DEPARTMENT OF COMPUTER SCIENCE
B.Sc. (H) Computer Science

CATEGORY-I

DISCIPLINE SPECIFIC CORE COURSE – 1: PROGRAMMING USING PYTHON

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Programming using Python	4	3	0	1	Class XII pass	Nil

Learning Objectives

This course is designed as the first course that:

- Introduces programming concepts using Python to Computer Science students.
- Focuses on the development of Python programming to solve problems of different domains.
- Introduces the concept of object- oriented programming.

Learning Outcomes:

On successful completion of the course, students will be able to:

- Understand the basics of programming language
- Develop, document, and debug modular Python programs.
- Apply suitable programming constructs and built-in data structures to solve a problem.
- Use and apply various data objects in Python.
- Use classes and objects in application programs and handle files.

SYLLABUS OF DSC-1

Theory

Unit – 1 **(6 hours)**

Introduction to Programming

Problem solving strategies; Structure of a Python program; Syntax and semantics; Executing simple programs in Python.

Unit – 2 **(12 hours)**

Creating Python Programs

Identifiers and keywords; Literals, numbers, and strings; Operators; Expressions; Input/output statements; Defining functions; Control structures (conditional statements, loop control statements, break, continue and pass, exit function), default arguments.

Unit – 3 **(15 hours)**

Built-in Data Structures

Mutable and immutable objects; Strings, built-in functions for string, string traversal, string operators and operations; Lists creation, traversal, slicing and splitting operations, passing list to a function; Tuples, sets, dictionaries and their operations.

Unit – 4 **(6 hours)**

Object Oriented Programming

Introduction to classes, objects and methods; Standard libraries.

Unit – 5 **(6 hours)**

File and Exception Handling

File handling through libraries; Errors and exception handling.

Practical **(30 hours)**

List of Practicals:

1. WAP to find the roots of a quadratic equation
2. WAP to accept a number 'n' and
    j. Check if 'n' is prime
    k. Generate all prime numbers till 'n'
    l. Generate first 'n' prime numbers This program may be done using functions
3. WAP to create a pyramid of the character '*' and a reverse pyramid

```
    *
   ***
  *****
 *******
*********
```

```
*********
 *******
  *****
   ***
    *
```

8. WAP that accepts a character and performs the following:
    a. print whether the character is a letter or numeric digit or a special character
    b. if the character is a letter, print whether the letter is uppercase or lowercase
    c. if the character is a numeric digit, prints its name in text (e.g., if input is 9, output is NINE)

9. WAP to perform the following operations on a string
    a. Find the frequency of a character in a string.
    b. Replace a character by another character in a string.
    c. Remove the first occurrence of a character from a string.
    d. Remove all occurrences of a character from a string.
10. WAP to swap the first n characters of two strings.
11. Write a function that accepts two strings and returns the indices of all the occurrences of the second string in the first string as a list. If the second string is not present in the first string then it should return -1.
12. WAP to create a list of the cubes of only the even integers appearing in the input list (may have elements of other types also) using the following:
    a. 'for' loop
    b. list comprehension
13. WAP to read a file and
    m. Print the total number of characters, words and lines in the file.
    n. Calculate the frequency of each character in the file. Use a variable of dictionary type to maintain the count.
    o. Print the words in reverse order.
    p. Copy even lines of the file to a file named 'File1' and odd lines to another file named 'File2'.
14. WAP to define a class Point with coordinates x and y as attributes. Create relevant methods and print the objects. Also define a method distance to calculate the distance between any two point objects.
15. Write a function that prints a dictionary where the keys are numbers between 1 and 5 and the values are cubes of the keys.
16. Consider a tuple t1=(1, 2, 5, 7, 9, 2, 4, 6, 8, 10). WAP to perform following operations:
    a. Print half the values of the tuple in one line and the other half in the next line.
    b. Print another tuple whose values are even numbers in the given tuple.
    c. Concatenate a tuple t2=(11,13,15) with t1.
    d. Return maximum and minimum value from this tuple
17. WAP to accept a name from a user. Raise and handle appropriate exception(s) if the text entered by the user contains digits and/or special characters.

Essential Readings

- Taneja, S., Kumar, N. Python Programming- A modular Approach, 1st edition, Pearson Education India, 2018.

- Balaguruswamy E. Introduction to Computing and Problem Solving using Python, 2nd edition, McGraw Hill Education, 2018.

