Syllabus
of
B.Tech.
in
Mathematics and Computing

Department of Engineering Mathematics and Computing

Madhav Institute of Technology & Science
Gwalior-474005

Department of Engineering Mathematics and Computing
B. Tech. (First Semester)

Introduction to Computing
(MAC-250121)

L	T	P	C
3	0	0	3

Course Objectives:
.
- Explain the needs of hardware and software required for a computation task.
- State typical provisions of cyber law that govern the proper usage of Internet and computing resources.
- Explain the working of important application software and their use to perform any engineering activity.
- Demonstrate the use of Operating system commands and shell script

Unit 1:
Computer: Definition, Classification, Organization i.e. CPU, register, Instruction set, Memory & Storage Systems, I/O Devices, and System & Application Software. Operating System: Definition, Function, Types, Management of File, Process & Memory, Introduction to Assemblers, Interpreters, Compilers and Debuggers.

Unit 2:
Computer Networking: Introduction, Introduction to Internet, World Wide Web, E-commerce Computer Security Basics: Introduction to viruses, worms, malware, Trojans, Spyware and Anti-Spyware Software, Different types of attacks like Money Laundering, Information Theft, Email spoofing, Hacking Spamming, Cyber Defamation, pharming Security measures Firewall, Computer Ethics & Good Practices

Unit 3:
Data base Management System: Introduction, File oriented approach and Database approach, Data Models, Architecture of Database System, Data independence, Data dictionary, DBA, Primary Key, Data definition language and Manipulation Languages.

Unit 4:
A brief history of LINUX, architecture of LINUX, features of LINUX, introduction to vi editor. Linux commands- PATH, man, echo, printf, script, passwd, uname, who, date, stty, pwd, cd, mkdir, rmdir, ls, cp, mv, rm, cat, more, wc, lp, od, tar, gzip, file handling utilities, security by file permissions, process utilities, disk utilities, networking commands, unlink, du, df, mount, umount, find, unmask, ulimit, ps, w, finger, arp, ftp, telnet, rlogin.Text Processing utilities and backup utilities , tail, head , sort, nl, uniq, grep, egrep, fgrep, cut, paste, join, tee, pg, comm, cmp, diff, tr, awk, cpio

Unit 5:
Career opportunities Entrepreneurship, Start up: Scope, Funding Opportunities, Other career opportunities; Case Studies Success stories of Google, Facebook, Walmart, Uber etc. in socio-economic progress.

Course Outcomes
After completing this course, the students will be able to:

CO's	Description of CO's
CO1	Explain core components of computing and linking
CO2	Explaining ideas of networking aspect of computer engineering and communication
CO3	Apply knowledge of database system
CO4	Summarizing role of operating system
CO5	Implementing the role of computing in real world applications

RECOMMENDED BOOKS:
1. J. Glenn Brookshear, and Dennis Brylow: Computer Science: An Overview, Pearson, 2010
2. V. Rajaraman, Neeharika and A Dabala: Fundamentals of Computers, PHI, 2011
3. Peter Norton, Introduction to Computers:McGraw Hill Education, 2nd , 2012
4. PradeepSinha: Introduction to Computer Science: A Textbook for Beginners in Informatics, BPB Publication, 6th

Edition, 2015
5. Patt Yale: Introduction to Computing Systems,McGraw Hill Education $_{\text{Ind=ia}}$, 2nd, 2014

Department of Engineering Mathematics and Computing

B. Tech. (First Semester)

Introduction to Computer Programming (MAC – 250122)

L	T	P	C
3	0	2	4

Course Objective:

- Develop ability to write a computer program to solve specified problems.
- Develop skills in algorithmic problem-solving, expressed in a programming language like C.
- Learn what computer science is about, especially hardware, data representations, algorithms, encodings, form of programming.
- Understand fundamentals of programming such as variables, conditional and iterative statement, function and its execution etc.

Unit 1:

Introduction to Programming, Machine Level Languages, Assembly Level Languages, High Level Languages, Program Execution and Translation Process, Problem solving using Algorithms and Flowcharts. Introduction to Programming: Data Types, Constants, Keywords, Operators & Expressions, Precedence of operators and input/output functions.

Unit 2:

Control Statements and Decision Making: The go to statement, The if statement, The if-else statement, Nesting of if statements, The conditional expression, The switch statement, The while loop, The do…while loop, The for loop, The nesting of for loops, The break and continue statement.

Unit 3:

Arrays, Strings & Pointers: One dimensional Arrays, Passing Arrays to Functions, Multidimensional Arrays, Strings, Basics of Pointers & Addresses, Pointer to Pointer, Pointer to Array, Array of Pointers, Types of pointers, Pointer to Strings.

Unit 4:

Functions & Structures: Function Basics, Function Prototypes, Passing Parameter by value and by reference, Passing string to function, Passing array to function, Function returning address, Recursion, Structures & Union, Pointer to Structure, Self-Referential Structures, Dynamic memory allocation by call of function, Storage Classes.

Unit 5:

File Handling: Defining and Opening a file, Closing Files, Input/output Operations on Files, Predefined Streams, Error Handling during I/O Operations, Command Line Arguments, Pre-processor, Directives, Formatted I/O.

