| Core Practical 2<br>For the students admitted from A.Y. 2023-2024& onwards | |
|---|---|
| Offering Department: **Computer Application** | Offered to: **Master of Computer Application** |

| Semester – I | | |
|---|---|---|
| Course Code | Course Title | Course Credit and Hours |
| **23MCACC106** | **Core Practical 2:** Problem Solving Methodologies using C Language | **2 Credits - 4 hrs/wk** |

## Course Description:

Introduction to C Programming is a course that provides students with a foundation in programming using the C programming language. The course is designed to equip students with the skills and knowledge required to write efficient and effective code and design algorithms in C. Throughout the course, students will learn about programming concepts such as variables, data types, control structures, arrays, functions, and pointers. They will also gain an understanding of software development, debugging, and testing.At the end of the course, students will have gained a strong foundation in C programming, including an understanding of the syntax and semantics of the language. They will be able to write programs using variables, data types, control structures, arrays, functions, and pointers. Additionally, they will be able to design and implement algorithms in C, debug and test programs, and apply software development principles to build software solutions using C.

## Course Purpose:

C programming helps students to develop their problem-solving skills by enabling them to break down complex problems into smaller, more manageable pieces. C programming also teaches students about computer memory management, including concepts such as pointers, arrays, and dynamic memory allocation. Writing efficient and optimized code is another key aspect of C programming, which is essential for developing high-performance software. Moreover, C programming provides a strong foundation for students to learn more advanced programming languages, such as Java, Android and Python. Therefore, teaching C programming in university syllabus is an important part of computer science education that helps students to develop their programming skills and prepare them for careers in software development and related fields.

**Course Outcomes:** Upon completion of this course, the learners will be able to

| CO No. | CO Statement | Bloom's Taxonomy Level ($K_1$ to $K_6$) |
|---|---|---|
| $CO_1$ | Apply control structures such as loops and decision-making constructs to solve programming problems that involve file handling | K2 |
| $CO_2$ | Utilize C language features to develop efficient and scalable programs that address regular problems involving file handling | K5 |
| $CO_3$ | Debug and troubleshoot C code using appropriate tools and techniques, including debugging file I/O errors | K4 |
| $CO_4$ | Write simple C programs that utilize variables, data types, and basic input/output operations, including file I/O | K1 |
| $CO_5$ | Design and implement complex algorithms using functions, arrays, pointers, and file handling operations | K3 |

| Course Content | Hours |
|---|---|
| **Exercise-I:Managing I/O Operations, Control Structures** | |
| ▪ **I/O Operations**<br>1. Calculate the area of a circle.<br>2. Convert temperature from Fahrenheit to Celsius.<br>3. Create a program that prints the Fibonacci series.<br>4. Create a program that calculates the sum of digits of a number.<br>5. Create a program that calculates the factorial of a number using recursion.<br>▪ **Control Statements:**<br>1. Write a program to check whether a given number is positive, negative, or zero using if-elsestatements.<br>2. Write a program to find the largest of two given numbers using if-else statements.<br>3. Write a program to find the largest of three given numbers using nested if-else statements.<br>4. Write a program to check whether a given year is a leap year or not using if-else statements.<br>5. Write a program to check whether a given character is a vowel or consonant using switch Statements.<br>6. Write a program to check whether a given number is prime or not using if-else statements.<br>7. Write a program to check whether a given string is a palindrome or not using if-else statements.<br>8. Write a program to find the roots of a quadratic equation ax^2 + bx + c = 0 using if-elseStatements.<br>9. Write a program to find the grade of a student based on their marks using if-else statements.<br>10. Write a program to calculate the electricity bill for a given number of units using if-elseStatements.<br>▪ **Loop Control Structures:**<br>1. Write a program to print the first n natural numbers using a while loop.<br>2. Write a program to print the multiplication table for a given number using a for loop.<br>3. Write a program to calculate the sum of all even numbers from 1 to n using a do-while loop.<br>4. Write a program to find the factorial of a given number using a while loop.<br>5. Write a program to generate the Fibonacci series up to a given number using a for loop.<br>6. Write a program to print the sum of all prime numbers between 1 and n using a for loop.<br>7. Write a program to print the sum of digits of a given number using a while loop.<br>8. Write a program to check whether a given number is a palindrome or not using a while loop.<br>9. Write a program to find the largest and smallest elements in an array using a for loop. | **9 Hrs** |
| **Exercise-II: Functions , Arrays and String handling** | |
| ▪ **Functions:**<br>1. Write a program to find the factorial of a given number using a function.<br>2. Write a program to find the maximum of two given numbers using a | **9 Hrs** |