Suggestive Readings

- Brown, Martin C. Python: The Complete Reference, 2nd edition, McGraw Hill Education, 2018.
- Guttag, J.V. Introduction to computation and programming using Python, 2nd edition, MIT Press, 2016.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DISCIPLINE SPECIFIC CORE COURSE – 2: COMPUTER SYSTEM ARCHITECTURE

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Computer System Architecture	4	3	0	1	Class XII pass	NIL

Learning Objectives

The Learning Objectives of this course are as follows:

- Introduces the students to the fundamental concepts of digital computer organization, design and architecture.
- Develop a basic understanding of the building blocks of the computer system and highlights how these blocks are organized together to architect a digital computer system.

Learning Outcomes

On successful completion of the course, students will be able to:

- Design Combinational Circuits using basic building blocks. Simplify these circuits using Boolean algebra and Karnaugh maps. Differentiate between combinational circuits and sequential circuits.
- Represent data in binary form, convert numeric data between different number systems and perform arithmetic operations in binary.
- Determine various stages of instruction cycle, pipelining and describe interrupts and their handling.
- Explain how CPU communicates with memory and I/O devices and distinguish between different types of processors.
- Simulate the design of a basic computer using a software tool.

SYLLABUS OF DSC - 2

Theory

Unit – 1 **(6 hours)**

Digital Logic Circuits

Logic Gates, Truth Tables, Boolean Algebra, Digital Circuits, Combinational Circuits, Introduction to Sequential Circuits, Circuit Simplification using Karnaugh Map, Don't Care Conditions, Flip-Flops, Characteristic Tables, Excitation Table.

Unit – 2 **(9 hours)**

Digital Components (Fundamental building blocks)

Designing of combinational circuits- Half Adder, Full Adder, Decoders, Encoders, Multiplexers, Registers and Memory (RAM, ROM and their types), Arithmetic Microoperations, Binary Adder, Binary Adder-Subtractor.

Unit – 3 **(6 hours)**

Data Representation and Basic Computer Arithmetic

Number System, r and (r-1)'s Complements, data representation and arithmetic operations.

Unit – 4 **(9 hours)**

Basic Computer Organization and Design

Bus organization, Microprogrammed vs Hardwired Control, Instruction Codes, Instruction Format, Instruction Cycle, Instruction pipelining, Memory Reference, Register Reference and Input Output Instructions, Program Interrupt and Interrupt Cycle..

Unit – 5 **(6 hours)**

Processors

General register organization, Stack Organization, Addressing Modes, Overview of Reduced Instruction Set Computer (RISC), Complex Instruction Set Computer (CISC), Multicore processor and Graphics Processing Unit (GPU).

Unit – 6 **(9 hours)**

Memory and Input-Output Organization

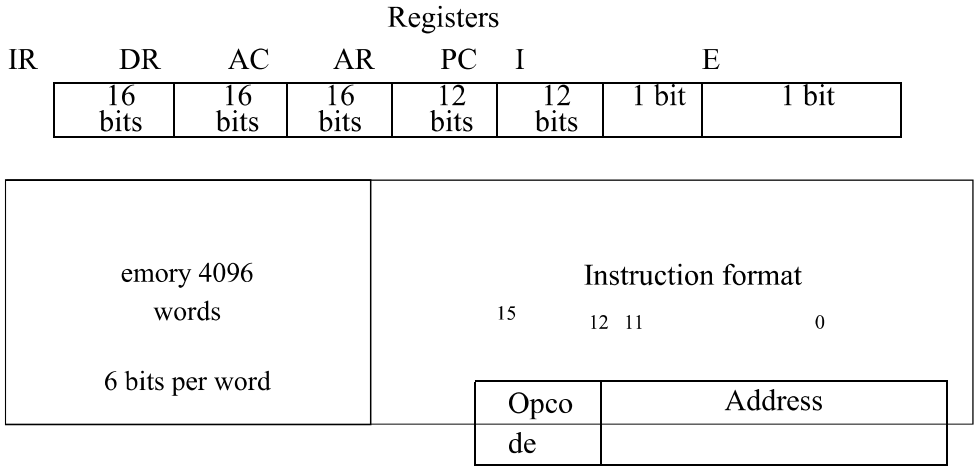
Memory hierarchy (main, cache and auxiliary memory), Input-Output Interface, Modes of Transfer: Programmed I/O, Interrupt initiated I/O, Direct memory access.