Course Outcomes

CO's	Description of CO's
CO1	Recognizing the concept of programming Languages.
CO2	Testing the principles of imperative and structural programming.
CO3	Apply the concept of Arrays and Pointer in programming
CO4	Illustrate the problems and choose suitable programming techniques to develop solutions
CO5	Implementing input/ output operations and basic commands

Reference Books

1. E. Balagurusamy: Programming in ANSI C, Tata McGraw Hill,Sevenths Edition, 2017.
2. ReemaThareja: Programming in C, Oxford publication, Second Edition , 2016.
3. W. Kernighan and Dennis M. Ritchie: The C Programming Language, Pearson , 2015.
4. Matthias Felleisen, Robert BruceFindler , Mathew Flatt, ShriramKrishnamurthi: How to Design Programs: An Introduction to Programming and Computing, MIT Press,Second Edition, 2018.

5. E. Balagurusamy: Object Oriented Programming with C++, Tata McGraw Hill, 2009.
6. B.S. Gottfried: Programming with C, Tata McGraw Hill,3rd Edition, 2018.

Department of Engineering Mathematics and Computing

B. Tech. (First Semester)
Statistical Methods
(MAC -250123)

L	T	P	C
3	1	0	4

Course Objective

- To have knowledge of Data and Central Tendency
- To describe Concept of probability theory and distribution
- To familiarize Correlation and Regression
- To know about the Hypothesis analysis

Unit 1:
Data: quantitative and qualitative, attributes, variables, scales of measurement nominal, ordinal, interval and ratio. Presentation: tabular and graphical, including histogram and ogive, Measures of Central Tendency, Measures of Dispersion: range, quartile deviation, mean deviation, standard deviation, coefficient of variation, Moments and moment generating function, skewness and kurtosis.

Unit 2:
Definition of Bivariate, Correlation and Regression analysis, rank of correlation, Coefficient of correlation, Principle of least squares and Curve fitting (polynomials and exponential curves).

Unit 3:
Basic concept of Probability, Compound probability, Conditional Probability, Bayes' theorem, Definition of random variable, discrete and continuous random variables, functions of random variables, probability mass function and probability density function with illustrations, concepts of expectation, Probability distribution function, discrete and continuous Probability distribution, Binomial, Poisson, Normal, Exponential, uniform distribution

Unit 4:
Sampling Theory, Methods of sampling, sampling distribution of a statistic, types of sampling, test of significance, Weak law of large numbers, Central Limit Theorem, Theory of estimation, types of estimation, interval estimation for large sampling, Maximum likelihood estimator

Unit 5:
Testing of hypothesis, Null and alternative hypothesis, Chi-square χ^2 distribution, t-distribution, Fisher's Z-distribution, Analysis of variance.

Course Outcomes

After completing this course, the students will be able to:

CO's	Description of CO's
CO1	Determine the Central of tendency, Skewness and Kurtosis.
CO2	Interpret the theory of Probability
CO3	Evaluate the Probability distributions
CO4	Acquire the knowledge of correlation and regression analysis
CO5	Analyze the test of hypothesis.

Recommended Books:

1. M Ray and H.S. Sharma:Mathematical Statistics, Ram Prasad Publications, 3rd Edition 2017.
2. V. K. Kapoor, S.C. Gupta: Statistical Methods, S. Chand& Company, 11th Edition 2018.
3. T. Veerarajan: Probability, Statistics and Random Processes, McGraw Hill, 3rd Edition 2008.

4. S. M. Rose: Introduction to Probability Models, Elsevier, 10th Edition 2011.

Department of Engineering Mathematics and Computing

B. Tech. (First Semester)
Elements of Calculus
(MAC-250124)

L	T	P	C
3	1	0	4

Objective of Course

- To understand the basic concepts of differential calculus
- To explore the applications of derivatives
- To familiarize the integral calculus
- To describe multiple integral
- To understand the concepts of Convergence and divergence

Unit 1:
Maclaurins's and Taylor's theorem, Partial differentiation, Euler's theorem, Tangent and Normal, Maxima and Minima of one and two variables.

Unit 2:
Jacobian, Rolle 's Theorem, First mean value theorem, Second mean value theorem, Curvature, radius of curvature, Asymptotes of Cartesian and Polar forms.

Unit 3:
Definite integral as limit of a sum, application in summation of series, Improper integral, Beta and Gamma function and its properties, some transformation of Beta function, some transformations of Gama function, relation between Beta and Gama function, Legendre's duplication formula.

Unit 4:
Multiple integral and their applications, Double and Triple integral, Change of order of integration, Length of the curves, Volumes and Surfaces of solids of revolution.

Unit 5:
Concept of convergence and divergence, Basic test of convergence for sequence and series, P-Series test, Ratio test, Comparison test, Integral test, Cauchy's root test, Test of convergence and divergence of improper integral.

