माधव प्रौद्योगिकी एवं विज्ञान संस्थान, ग्वालियर
MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR
Deemed to be University
(Declared under Distinct Category by Ministry of Education, Government of India)
NAAC ACCREDITED WITH A++ Grade
Gola Ka Mandir, Gwalior (M.P.)- 474005, INDIA
Ph.:+91-751-2409300, E-mail: director@mitsgwalior.in, Website: www.mitsgwalior.in

Department of Engineering Mathematics and Computing 2025

B. Tech. (First Semester)

Computer Programming Lab

25251106

List of Experiments and Programs

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.

Course Outcomes

CO1 Explain basic programming terms, syntax, algorithm and flow chart.

CO2 Solve computational problems using decision control and loops by choosing Appropriate Functions & Structures and file handling operations

CO3 Design a program using the concept of array, pointer, functions and use of modern programming tools and techniques to write efficient program.

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	2	1		1	1	1	1	1	1	2	1	2	1	1
CO2	2	2	2	1	1	1	1	1	1	2	1	2	1	1
CO3	2	1	3	1	1	1	1	1	1	2	1	2	1	1

1 - Slightly; 2 - Moderately; 3 – Substantially

B. Tech. (First Semester)
Computing Lab for Data Analysis
25251107

List of Experiments

1. Lab 1 Introduction to Microsoft Excel
2. Lab 2 Frequency Distributions and Graphs
3. Lab 3 Data Description
4. Lab 4 Probability and Counting Rules
5. Lab 5 Discrete Probability Distributions
6. Lab 6 The Normal Distribution
7. Lab 7 Confidence Intervals
8. Lab 8 Hypothesis Testing
9. Lab 9 Testing the Difference Between Parameters from Two Populations
10. Lab 10 Correlation and Regression
11. Lab 11 Tests for Categorical Variables
12. Lab 12 One-Way Analysis of Variance (ANOVA)
13. Lab 13 Stem-and-Leaf Plots and Frequency Tables
14. Lab 14 Summary Statistics USING SPSS.
15. Lab 15 To calculate and interpret binomial and normal probabilities.
16. Lab 16 Testing a Mean.
17. Lab 17 Paired Samples and Their Differences/
18. Lab 18 Independent Sample and their Differences.

Course Outcomes

CO1 Explain basic Excel Concepts and Formulas terms, syntax, algorithm and flow chart.

CO2 Solve problems of Discrete & Continuous Probability distribution using Hypothesis Testing and testing difference

CO3 Apply ANOVA and SPSS to the real life probability and statistics problems.

Course Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	2	1	2	1	1	1	1	1	1	2	1	2	1	1
CO2	2	2	2	1	1	1	1	1	1	2	1	2	1	1
CO3	2	1	3	1	1	1	1	1	1	2	1	2	1	1

1 - Slightly; 2 - Moderately; 3 – Substantially

Department of Engineering Mathematics and Computing 2025

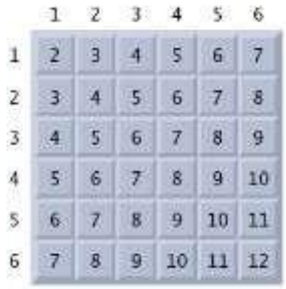
B. Tech. (First Semester)

Micro Project-I
25252109

List of Project

Based on Computer Programming Lab

1. (Dice Rolling) Write a program that simulates the rolling of two dice. The program should use rand twice to roll the first die and second die, respectively. The sum of the two values should then be calculated. [Note: Because each die can show an integer value from 1 to 6, then the sum of the two values will vary from 2 to 12, with 7 being the most frequent sum and 2 and 12 the least frequent sums.] Figure shows the 36 possible combinations of the two dice. Your program should roll the two dice 36,000 times. Use a one-dimensional array to tally the numbers of times each possible sum appears. Print the results in a tabular format. Also, determine if the totals are reasonable; i.e., there are six ways to roll a 7, so approximately one-sixth of all the rolls should be 7.



	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	10
5	6	7	8	9	10	11
6	7	8	9	10	11	12

2. (Snake Ladder Game): Design a program to implement the snake ladder game with three complexity levels: Simple, Medium and Hard. Implement all rules usually followed in playing the Game,

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.

6. **Missing number in array**: Given an array of size N-1 such that it only contains distinct integers in the range of 1 to N. Display missing element. Complete the function MissingNumber() that takes array and N as input parameters and returns the value of the missing number.

Input:
N = 5
A[] = {1,2,3,5}
Output: 4

7. Leaders in an Array: Given an array A of positive integers. Your task is to find the leaders in the array. An element of array is leader if it is greater than or equal to all the elements to its right side. The rightmost element is always a leader.

The task is to complete the function leader() which takes array A and n as input parameters and returns an array of leaders in order of their appearance.

Input:
n = 6
A[] = {16,17,4,3,5,2}
Output: 17 5 2
Explanation: The first leader is 17
as it is greater than all the elements
to its right. Similarly, the next

leader is 5. The right most element
is always a leader so it is also
included.

8. Kth Smallest Element: Given an array arr[] and an integer K where K is smaller than size of array, the task is to find the Kth smallest element in the given array. It is given that all array elements are distinct. Your task is to complete the function kthSmallest() which takes the array arr[], integers l and r denoting the starting and ending index of the array and an integer K as input and returns the Kth smallest element.