Practical **(30 hours)**

List of Practicals:

(Use Simulator – CPU Sim 3.6.9 or any higher version for the implementation)

1. Create a machine based on the following architecture:



Basic Computer Instructions

Memory Reference			Register Reference	
Symbol	Hex		Symbol	Hex
AND	0xxx	Direct Addressing	CLA	7800
ADD	1xxx		CLE	7400
LDA	2xxx		CMA	7200
STA	3xxx		CME	7100
BUN	4xxx		CIR	7080
BSA	5xxx		CIL	7040
ISZ	6xxx		INC	7020
AND_I	8xxx	Indirect Addressing	SPA	7010
ADD_I	9xxx		SNA	7008
LDA_I	Axxx		SZA	7004
STA_I	Bxxx		SZE	7002
BUN_I	Cxxx		HLT	7001
BSA_I	Dxxx		INP	F800
ISZ_I	Exxx		OUT	F400

Refer to Chapter-5 of reference 1 for description of instructions.

Design the register set, memory and the instruction set. Use this machine for the assignments of this section.

2. Create a Fetch routine of the instruction cycle.

3. Write an assembly program to simulate ADD operation on two user-entered numbers.
4. Write an assembly program to simulate SUBTRACT operation on two user-entered numbers.
5. Write an assembly program to simulate the following logical operations on two user-entered numbers.
    i. AND
    ii. OR
    iii. NOT
    iv. XOR
    v. NOR
    vi. NAND
6. Write an assembly program for simulating following memory-reference instructions.
    i. ADD
    ii. LDA
    iii. STA
    iv. BUN
    v. ISZ
7. Write an assembly language program to simulate the machine for following register reference instructions and determine the contents of AC, E, PC, AR and IR registers in decimal after the execution:
    i. CLA
    ii. CMA
    iii. CME
    iv. HLT
8. Write an assembly language program to simulate the machine for following register reference instructions and determine the contents of AC, E, PC, AR and IR registers in decimal after the execution:
    i. INC
    ii. SPA
    iii. SNA
    iv. SZE

9. Write an assembly language program to simulate the machine for following register reference instructions and determine the contents of AC, E, PC, AR and IR registers in decimal after the execution:

    i. CIR

    ii. CIL

10. Write an assembly program that reads in integers and adds them together; until a negative non-zero number is read in. Then it outputs the sum (not including the last number).

11. Write an assembly program that reads in integers and adds them together; until zero is read in. Then it outputs the sum.

Essential Readings

- David A. Patterson and John L. Hennessy. "Computer Organization and Design: The Hardware/Software interface", 5th edition, Elsevier, 2012.
- Mano, M. Computer System Architecture, 3rd edition, Pearson Education, 1993.

Suggestive Readings

- Mano, M. Digital Design, Pearson Education Asia, 1995.
- Null, L., & Lobur, J. The Essentials of Computer Organization and Architecture. 5th edition, (Reprint) Jones and Bartlett Learning, 2018.
- Stallings, W. Computer Organization and Architecture Designing for Performance 8th edition, Prentice Hall of India, 2010.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DISCIPLINE SPECIFIC CORE COURSE – 3: MATHEMATICS FOR COMPUTING

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course				Eligibility criteria	Pre-requisite c the course (if any)
		Lecture	Tutorial	Practical/ Practice			
Mathematics for Computing	4	3	0	1		Class XII pass	NIL

Learning Objectives

The Learning Objectives of this course are as follows:

- Introduces the students to the fundamental concepts and topics of linear algebra and vector calculus.
- To build the foundation for some of the core courses in later semesters.

Learning Outcomes

This course will enable the students to:

- Perform operations on matrices and sparse matrices.
- Compute the determinant, rank and eigenvalues of a matrix.
- Perform diagonalization.
- Perform operations on vectors, the dot product and cross product.
- Represent vectors geometrically and calculate the gradient, divergence, curl.
- Apply linear algebra and vector calculus to solve problems in sub-disciplines of computer science.

SYLLABUS OF DSC – 3

Theory

Unit – 1 **(6 hours)**

Introduction to Matrix Algebra

Echelon form of a Matrix, Rank of a Matrix, Determinant and Inverse of a matrix, Solution of System of Homogeneous & Non-Homogeneous Equations: Gauss elimination and Solution of System of Homogeneous Equations: Gauss Jordan Method.