Course Outcomes
After completing this course, the students will be able to:

CO's	Description of CO's
CO1	Applying various theorems to expand the functions of one and two variables.
CO2	Demonstrate the application of derivatives
CO3	Examine the various integrals
CO4	Evaluate the volume and area of surface by using multiple integrals
CO5	Summarising the various convergence test

Recommended Books:

1. E. Kreyszig: Advance Engineering Mathematics, John Wiley & Sons, 10th Edition (2011).
2. R. K. Jain, S. R. K. Iyengar: Advance Engineering Mathematics,Narosa Publishing House Pvt.Ltd, 5th Edition (2016).
3. F. B. Hildebrand: Advanced Calculus for application, Englewood Cliffs, N. J. Prentice- Hall, 2nd Edition (1980).
4. B. S. Grewal: Higher Engineering Mathematics, Khanna Publishers, 43rd Edition (2015).
5. B.V. Ramanna: Higher Engineering Mathematics, McGraw Hill, 1st Edition (2017).

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

Department of Engineering Mathematics and Computing

B. Tech. (First Semester)

Computer Architecture & Organization

MAC-250221

L	T	P	C
3	1	0	4

Course Objectives

- To Explain the computer architecture
- To understand the and CPU
- .• To Explore the arithmetic operation
- To ability of recall the knowledge of understand memory organization and Input –output

UNIT-I

Introduction to Computer Architecture. Von Neuman Architecture, Flynn Classification.
Register Transfer and Micro operations: Register transfer language, Arithmetic Micro-operations, Logic Micro-operations, Shift Micro-operations, Bus and memory transfers. Computer Organization and Design: Instruction cycle, computer registers, common bus system, computer instructions, addressing modes, design of a basic computer

UNIT-II

Central Processing Unit: General register organization, stack organization, Instruction formats, Data transfer and manipulation, program control. RISC, CISC characteristics. Pipeline and Vector processing: Pipeline structure, speedup, efficiency, throughput and bottlenecks. Arithmetic pipeline and Instruction pipeline.

UNIT-III

Computer Arithmetic: Adder, Ripple carry Adder, carry look Ahead Adder, Multiplication, Add and Shift, Array multiplier and Booth Multiplier, Division, restoring and Non-restoring Techniques, Floating Point Arithmetic, Floating point representation, Add, Subtract, Multiplication, Division.

UNIT-IV

Memory organization : Memory Hierarchy, Main memory, Associative Memory, Cache memory-organization and mappings , Virtual memory, Memory management, Introduction to pipelining & Multiprocessor.

UNIT-V

Input –output organization : peripheral devices, I/O interface, Asynchronous Data transfer, Modes of transfer, priority interrupt, DMA (DMA controller, DMA transfer), Input output processor (IoP), Data Transfer- Serial, parallel , Simplex, Half Duplex, Full Duplex.

<u>Course Outcomes:</u>

After completing this course student will be able to:

CO's	Description of CO's
CO1	Acquire the knowledge of Computer Architecture
CO2	Understand the theory and architecture of central processing unit.
CO3	Analyze the arithmetic requirements for a problem
CO4	Learn the concepts of memory organization
CO5	Acquire in a better way the Input -Output of the system.

Reference Books:

1. Computer Organization and Architecture –William Stallings Sixth Edition, PHI Fundamentals of Computer organization and design- Sivaraama Dandamudi, Springer
2. Structured Computer Organization- Andrew S Tanenbaum, Fourth Edition PHI.
3. Computer Organization and Architecture - William Stallings (Pearson Education Asia)
4. Computer Organization and Architecture -John P. Hayes (McGraw -Hill)

5. Computer Organization -V. Carl. Hamacher (McGraw-Hill)

Department of Engineering Mathematics and Computing

B. Tech. (Second Semester)
Differential Equations
(MAC-250222)

L	T	P	C
3	1	0	4

Course Objective

- To understand the concept of Ordinary differential equations
- To familiarize the solution techniques of ODE of higher order with constant and variable coefficients
- To describe how to form partial differential equation
- To explorevarious type of partial differential equation and its application

Unit 1:
Ordinary differential equations of first and higher order, Differential equations in which variables are separable, Homogeneous differential equations, Differential equation reducible to homogeneous form (Non-homogeneous differential of first degree), Linear differential equation (Leibnitz's linear differential equation), Bernoulli's equation of differential equation reducible to linear form, Exact differential equations.

Unit-2:
Linear higher order differential equation with constant coefficients, Homogeneous linear equations or Cauchy's Euler's equations, Solution of simultaneous differential equations.

Unit 3:
Second order differential equations with variable coefficients, Methods: one integral is known, Removable of first derivative, changing of independent variable and variation of parameters, Solution of Differential equation by Series Solution method.

Unit 4:
Introduction of partial differential equation, Formulation of partial differential equation, Linear Partial differential equations of first order and solution techniques Lagrange's method, and Non-LinearPartial differential equations of first orderand standard form I, II, III & IV and Charpit's method.

Unit-5:
Partial differential equations of higher order with constant coefficients, Homogeneousand Non-Homogeneous Linear Partial differential equations, Classification of Partial differential equations, ApplicationofPartial differential equations to solve wave equation and heat equation (one-dimensional) by Separation of variables method.