Input:
N = 6
arr[] = 7 10 4 3 20 15
K = 3
Output : 7
Explanation :
3rd smallest element in the given
array is 7.

9. Majority Element: Given an array A of N elements. Find the majority element in the array. A majority element in an array A of size N is an element that appears more than N/2 times in the array. The task is to complete the function majorityElement() which returns the majority element in the array. If no majority exists, return -1.

Input:
N = 5
A[] = {3,1,3,3,2}
Output:
3
Explanation:
Since, 3 is present more
than N/2 times, so it is
the majority element.

10. Minimum Number of Jumps: Given an array of N integers arr[] where each element represents the maximum length of the jump that can be made forward from that element. This means if arr[i] = x, then we can jump any distance y such that y ≤ x.
Find the minimum number of jumps to reach the end of the array (starting from the first element). If an element is 0, then you cannot move through that element.
Note: Return -1 if you can't reach the end of the array.
Your task is to complete function minJumps() which takes the array arr and it's size N as input parameters and returns the minimum number of jumps. If not possible return -1.

Input:
N = 11
arr[] = {1, 3, 5, 8, 9, 2, 6, 7, 6, 8, 9}
Output: 3
Explanation:
First jump from 1st element to 2nd
element with value 3. Now, from here

we jump to 5th element with value 9,
and from here we will jump to the last.

Based on Computing Lab for Data Analysis

1. **Personal Budget Tracker**
   Create a spreadsheet to track monthly income, expenses, and savings goals.
2. **Monthly Expense Report**
   Design a report to categorize and visualize spending patterns over the month.
3. **Sales Performance Dashboard**
   Develop an interactive dashboard to analyse sales data and visualize trends using charts.
4. **Basic Inventory Management**
   Build a simple inventory system to track stock levels, suppliers, and reorder points.
5. **Employee Attendance Tracker**
   Create a sheet to record employee attendance, absences, and calculate attendance rates.
6. **Weekly Meal Planning Sheet**
   Design a planner that organizes meals for the week along with a grocery list.
7. **Fitness Goal Tracker**
   Develop a log to track workouts, nutrition, and progress towards fitness goals.
8. **Gantt Chart for Project Management**
   Create a Gantt chart to visualize project timelines, tasks, and milestones.
9. **Simple Invoice Generator**
   Design a template for generating invoices that includes client details and itemized billing.
10. **Customer Feedback Analysis**
    Analyze customer feedback data and present insights through charts and graphs.
11. **Event Planning Checklist**
    Build a checklist to help organize tasks and deadlines for event planning.
12. **Time Tracking Sheet**
    Create a time log for tracking hours spent on different tasks or projects.
13. **Book Reading Log**
    Develop a spreadsheet to record books read, authors, genres, and personal ratings.
14. **Social Media Content Calendar**
    Design a calendar to schedule and track social media posts and engagement.
15. **Sales Forecasting Model**
    Create a model to predict future sales based on historical data and trends.
16. **Travel Expense Tracker**
    Build a tracker to monitor expenses during travel, including transportation, lodging, and meals.
17. **Student Gradebook**
    Develop a gradebook to record student grades, assignments, and calculate averages.
18. **Product Pricing Comparison**
    Create a comparison sheet to analyze prices of similar products across different retailers.
19. **Daily Water Intake Tracker**
    Design a log to monitor daily water intake and hydration goals.
20. **Household Chores Schedule**
    Build a schedule to assign and track household chores among family members.
21. **Job Application Tracker**
    Create a tracker to monitor job applications, statuses, and follow-up actions.
22. **Charity Donation Log**
    Develop a log to record donations made, recipients, and total contributions over time.

23. **Recipe Organizer**
    Create a spreadsheet to categorize and store favorite recipes with ingredients and instructions.
24. **Debt Repayment Schedule**
    Build a plan to track debts, payments, and remaining balances over time.
25. **Market Research Survey Analysis**
    Analyze survey data for market research and summarize findings with visualizations.
26. **Email Subscription Tracker**
    Create a log to track email subscriptions, unsubscribe dates, and frequency.
27. **Simple Profit and Loss Statement**
    Develop a template to calculate and present profits and losses for a small business.
28. **Hobby Expense Tracker**
    Create a sheet to track expenses related to hobbies or interests over time.
29. **Real Estate Investment Calculator**
    Build a calculator to assess potential returns on real estate investments.
30. **Seasonal Sales Tracker**
    Design a spreadsheet to monitor sales data across different seasons and identify trends.

Based on Differential Equations

1. Mathematical Modeling and Analysis of Heat Transfer Dynamics Using Newton's Law of Cooling through Differential Equations
2. Modeling Radioactive Decay Processes Using Differential Equations: A Mathematical Approach
3. Analysis of Transient Behavior in L-R Circuits Using Differential Equations: A Study of Inductive-Resistive Systems
4. Exploring Charge-Discharge Dynamics in C-R Circuits Through Differential Equation Modeling
5. Analyzing Compound Interest Growth: A Differential Equation Approach to Banking Finance
6. Dynamics of Free Fall: Analyzing the Velocity of Falling Objects with Air Resistance Using Differential Equations
7. Modeling Concentration Changes in Mixing Problems: Application of Differential Equations in Chemistry and Environmental Science
8. Study of Heat Transfer in a Rod: Application of Differential Equations in Thermal Conductivity