Unit – 2 **(21 hours)**

Vector Space and Linear Transformation

Vector Space, Sub-spaces, Linear Combinations, Linear Span, Convex Sets, Linear Independence/Dependence, Basis & Dimension, Linear transformation on finite dimensional vector spaces, Inner Product Space, Schwarz Inequality, Orthonormal Basis, Gram-Schmidt Orthogonalization Process.

Unit – 3 **(9 hours)**

EigenValue and EigenVector

Characteristic Polynomial, Cayley Hamilton Theorem, Eigen Value and Eigen Vector of a matrix, Eigenspaces, Diagonalization, Positive Definite Matrices, Applications to Markov Matrices.

Unit – 4 **(9 hours)**

Vector Calculus

Vector Algebra, Laws of Vector Algebra, Dot Product, Cross Product, Vector and Scalar Fields, Ordinary Derivative of Vectors, Space Curves, Partial Derivatives, Del Operator, Gradient of a Scalar Field, Directional Derivative, Gradient of Matrices, Divergence of a Vector Field, Laplacian Operator, Curl of a Vector Field.

Practical **(30 hours)**

List of Practicals:

1. Create and transform vectors and matrices (the transpose vector (matrix) conjugate transpose of a vector (matrix))
2. Generate the matrix into echelon form and find its rank.
3. Find cofactors, determinant, adjoint and inverse of a matrix.
4. Solve a system of Homogeneous and non-homogeneous equations using Gauss elimination method.
5. Solve a system of Homogeneous equations using the Gauss Jordan method.
6. Generate basis of column space, null space, row space and left null space of a matrix space.
7. Check the linear dependence of vectors. Generate a linear combination of given vectors of Rn/ matrices of the same size and find the transition matrix of given matrix space.
8. Find the orthonormal basis of a given vector space using the Gram-Schmidt orthogonalization process.
9. Check the diagonalizable property of matrices and find the corresponding eigenvalue and verify the Cayley- Hamilton theorem.
10. Application of Linear algebra: Coding and decoding of messages using nonsingular matrices.
    eg code “Linear Algebra is fun” and then decode it.
11. Compute Gradient of a scalar field.
12. Compute Divergence of a vector field.
13. Compute Curl of a vector field.

Essential Reading

- Strang Gilbert. Introduction to Linear Algebra, 5th Edition, Wellesley-Cambridge Press, 2021.
- Kreyszig Erwin. Advanced Engineering Mathematics, 10th Edition, Wiley, 2015.
- Strang Gilbert. Linear Algebra and Learning from Data, 1st Edition, Wellesley-Cambridge Press, 2019.
- Jain R. K., Iyengar S.R. K. Advanced Engineering Mathematics, 5th Edition, Narosa, 2016.

Suggestive Reading

- Deisenroth, Marc Peter, Faisal A. Aldo and Ong Cheng Soon. Mathematics for Machine Learning, 1st Edition, Cambridge University Press, 2020.
- (Lipschutz Seymour and Lipson Marc. Schaum's Outline of Linear Algebra, 6th Edition, McGraw Hill, 2017.

DEPARTMENT OF COMPUTER SCIENCE

BSc. (Hons.) Computer Science -DSC

Category I

DISCIPLINE SPECIFIC CORE COURSE – 4: Object Oriented Programming

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC04 Object Oriented Programming with C++	4	3	0	1	Class XII pass with Mathematics	Nil

Learning Objectives

This course is designed to introduce programming concepts using C++ to students. The course aims to develop structured as well as object-oriented programming skills using C++ programming language. The course also aims to achieve competence amongst its students to develop correct and efficient C++ programs to solve problems spanning multiple domains.

Learning outcomes

On successful completion of the course, students will be able to:

- Write simple programs using built-in data types of C++.
- Implement arrays and user defined functions in C++.
- Write programs using dynamic memory allocation, handling external files, interrupts and exceptions.
- Solve problems spanning multiple domains using suitable programming constructs in C++.
- Solve problems spanning multiple domains using the concepts of object oriented programming in C++.