Course Outcomes
After completing this course, the students will be able to:

CO's	Description of CO's
CO1	Determine the analytic solution of ordinary differential equations
CO2	Interpret the solution of ordinary differential equations with constant and variable coefficients
CO3	Acquire the knowledge of second and higher order differential equation
CO4	Formulate the Partial differential equations
CO5	Evaluate the Partial differential equations of higher order with its application

Recommended Books:

1. E. Kreyszig: Advance Engineering Mathematics, John Wiley & Sons, 10th Edition (2011).
2. R. K. Jain, S. R. K. Iyengar: Advance Engineering Mathematics, Narosa Publishing House Pvt. Ltd., 5th Edition (2016).
3. B. S. Grewal: Higher Engineering Mathematics, Khanna Publisher, 43rd Edition (2015).
4. H. K. Dass: Advance Engineering Mathematics, S. Chand Publisher (2018).
5. B.V. Ramanna: Higher Engineering Mathematics, McGraw Hill Education, 1st Edition (2017).

Department of Engineering Mathematics and Computing
B. Tech. (Second Semester)
Object Oriented Methodology and Programming with C++
(MAC-250223)

L	T	P	C
2	1	2	4

Course Objectives
- To study about the concept of object orientedprogramming.
- To create C++ programs that leverage the object oriented features.
- To apply object oriented techniques to solve real world problems.

Unit-1:
Object Oriented Paradigm, Features of OOPs: Encapsulation, Class and Object, Inheritance, Reusability, Polymorphism, Abstraction etc; Comparison with Procedural Oriented Programming & Object Oriented Programming, Introduction to C++: Data types; operators, their priority and associativity, steps of program execution, First program in C++ and its basic elements; Function overloading; Default augments; References; Inline functions.

Unit-2:
Classes & Objects: Specification of Class, Visibility Modes: Private, Public, Protected, Defining Member Functions, Creating of Objects, Static Data Member, Static Member Function, Array of Objects, Object as Arguments, Friend Function and Class, Member Function,member initializer list.Constructors and Destructors: Introduction, Types of Constructors-Default Constructor, User Defined Constructor, Parameterized Constructor, Destructors. Deference between class and structure.

Unit-3:
Dynamic Allocations: new and delete; Difference between new, delete, malloc and free; Dynamic allocation of objects; Array of Objects; Mutable data members; Self-referential class; Shallow and Deep Copying, Copy Constructor; The *this* pointer; Static member and this pointer; Proxy classes. Operator overloading: overloading unary and binary operators using member and non-member functions.Type casting: implicit, explicit, dynamic, static, reinterpret, Conversion between objects of various classes.

Unit-4:
Inheritance: Introduction to Code Reuse, Visibility Modes, Types of Inheritance: Single Level, Multilevel, Multiple, Hybrid, Ambiguity in Inheritance; Virtual Base Classes, Constructors in Derived Classes.Polymorphism; Dynamic and static binding; Pure virtual function; Abstract and Concrete Classes; Virtual Destructors; Containership: Nesting of classes.

Unit-5:
Exception Handling: try, catch and throw; catch all exception handler, streams and File: Basic concept and class hierarchy, File I/O Operations; Modes of opening a file; Various types of I/O Operations; Function like; open, read, write, seek, tell.Templates: Function template; class template; Template specialization; Default type arguments and templates.Namespaces and their uses.

Course Outcomes
After completing this course, the students will be able to:

CO's	Description of CO's
CO1	Tell the concepts of classes & objects and their significance in real world.
CO2	Explain the benefits of object oriented design.
CO3	Build C++ classes using appropriate features of object oriented programming
CO4	Analyze the utilization of inheritance and polymorphism in the solution of problems.
CO5	Apply object orient programming concepts for real world problem.

Recommended books
1. H M Deitel and P J Deitel:C++ How to Program, Prentice Hall,1998.
2. Robert Lafore:Object Oriented Programming in Turbo C++,The WAITE Group Press, 1994.
3. D Ravichandran:Programming with C++ By, T.M.H, 2003.
4. E Balagurusamy:Object oriented Programming with C++, Tata McGraw-Hill,2001.
5. Herbert Schildt:The Complete Reference in C++, TMH,2002.
6. G. Booch: Object Oriented Analysis & Design, Addison Wesley, 2006.
7. James Martin: Principles of Object Oriented Analysis and Design, Prentice Hall, 1992.

Department of Engineering Mathematics and Computing

B. Tech. (Second Semester)

Linear Algebra

(MAC -250224)

&

(EE (IoT), IT(IoT) & AIR- 250100)

Course Objectives

L	T	P	C
3	1	0	4

- To understand the concept Matrices and its applications
- To understand the various aspect of algebraic structures'
- To explore vector space
- To perceive knowledge of linear transformation and their application

Unit 1:
Matrix, Rank of Matrix, Echelon form, Normal form of matrix, Solution of simultaneous equation by elementary transformation, Consistency of equation, Eigen values and Eigenvectors, Normalized eigenvector, Cayley Hamilton theorem and its application to finding inverse of matrix.

Unit 2:
Introduction of Groups and its properties, Sub-groups, Coset, Lagrange's theorem for finite group, Normal sub-group, Cyclic group, Ring and its properties, Field, Finite field, Integral domain and its properties.

