Simplifying Factor Levels in R, Converting factors to other data types.

Unit 3: Control Structures in R (5 Lectures)

Conditional Statements: Basic if and else statements, else if and multiple conditions, Vectorized conditions with ifelse() - Loops in R : For Loop, Nested For Loops, break and next statements, While Loop, Repeat Loop.

Unit 4: Variables, Data Types and operators (5 Lectures)

Concept of Variables: Variable assignment, Finding Variable, Deleting Variables – Atomic Data Types in R (Numeric, Integer, Logical, Character, Complex and Raw) - Operators in R: Arithmetic Operators, Logical Operators, Comparison Operator, Assignment Operators.

Unit 5: Data Manipulation and Cleaning (5 Lectures)

Working with Strings : String manipulation - Data Cleaning: Handling missing values with is.na() and na.omit() - Reshaping Data : Reshaping data frames with reshape(), melt(), cast() - Data Import and Export: Reading data from CSV, Excel, and other formats.

Unit 6: Data Visualization in R (7 Lectures)

Concept of Plotting, Base R plotting functions: plot(), hist(), boxplot(), Customizing plots (titles, labels, colors), Saving plots to file (PDF, PNG) - Visualizations with ggplot2: ggplot2 package, Creating scatter plots, bar plots, histograms, and more, Faceting in ggplot2: Splitting Data into Subplots, Themes in ggplot2 - 3D Visualizations: concept of 3D visualization, Generating 3D scatter plots, surface plots.

Unit 7: R in Statistical Analysis and Resampling Techniques (7 Lectures)

Concept of R in Statistics: R as a tool for statistical analysis, calculation: mean, median, mode, variance, standard deviation, creating frequency tables and cross-tabulations - Probability Distributions in R: Normal Distribution in R, Plotting of Normal Distribution over Histogram in R - Hypothesis Testing: hypothesis testing concepts, T-tests, Chi-squared tests.

Suggested Readings:

1. The Book of R by Tilman M. Davies, no starch press (San Francisco)

2. The Art of R programming by Norman Matloff, no starch press(San Francisco)

3 R for Beginners by Emmanuel Paradis

COMSMIN707	R PROGRAMMING LAB	[Credits: 1, Lecture Hours:30]

Students are advised to do laboratory/practical practice not limited to, but including the following types of problems:

1. Download and install R-Programming environment and install basic packages using install. Packages() command in R.
2. Learn all the basics of R-Programming (Data types, Variables, Operators etc.)
3. Implement R-Loops with different examples.
4. Write an R program to demonstrate variable declaration, data types, arithmetic operations, and output display.
5. Create two vectors and perform addition, subtraction, multiplication, division, and indexing operations.
6. Create two matrices and perform matrix addition, subtraction, multiplication, and transpose operations.
7. Create a list containing student information (Name, Roll No., Marks, Grade) and demonstrate accessing and modifying list elements.
8. Create a three-dimensional array and display specific elements, rows, and columns.
9. Write an R program using if-else and nested if-else statements to determine grades based on student marks.
10. Write R programs using for, while, and repeat loops to generate multiplication tables and factorials.
11. Read data from a CSV file, perform basic analysis, and export the processed data to a new CSV file.
12. Create scatter plots, bar plots, and histograms with appropriate labels and themes using ggplot2.
13. Calculate mean, median, mode, variance, and standard deviation. Perform a t-test and Chi-Square test on a given dataset and interpret the results.
14. Write an R Program for "Hello World ".
15. Write an R Program to Add Two Vectors
16. Find the Sum, Mean and Product of the Vector in R Programming
17. Create an R Program to Take Input From the User
18. Create an R Program to Find the Minimum and Maximum
19. How to create R Multiplication Table
20. Write a R Program to Check Prime Number
21. Write a R Program to check Armstrong Number
22. Write a R Program to Print the Fibonacci Sequence
23. Write a R Program to Check for Leap Year
24. Write a R Program to Check if a Number is Odd or Even in R Programming
25. Write a R Program to print the elements of a vector.

SEMESTER-VIII

Course-MINOR Paper:	**Paper Code-COMSMIN808 ARTIFICIAL INTELLIGENCE**	**Credits-3**	**Lectures/Week-3**

Course Name: **ARTIFICIAL INTELLIGENCE**

Brief Course Description

This Artificial Intelligence (AI) course introduces students to the basic concepts and applications of AI. It covers important topics such as machine learning, natural language processing (NLP), neural networks, and robotics. The course helps students understand how intelligent systems are built and how they can be used to solve real-world problems.