SYLLABUS OF DSC-4

UNIT – I (3 Hours)

Introduction to C++: Overview of Procedural and Object-Oriented Programming, Using main() function, Header Files, Compiling and Executing Simple Programs in C++

UNIT – II (12 Hours)

Programming Fundamentals: Data types, Variables, Operators, Expressions, Arrays, Keywords, Decision making constructs, Iteration, Type Casting, Input-output statements, Functions, Command Line Arguments/Parameters

UNIT – III (15 Hours)

Object Oriented Programming: Concepts of Abstraction, Encapsulation. Creating Classes and objects, Modifiers and Access Control, Constructors, Destructors, Implementation of Inheritance and Polymorphism, Template functions and classes

UNIT – IV (9 Hours)

Pointers and References: Static and dynamic memory allocation, Pointer and Reference Variables, Implementing Runtime polymorphism using pointers and references

UNIT – V (6 Hours)

Exception and File Handling: Using try, catch, throw, throws and finally; Nested try, creating user defined exceptions, File I/O Basics, File Operations

Practical component (if any) -30 Hours

1. Write a program to compute the sum of the first n terms of the following series:

   $sum = 1 - \frac{1}{2^2} + \frac{1}{3^3} -$ . . . .

   The number of terms n is to be taken from the user through the command line. If the command line argument is not found then prompt the user to enter the value of n.
2. Write a program to remove the duplicates from an array.
3. Write a program that prints a table indicating the number of occurrences of each alphabet in the text entered as command line arguments.
4. Write a menu driven program to perform string manipulation (without using inbuilt string functions):
   a. Show address of each character in string
   b. Concatenate two strings.
   c. Compare two strings
   d. Calculate length of the string (use pointers)
   e. Convert all lowercase characters to uppercase
   f. Reverse the string
   g. Insert a string in another string at a user specified position
5. Write a program to merge two ordered arrays to get a single ordered array.

6. Write a program to search a given element in a set of N numbers using Binary search

   (i) with recursion (ii) without recursion.

7. Write a program to calculate GCD of two numbers (i) with recursion (ii) without recursion.

8. Create a Matrix class. Write a menu-driven program to perform following Matrix operations (exceptions should be thrown by the functions if matrices passed to them are incompatible and handled by the main() function):

   a. Sum

   b. Product

   c. Transpose

9. Define a class Person having name as a data member. Inherit two classes Student and Employee from Person. Student has additional attributes as course, marks and year and Employee has department and salary. Write display() method in all the three classes to display the corresponding attributes. Provide the necessary methods to show runtime polymorphism.

10. Create a Triangle class. Add exception handling statements to ensure the following conditions: all sides are greater than 0 and sum of any two sides are greater than the third side. The class should also have overloaded functions for calculating the area of a right angled triangle as well as using Heron's formula to calculate the area of any type of triangle.

11. Create a class Student containing fields for Roll No., Name, Class, Year and Total Marks. Write a program to store 5 objects of Student class in a file. Retrieve these records from the file and display them.

12. Copy the contents of one text file to another file, after removing all whitespaces.

Essential/recommended readings

1. Stephen Prata, *C++ Primer Plus*, 6th Edition, Pearson India, 2015.
2. E Balaguruswamy, *Object Oriented Programming with C++*, 8th edition, McGraw-Hill Education, 2020.
3. D.S. Malik, *C++ Programming: From Problem Analysis to Program Design*, 6th edition, Cengage Learning, 2013.

Suggestive readings

(i) Schildt, H. *C++: The Complete Reference*, 4th edition, McGraw Hill, 2003

(ii) Forouzan, A. B., Gilberg, R. F. *Computer Science: A Structured Approach using C++*, 2nd edition, Cengage Learning, 2010

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DISCIPLINE SPECIFIC CORE COURSE – 5: Discrete Mathematical Structures

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC 05 Discrete Mathematical Structures	4	3	0	1	Class XII pass with Mathematics	Nil

Learning Objectives

This course is designed as a foundational course to make students learn about the mathematical constructs that are used in Computer Science such as Boolean algebra, sets, relations, functions, principles of counting, and recurrences. In this course, the knowledge of mathematical notation, ideas and concepts learnt at the pre-college levels is extended to orient the students towards mathematical thinking required in Computer Science.

Learning outcomes

On successful completion of the course, students will be able to:

- Relate mathematical concepts and terminology to examples in the domain of Computer Science.
- Model real world problems using various mathematical constructs.
- Use different proofing techniques; construct simple mathematical proofs using logical arguments.
- Formulate mathematical claims and construct counterexamples.