Unit 3:
Vector spaces over the field and its properties, sub-spaces, linear dependent vectors and linear independent vectors, linear span of a set of vectors, basis and dimension of a vector space, sum and direct sum.

Unit 4:
Linear transformation, Kernel and range space of linear transformation, Nullity and Rank, Singular and Non- Singular transformation, Matrix representation of a linear transformation, change of basis and similarity.

Unit 5:
Inner product spaces, Properties of inner product space, Norm space, Schwarz's inequality, Triangular inequality, Parallelogram Law, Orthogonality, Generalized theorem of Pythagoras.

Course Outcomes
After completing this course, the students will be able to:

CO's	Description of CO's
CO1	Determine the solution of Matrix
CO2	Interpret the Group theory and its properties
CO3	Express the vector space
CO4	Acquire the knowledge of Linear transformation
CO5	Illustrate the concept of Inner product spaces

Recommended Books:

1. S. Lipschutz and M. Lipson: Linear Algebra,), Schaum's Outline series, Mc- Graw Hill,4th Edition,2009.
2. S. Boyd and L. Vandenberghe: Introduction to Applied Linear Algebra Vectors, Matrices, and Least Squares, University Printing House, Cambridge CB2 8BS, United Kingdom One Liberty Plaza, 20th Floor, New York, NY 10006, USA, 2018.
3. E. Kreyszig: Advance Engineering Mathematics, John Wiley & Sons, 10th Edition, 2011.
4. R. K. Jain, S. R. K. Iyengar: Advance Engineering Mathematics, Narosa Publishing House Pvt. Ltd, 5th Edition 2016.

Department of Engineering Mathematics and Computing
B. Tech. (Second Semester)
Simulation Modelling and Analysis
(MAC - 250225)

Course Objectives

To understand the system, specify systems using natural models of computation
To explore the modelling techniques
To discuss the prediction of behaviour and decision support

L	T	P	C
3	0	0	3

Unit-1:
Introduction – Simulation Terminologies, Application areas, Model Classification, Types of Simulation, Steps in a Simulation study, Concepts in Discrete Event Simulation, Simulation Examples.

Unit-2:
Statistical Models – Concepts – Discrete Distribution- Continuous Distribution, Poisson Process- Empirical Distributions, Queuing Models – Characteristics- Notation– Queuing Systems, Markovian Models, Generation of Pseudo Random numbers- Properties of random numbers, Techniques for generating random numbers, testing random number generators, Generating Random-Variates- Inverse Transform technique– Acceptance- Rejection technique, Composition & Convolution Method.

Unit-3:
Input Modeling, Data collection, assessing sample independence, Hypothesizing distribution family with data Parameter Estimation, Goodness-of-fit tests, Selecting input models in absence of data, Output analysis for a Single system - Terminating Simulations– Steady state simulations.

Unit-4:
Model Building – Verification of Simulation Models, Calibration and Validation of Models, Validation of Model Assumptions, Validating Input – Output Transformations.

Unit-5:
Simulation Tools, Model Input, High level computer system simulation, CPU Memory Simulation, Comparison of systems via simulation, Simulation Programming techniques, Development of Simulation models.

Course Outcomes

After completing this course, the students will be able to:

CO's	Description of CO's
CO1	Acquire the knowledge Simulation
CO2	Analyze the discrete and continuous Simulation Models
CO3	Evaluate the hypothetical parameters
CO4	Interpret the model building
CO5	Determine the input output of system simulation

Recommended Books:

1. Jerry Banks and John Carson: Discrete Event System Simulation, PHI,Fourth Edition, 2005.
2. Geoffrey Gordon: System Simulation, PHI,Second Edition, 2006.
3. Frank L. Severance: System Modeling and Simulation, Wiley, 2001.
4. Averill M. Law and W.David Kelton: Simulation Modeling and Analysis, McGraw Hill, Third Edition, 2006.
5. Jerry Banks: Handbook of Simulation: Principles, Methodology, Advances, Applications and Practice, Wiley,First edition, 1998.

Department of Engineering Mathematics & Computing
B. Tech. (First Semester)

Programming and Problem Solving (250122)

Experiment List

1. C "Hello, World!" Program
2. C Program to Print an Integer (Entered by the User)
3. C Program to Add Two Integers
4. C Program to Multiply Two Floating-Point Numbers
5. C Program to Find ASCII Value of a Character
6. C Program to Compute Quotient and Remainder
7. C Program to Find the Size of int, float, double and char
8. C Program to Demonstrate the Working of Keyword long
9. C Program to Swap Two Numbers
10. C Program to Check Whether a Number is Even or Odd
11. C Program to Check Whether a Character is a Vowel or Consonant
12. C Program to Find the Largest Number Among Three Numbers
13. C Program to Find the Roots of a Quadratic Equation
14. C Program to Check Leap Year
15. C Program to Check Whether a Number is Positive or Negative
16. C Program to Check Whether a Character is an Alphabet or not
17. C Program to Calculate the Sum of Natural Numbers
18. C Program to Find Factorial of a Number
19. C Program to Generate Multiplication Table
20. C Program to Display Fibonacci Sequence
21. C Program to Find GCD of two Numbers
22. C Program to Find LCM of two Numbers
23. C Program to Display Characters from A to Z Using Loop
24. C Program to Count Number of Digits in an Integer
25. C Program to Reverse a Number
26. C Program to Calculate the Power of a Number
27. C Program to Check Whether a Number is Palindrome or Not
28. C Program to Check Whether a Number is Prime or Not
29. C Program to Display Prime Numbers Between Two Intervals
30. C Program to Check Armstrong Number
31. C Program to Display Armstrong Number Between Two Intervals
32. C Program to Display Factors of a Number
33. C Program to Make a Simple Calculator Using switch...case
34. C Program to Display Prime Numbers Between Intervals Using Function
35. C Program to Check Prime or Armstrong Number Using User-defined Function
36. C Program to Check Whether a Number can be Expressed as Sum of Two Prime Numbers
37. C Program to Find the Sum of Natural Numbers using Recursion
38. C Program to Find Factorial of a Number Using Recursion
39. C Program to Find G.C.D Using Recursion