Prerequisite(s) and/or Note(s):
Students should have a basic understanding of mathematics, especially linear algebra, calculus, and probability. Knowledge of programming, particularly Python, is recommended. Familiarity with algorithms and data structures will also be helpful. This course is suitable for students in computer science, engineering, data science, and related fields.

Course Objectives
This course aims to provide students with a strong foundation in the fundamental concepts, principles, and techniques of Artificial Intelligence (AI). It enables learners to understand how AI can be applied to solve real-world problems, develop and implement AI algorithms, and evaluate the performance and effectiveness of AI systems. The course also encourages students to explore the ethical, social, and technological implications of AI while gaining insight into emerging trends and future developments in the field.

Course Outcomes
Upon successful completion of this course, students will be able to understand various Artificial Intelligence approaches, including machine learning and deep learning techniques, and gain knowledge of neural networks, natural language processing, computer vision, and robotics. They will also explore AI applications in diverse domains and evaluate the impact and future potential of AI technologies in modern society.

Knowledge Acquired
Students will gain a strong understanding of key Artificial Intelligence (AI) concepts and methodologies, including supervised learning, unsupervised learning, reinforcement learning, and deep learning. They will learn how neural networks are structured and operate, as well as the fundamental principles of natural language processing, robotics, and computer vision. The course will also introduce students to real-world AI applications in areas such as healthcare, finance, and autonomous systems, helping them understand the wide-ranging impact and potential of AI technologies across different industries.

Skills Gained
During this course, students will develop practical skills in designing, building, testing, and evaluating AI systems. They will learn to prepare and analyze data, train and optimize machine learning models, interpret model outcomes, apply programming techniques for AI solution development, and enhance their problem-solving, analytical, and critical-thinking abilities for real-world applications.

Competencies Developed:

By the end of the course, students will be able to design and implement AI-based solutions for complex real-world problems and apply AI techniques in areas such as intelligent systems, predictive analytics, and automated decision-making. They will be prepared for careers and advanced studies in Artificial Intelligence, Data Science, Software Development, AI Research, and Machine Learning Engineering, with a strong foundation to contribute to the development of innovative and intelligent technologies.

Detailed Syllabus
4th Year: Semester 8
COMSMIN808 **ARTIFICIAL INTELLIGENCE** **[Credits: 3, Lectures 45]**

UNIT 1 – Introduction to Artificial Intelligence **(5 Lectures)**

Introduction to Artificial Intelligence, Background and Applications, Turing Test and Rational Agent approaches to AI, Introduction to Intelligent Agents, their structure, behavior and environment.

UNIT 2 – Problem Solving and Search Techniques in Artificial Intelligence **(20 Lectures)**

Problem Characteristics, Production Systems, Control Strategies, Breadth First Search, Depth First Search, Hill climbing and its Variations, Heuristics Search Techniques: Best First Search, A* algorithm, AO* algorithm, Constraint Satisfaction Problem, Means-End Analysis, Introduction to Game Playing, Min-Max and Alpha-Beta pruning algorithms.

UNIT 3 – Knowledge Representation, Reasoning, and AI Programming **(15 Lectures)**

Introduction to First Order Predicate Logic, Resolution Principle, Unification, Semantic Nets, Conceptual Dependencies, Frames, and Scripts, Production Rules, Conceptual Graphs. Basics of Python Programming for Artificial Intelligence.

UNIT 4 –Reasoning Under Uncertainty and Natural Language Processing **(5 Lectures)**

Truth Maintenance System, Default Reasoning, Probabilistic Reasoning, Bayesian Probabilistic Inference, Possible World Representations. Parsing Techniques, Context-Free and Transformational Grammars, Recursive and Augmented Transition Nets.

Suggested Readings:

1. DAN.W. Patterson, Introduction to A.I and Expert Systems – PHI, 2007.
2. Russell & Norvig, Artificial Intelligence-A Modern Approach, LPE, Pearson Prentice Hall, 2nd edition, 2005.
3. Rich & Knight, Artificial Intelligence – Tata McGraw Hill, 2nd edition, 1991.
4. W.F. Clocksin and Mellish, Programming in PROLOG, Narosa Publishing House, 3rd edition, 2001.
5. Ivan Bratko, Prolog Programming for Artificial Intelligence, Addison-Wesley, Pearson Education, 3rd edition, 2000.
6. "Artificial Intelligence", I.K. International Publishing House, Dr. Ela Kumar.

COMSMIN808T ARTIFICIAL INTELLIGENCE [Credits: 1, Lecture Hours:30]

Artificial Intelligence tutorial as assigned and advised by teacher(s).