SYLLABUS OF DSC- 5
UNIT – I (06 Hours)
Sets, Functions, Sequences and Summations, Relations: Sets: Set Operations, Computer Representation of Sets, Countable and Uncountable Set, Principle of Inclusion and Exclusion, Multisets; Functions: One-to-one and Onto Functions, Inverse Functions and Compositions of

Functions, Graphs of Functions Sequences and Summations: Sequences, Special Integer Sequences, Summations; Relations: Properties of Binary Relations, Equivalence relations and Partitions, Partial Ordering Relations and Lattices.

UNIT – II (09 Hours)

Logic and Proofs: Propositional Logic, Propositional Equivalences, Use of first-order logic to express natural language predicates, Quantifiers, Nested Quantifiers, Rules of Inference, Introduction to Proofs, Proof Methods and Strategies, Mathematical Induction.

UNIT – III (09 Hours)

Number Theory: Division and Integers, Primes and Greatest Common Divisors, Representation of Integers, Algorithms for Integer Operations, Modular Exponentiation, Applications of Number Theory.

UNIT – IV (06 Hours)

Combinatorics/Counting: The Pigeonhole Principle, Permutations and Combinations, Binomial Coefficients, Generalized Permutations and Combinations, Generating Permutations and Combinations.

UNIT – V (09 Hours)

Graphs and Trees: Graphs: Basic Terminology, Multigraphs and Weighted Graphs, Paths and Circuits, Eulerian Paths and Circuits, Hamiltonian paths and Circuits, Shortest Paths, Spanning Trees, Graph Isomorphism, Planar Graphs; Trees: Trees, Rooted Trees, Path Lengths in Rooted Trees.

UNIT – VI (06 Hours)

Recurrence: Recurrence Relations, Generating Functions, Linear Recurrence Relations with Constant Coefficients and their solution.

Practical component (if any) – 30 Hours

1. Create a class SET. Create member functions to perform the following SET operations:
    1) is member: check whether an element belongs to the set or not and return value as true/false.
    2) powerset: list all the elements of the power set of a set .
    3) subset: Check whether one set is a subset of the other or not.
    4) union and Intersection of two Sets.
    5) complement: Assume Universal Set as per the input elements from the user.
    6) set Difference and Symmetric Difference between two sets.
    7) cartesian Product of Sets.

    Write a menu driven program to perform the above functions on an instance of the SET class.

2. Create a class RELATION, use Matrix notation to represent a relation. Include member functions to check if the relation is Reflexive, Symmetric, Anti-symmetric, Transitive. Using these functions check whether the given relation is: Equivalence or Partial Order relation or None

3. Write a Program that generates all the permutations of a given set of digits, with or without repetition.
4. For any number n, write a program to list all the solutions of the equation $x_1 + x_2 + x_3 + ... + x_n = C$, where C is a constant (C<=10) and $x_1, x_2, x_3, ..., x_n$ are nonnegative integers, using brute force strategy.
5. Write a Program to evaluate a polynomial function. (For example store $f(x) = 4n^2 + 2n + 9$ in an array and for a given value of n, say n = 5, compute the value of f(n)).
6. Write a Program to check if a given graph is a complete graph. Represent the graph using the Adjacency Matrix representation.
7. Write a Program to check if a given graph is a complete graph. Represent the graph using the Adjacency List representation.
8. Write a Program to accept a directed graph G and compute the in-degree and out-degree of each vertex.

Essential/recommended readings

1. Liu, C. L., Mohapatra, D. P. *Elements of Discrete Mathematics: A Computer Oriented Approach*, 4th edition, Tata McGraw Hill, 2017.
2. Rosen, K. H.. *Discrete Mathematics and Its Applications*, 8th edition, McGraw Hill, 2018.

Suggestive readings

(i) Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein C. *Introduction to Algorithms*, 4th edition, Prentice Hall of India. 2022.
(ii) Trembley, J. P., Manohar, R. *Discrete Mathematical Structures with Application to Computer Science*, Tata McGraw Hill, 1997.
(iii) Albertson, M. O. and Hutchinson, J. P. *Discrete Mathematics with Algorithms*, John Wiley and Sons, 1988.

DISCIPLINE SPECIFIC CORE COURSE – 6: Probability for Computing

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC06 Probability for computing	**4**	**3**	**0**	**1**	Class XII pass with Mathematics	**Nil**

Learning Objectives

This course introduces the students to the fundamental concepts and topics of probability and statistics, whose knowledge is important in other computer science courses. The course aims to build the foundation for some of the core courses in later semesters.