40. C Program to Convert Binary Number to Decimal and vice-versa
41. C Program to Convert Octal Number to Decimal and vice-versa
42. C Program to Convert Binary Number to Octal and vice-versa
43. C Program to Reverse a Sentence Using Recursion
44. C program to calculate the power using recursion
45. C Program to Calculate Average Using Arrays
46. C Program to Find Largest Element in an Array
47. C Program to Calculate Standard Deviation
48. C Program to Add Two Matrices Using Multi-dimensional Arrays
49. C Program to Multiply Two Matrices Using Multi-dimensional Arrays
50. C Program to Find Transpose of a Matrix
51. C Program to Multiply two Matrices by Passing Matrix to a Function
52. C Program to Access Array Elements Using Pointer
53. C Program Swap Numbers in Cyclic Order Using Call by Reference
54. C Program to Find Largest Number Using Dynamic Memory Allocation
55. C Program to Find the Frequency of Characters in a String
56. C Program to Count the Number of Vowels, Consonants and so on
57. C Program to Remove all Characters in a String Except Alphabets
58. C Program to Find the Length of a String
59. C Program to Concatenate Two Strings
60. C Program to Copy String Without Using strcpy()
61. C Program to Sort Elements in Lexicographical Order (Dictionary Order)
62. C Program to Store Information of a Student Using Structure
63. C Program to Add Two Distances (in inch-feet system) using Structures
64. C Program to Add Two Complex Numbers by Passing Structure to a Function
65. C Program to Calculate Difference Between Two Time Periods
66. C Program to Store Information of Students Using Structure
67. C Program to Store Data in Structures Dynamically
68. C Program to Write a Sentence to a File
69. C Program to Read the First Line From a File
70. C Program to Display its own Source Code as Output
71. C Program to Print Pyramids and Patterns

Skill Based Mini Projects

1. The mouse pointer can be restricted in particular rectangle. The idea is to create a function called **restrictmouse()** which takes four parameters which containing X coordinate and Y coordinate. First point mention the top of the rectangle and the second point mention the bottom of the rectangle. Below are the functions used for the same:

- **initmouse():** use to initialize mouse.
- **showmouse():** shows the mouse pointer on the output screen.
- **restrictmouse():** used to set Horizontal and vertical limit of the mouse pointer by setting the following parameters. **AX = 7** for horizontal and **AX = 8** for vertical.

2. This following program makes use of some sub function, which were already discussed previously, and shows how they can be used to write useful programs like free-hand drawing. Below are the functions used:

- **initmouse():** use to initialize mouse.
- **showmouse():** shows mouse pointer on the output screen.
- **hidemouse():** used to hide mouse while drawing.

- **getmouseposition():** Fetches current location of the pointer and draw line accordingly.

3. (The Sieve of Eratosthenes) A prime integer is any integer greater than 1 that can be divided evenly only by itself and 1. The Sieve of Eratosthenes is a method of finding prime numbers. It works as follows:
a) Create an array with all elements initialized to 1 (true). Array elements with prime subscripts will remain 1. All other array elements will eventually be set to zero.
b) Starting with array subscript 2 (subscript 1 is not prime), every time an array element is found whose value is 1, loop through the remainder of the array and set to zero every element whose subscript is a multiple of the subscript for the element with value 1. For array subscript 2, all elements beyond 2 in the array that are multiples of 2 will be set to zero (subscripts 4, 6, 8, 10, and so on.). For array subscript 3, all elements beyond 3 in the array that are multiples of 3 will be set to zero (subscripts 6, 9, 12, 15, and so on.).
When this process is complete, the array elements that are still set to 1 indicate that the subscript is a prime number. Write a program that uses an array of 1000 elements to determine and print the prime numbers between 1 and 999. Ignore element 0 of the array.

4. (Airline Reservations System) A small airline has just purchased a computer for its new automated reservations system. The president has asked you to program the new system. You'll write a program to assign seats on each flight of the airline's only plane (capacity: 10 seats). Your program should display the following menu of alternatives:

Please type 1 for "first class"
Please type 2 for "economy"

If the person types 1, then your program should assign a seat in the first class section (seats 1–5). If the person types 2, then your program should assign a seat in the economy section (seats 6–10). Your program should then print a boarding pass indicating the person's seat number and whether it's in the first class or economy section of the plane.
Use a single-subscripted array to represent the seating chart of the plane. Initialize all the elements of the array to 0 to indicate that all seats are empty. As each seat is assigned, set the corresponding element of the array to 1 to indicate that the seat is no longer available.
Your program should, of course, never assign a seat that has already been assigned. When the first class section is full, your program should ask the person if it's acceptable to be placed in the economy section (and vice versa). If yes, then make the appropriate seat assignment. If no, then print the message "Next flight leaves in 3 hours."

5. (Total Sales) Use a double-subscripted array to solve the following problem. A company as four salespeople (1 to 4) who sell five different products (1 to 5). Once a day, each salesperson passes in a slip for each different type of product sold. Each slip contains:

a) The salesperson number
b) The product number
c) The total dollar value of that product sold that day

Thus, each salesperson passes in between 0 and 5 sales slips per day. Assume that the information from all of the slips for last month is available. Write a program that will read all this information for last month's sales and summarize the total sales by salesperson by product. All totals should be stored in the double-subscripted array sales. After processing all the information for last month, print the results in tabular format with each of the columns representing a particular salesperson and each of the rows representing a particular product. Cross total each row to get the total sales of each product for last month; cross total each column to get the total sales by salesperson for last month. Your tabular printout should include these cross totals to the right of the totaled rows and to the bottom of the totaled columns.

Department of Engineering Mathematics & Computing
B. Tech. (Second Semester)
OBJECT ORIENTED PROGRAMMING AND METHODOLOGY USING C++
(MAC-250223)

List of Experiments

1. C++ program to demonstrate example of default constructor or no argument constructor.
2. C++ program to demonstrate example of parameterized constructor.
3. C++ program to demonstrate example of copy constructor.
4. C++ program to demonstrate example of constructor overloading.
5. C++ program to demonstrate example of member initializer list.
6. C++ program to demonstrate example of destructors.
7. C++ program to demonstrate example of constructor using this pointer.
8. C++ program to demonstrate example of constructor with default arguments.
9. C++ program to Dynamic Initialization of Objects in C++.
10. C++ program to Set values of data members using default, parameterized and copy constructor
11. C++ program to demonstrate example of simple inheritance.
12. C++ program to demonstrate example of private simple inheritance.
13. C++ program to read and print student's information using two classes and simple inheritance.
14. C++ program to demonstrate example of multilevel inheritance.
15. C++ program to read and print employee information using multiple inheritance.
16. C++ program to demonstrate example of multiple inheritance.
17. C++ program to demonstrate example of hierarchical inheritance to get square and cube of a number.
18. C++ program to read and print employee information with department and pf information using hierarchical inheritance.
19. C++ program for unary minus (-) operator overloading.
20. C++ program for unary increment (++) and decrement (--) operator overloading.
21. C++ program for unary logical NOT operator overloading.
22. C++ program to add two objects using binary plus (+) operator overloading.
23. C++ program to add two distances using binary plus (+) operator overloading.
24. C++ program to create a simple class and object.
25. C++ | Create an object of a class and access class attributes
26. C++ | Create multiple objects of a class
27. C++ | Create class methods
28. C++ | Define a class method outside the class definition
29. C++ | Assign values to the private data members without using constructor
30. C++ | Create an empty class (a class without data members and member functions)
31. C++ | Create a class with setter and getter methods
32. C++ program to create a class to read and add two distance.
33. C++ program to create a class for student to get and print details of a student.
34. C++ program to create a class for student to get and print details of N students.
35. C++ program to demonstrate example of array of objects.
36. C++ program to create class to read and add two times.
37. C++ program to create class to read time in seconds and convert into time in (HH:MM:SS) format.

38. C++ program to create class to read time in HH:MM:SS format and display into seconds.
39. C++ program to demonstrate example of friend function with class.
40. Count the created objects using static member function in C++.
41. Create an object of a class inside another class declaration in C++.
42. Example of private member function in C++.
43. Local Class with Example in C++.
44. Structure with private members in C++.
45. Const Member Functions in C++.
46. Demonstrate Example of public data members in C++.
47. Create a class Point having X and Y Axis with getter and setter functions in C++.
48. Passing an object to a Non-Member function in C++.
49. Accessing Member Function by pointer in C++.
50. Access the address of an object using 'this' pointer in C++.
51. Create a class with public data members only in C++
52. C++ program Input list of candidates and find winner of the Election based on received votes
53. C++ program for Banking Management System using Class.
54. C++ program to create a file.
55. C++ program to read a text file.
56. C++ program to write and read text in/from file.
57. C++ program to write and read values using variables in/from file.
58. C++ program to write and read object using read and write function.
59. C++ program to demonstrate example of tellg() and tellp() function.
60. C++ program to write and read time in/from binary file using fstream.
61. C++ program to write and read an object in/from a binary file.
62. C++ tellg(), seekg() and seekp() Example

Skill Development Projects

1. (Knight's Tour) One of the more interesting puzzlers for chess buffs is the Knight's Tour problem, originally proposed bythe mathematician Euler. The question is this: Can the chess piece called the knight move around an empty chessboard and touch each of the 64 squares once and only once?