Learning outcomes

After successful completion of this course, the student will be able to:

- Use probability theory to evaluate the probability of real-world events.
- Describe discrete and continuous probability distribution functions and generate random numbers from the given distributions.
- Find the distance between two probability distributions
- Define and quantify the information contained in the data.
- Perform data analysis in a probabilistic framework.
- Visualize and model the given problem using mathematical concepts covered in the course.

SYLLABUS OF DSC-6 UNIT-I (09 Hours)

Basic Probability: Introduction to the notion of probability, Random experiment, Sample space and Events, Probability defined on events, Algebra of events. Conditional probabilities, independent events, Bayes' theorem.

UNIT-II (12 Hours)

Random Variables: Introduction to Random Variables, Probability mass/density functions, Cumulative distribution functions. Discrete Random Variables (Bernoulli, Binomial, Poisson, Multinomial and Geometric). Continuous Random Variables (Uniform, Exponential and Normal). Expectation of a Random Variable, Expectation of Function of a Random Variable and Variance. Markov inequality, Chebyshev's inequality, Central Limit Theorem, Weak and Strong Laws of Large Numbers.

UNIT-III (09 Hours)

Joint Distributions: Jointly distributed Random Variables, Joint distribution functions, Independent Random Variables, Covariance of Random Variables, Correlation Coefficients,

Conditional Expectation.

UNIT-IV (15 Hours)

Markov Chain and Information Theory: Introduction to Stochastic Processes, Chapman–Kolmogorov equations, Classification of states, Limiting and Stationary Probabilities. Random Number Generation, Pseudo Random Numbers, Inverse Transformation Method, Rejection Method, Uncertainty, Information and Entropy, Mutual Information, KL Divergence.

Practical component (if any) – 30 Hours

The goal of this lab is to develop data interpretation skills. Following exercises are designed to enable students to understand data characteristics either by visualization or by interpreting computed measures. All the exercises are to be completed using MS Excel functions and graphs. At the end of each exercise, the student should be able to draw a conclusion and state in a concise manner. Teachers are expected to guide students to obtain real data available through the internet for the following exercises.

1. Plotting and fitting of Binomial distribution and graphical representation of probabilities.
2. Plotting and fitting of Multinomial distribution and graphical representation of probabilities.
3. Plotting and fitting of Poisson distribution and graphical representation of probabilities.
4. Plotting and fitting of Geometric distribution and graphical representation of probabilities.
5. Plotting and fitting of Uniform distribution and graphical representation of probabilities.
6. Plotting and fitting of Exponential distribution and graphical representation of probabilities.
7. Plotting and fitting of Normal distribution and graphical representation of probabilities.
8. Calculation of cumulative distribution functions for Exponential and Normal distribution.
9. Given data from two distributions, find the distance between the distributions.
10. Application problems based on the Binomial distribution.
11. Application problems based on the Poisson distribution.
12. Application problems based on the Normal distribution.
13. Presentation of bivariate data through scatter-plot diagrams and calculations of covariance.
14. Calculation of Karl Pearson's correlation coefficients.
15. To find the correlation coefficient for a bivariate frequency distribution.
16. Generating Random numbers from discrete (Bernoulli, Binomial, Poisson) distributions.

17. Generating Random numbers from continuous (Uniform, Normal) distributions.
18. Find the entropy from the given data set.

Essential/recommended readings

1. Ross **S**heldon M. *Introduction to Probability Models*, 12th Edition, Elsevier, 2019.
2. Trivedi, K. S. *Probability and Statistics with Reliability, Queuing and Computer Science Applications*, 2nd edition, Wiley, 2015.
3. Deisenroth, Marc Peter, Faisal A. Aldo and Ong Cheng Soon, *Mathematics for Machine Learning*, 1st edition, Cambridge University Press, 2020.
4. Ian F. Blake, *An Introduction to Applied Probability*, John Wiley.

Suggestive readings

(i) Johnson James L., *Probability and Statistics for Computer Science*, 6th edition, Wiley, 2004.

(ii) Forsyth David, *Probability and Statistics for Computer Science*, 1st edition, Springer, 2019.

(iii) Freund J.E., *Mathematical Statistics with Applications*, 8th edition, Pearson Education, 2013.

(iv) Devore Jay L., *Probability and Statistics for Engineering and the Sciences*, 9th edition, Cengage Learning, 2020.