2. Create a class HugeInteger that uses a 40-element array of digits to store integers as large as 40-digits each. Providemember functions inputHugeInteger, outputHugeInteger, addHugeIntegers and substractHugeIntegers. For comparing HugeInteger objects, provide functions isEqualTo, isNotEqualTo, isGreaterThan, isLessThan, IsGreaterThanOrEqualTo and isLessThanOrEqualTo—each of these is a "predicate" function that simply returns true if the relationship holds between the two huge integers and returns false if the relationship does not hold. Provide a predicate function isZero. If you feel ambitious, also provide member functions multiplyHugeIntegers, divideHugeIntegers and modulusHugeIntegers.

3. Create a class TicTacToe that will enable you to write a complete program to play the game of tic-tac-toe. The class contains as private data a 3-by-3 double array of integers. The constructor should initialize the empty board to all zeros. Allow two human players. Wherever the first player moves, place a 1 in the specified square; place a 2 wherever the second player moves. Each move must be to an empty square. After each move, determine if the game has been won or if the game is a draw. If you feel ambitious, modify your program so that the computer makes the moves for one of the players automatically. Also, allow the player to specify whether he or she wants to go first or second. If you feel exceptionally ambitious, develop a program that will play three dimensional tic-tac-toe on a 4-by-4-by-4 board (Caution: This is an extremely challenging project that could take many weeks of effort!).

4. Create a class called IntegerSet. Each object of class IntegerSet can hold integers in the range 0 through 100. A set is represented internally as an array of ones and zeros. Array element a[i] is 1 if integer i is in the set. Array element a[j] is 0 if integer j is not in the set. The default constructor initializes a set to the so-called "empty set," i.e., a set whose array

representation contains all zeros.
Provide member functions for the common set operations. For example, provide a unionOfIntegerSets member function that creates a third set which is the set-theoretic union of two existing sets (i.e., an element of the third set's array is set to 1 if that element is 1 in either or both of the existing sets, and an element of the third set's array is set to 0 if that element is 0 in each of the existing sets).
Provide an intersectionOfIntegerSets member function that creates a third set which is the set-theoretic intersection of two existing sets (i.e., an element of the third set's array is set to 0 if that element is 0 in either or both of the existing sets, and an element of the third set's array is set to 1 if that element is 1 in each of the existing sets).

- Provide an insertElement member function that inserts a new integer k into a set (by setting a[k] to 1). Provide a deleteElement member function that deletes integer m (by setting a[m] to 0).
- Provide a setPrint member function that prints a set as a list of numbers separated by spaces. Print only those elements that are present in the set (i.e., their position in the array has a value of 1). Print --- for an empty set.
- Provide an isEqualTo member function that determines if two sets are equal.
- Provide an additional constructor to take five integer arguments which can be used to initialize a set object. If you want to provide fewer than five elements in the set, use default arguments of

-1 for the others.
Now write a driver program to test your IntegerSet class. Instantiate several IntegerSet objects. Test that all your member functions work properly.

5. Create a class RationalNumber (fractions) with the following capabilities:
a) Create a constructor that prevents a 0 denominator in a fraction, reduces or simplifies fractions that are not in reduced form and avoids negative denominators.
b) Overload the addition, subtraction, multiplication and division operators for this class.
c) Overload the relational and equality operators for this class.

6. Develop class Polynomial. The internal representation of a Polynomial is an array of terms. Each term contains a coefficient and an exponent.
The term $2x^4$ has a coefficient of 2 and an exponent of 4. Develop a full class containing proper constructor and destructor functions as well as set and get functions. The class should also provide the following overloaded operator capabilities:
a) Overload the addition operator (+) to add two Polynomials.
b) Overload the subtraction operator (-) to subtract two Polynomials.
c) Overload the assignment operator (=) to assign one Polynomial to another.
d) Overload the multiplication operator (*) to multiply two Polynomials.
e) Overload the addition assignment operator (+=), the subtraction assignment operator
(-=), and the multiplication assignment operator (*=).

7. Write a program that accomplishes each of the following:
a) Create the user-defined class Complex that contains the private integer data members real and imaginary, and declares stream-insertion and stream-extraction overloaded operator functions as friends of the class.
b) Define the stream-insertion and -extraction operator functions. The stream-extraction operator function should determine if the data entered are valid, and if not, it should set ios::failbit to indicate improper input. The input should be of the form

3 + 8i

c) The values can be negative or positive, and it is possible that one of the two values is not provided. If a value is not provided, the appropriate data member should be set to 0. The stream-insertion operator should not be able to display the point if an input error occurred. The output format should be identical to the input format shown above. For negative imaginary values, a minus sign should be printed rather than a plus sign.
d) Write a main function that tests input and output of user-defined class Complex using the overloaded stream-extraction and stream-insertion operators.

8. Write a program with class template Array. The template can instantiate an Array of any element type. Override the template with a specific definition for an Array of float elements (class Array< float >). The driver should demonstrate the instantiation of an Array of int through the template and should show that an attempt to instantiate an Array of float uses the definition provided in class Array< float >.