| Course Content | Hours |
|---|---|
| function.<br>3. Write a program to find the sum of all elements in an array using a function.<br>4. Write a program to check whether a given number is Armstrong or not using a function.<br>5. Write a program to find the GCD of two given numbers using a function.<br>6. Write a program to swap two given numbers using a function.<br>7. Write a program to convert a given decimal number to binary using a function.<br>8. Write a program to calculate the area of a circle using a function.<br>9. Write a program to check whether a given number is a palindrome or not using a function.<br>10. Write a program to calculate the power of a given number using a function.<br>▪ **Arrays:**<br>1. Write a program to find the largest element in an array.<br>2. Write a program to reverse an array using a loop.<br>3. Write a program to find the second largest element in an array.<br>4. Write a program to find the sum of all even elements in an array.<br>5. Write a program to merge two sorted arrays into a new array.<br>6. Write a program to remove duplicates from an array.<br>7. Write a program to rotate an array to the left by a given number of positions.<br>8. Write a program to find the frequency of each element in an array.<br>9. Write a program to insert an element into an array at a given position.<br>▪ **Strings:**<br>1. Write a program to find the length of a given string using the strlen() function.<br>2. Write a program to concatenate two strings using the strcat() function.<br>3. Write a program to copy one string to another using the strcpy() function.<br>4. Write a program to compare two strings using the strcmp() function.<br>5. Write a program to convert a given string to uppercase using the toupper() function.<br>6. Write a program to convert a given string to lowercase using the tolower() function.<br>7. Write a program to reverse a given string using a loop.<br>8. Write a program to find the occurrence of a given character in a string using a loop.<br>9. Write a program to split a given string into words using the strtok() function.<br>10. Write a program to remove all the spaces from a given string using a loop. | |
| **Exercise-III: Structures, Unions and Pointer** | |
| ▪ **Structures**<br>1. Write a program to define and initialize a structure for a student with attributes such as name,roll number, and marks.<br>2. Write a program to read and display the details of a student using a structure.<br>3. Write a program to find the average marks of a group of students using a structure.<br>4. Write a program to find the total marks of a student using a structure and a function. | **9 Hrs** |

| Course Content | Hours |
|---|---|
| 5. Write a program to sort a group of students based on their marks using structures and bubble<br>6. sort algorithm.<br>7. Write a program to define and initialize a structure for a book with attributes such as title,author, and price.<br>8. Write a program to read and display the details of a book using a structure.<br>9. Write a program to update the price of a book in a library using a structure.<br>10. Write a program to delete a book from a library using a structure and array manipulationtechniques.<br>▪ **Unions :**<br>1. Write a program to define and initialize a union for a variable with attributes such as integer,float, and character.<br>2. Write a program to read and display the values of a union variable with multiple data types.<br>3. Write a program to define a union for a currency with attributes such as dollars, euros, and yen.<br>4. Write a program to read and display the values of a union variable with different currencies.<br>5. Write a program to perform arithmetic operations on a union variable with multiple data type.<br>▪ **Pointers:**<br>1. Write a program that uses pointers to swap two integer variables.<br>2. Write a program that uses pointers to find the largest and smallest numbers in an array of integers.<br>3. Write a program that uses pointers to find the length of a string.<br>4. Write a program that uses pointers to concatenate two strings.<br>5. Write a program that uses pointers to reverse a string.<br>6. Write a program that uses pointers to find the sum and average of an array of floating-point numbers.<br>7. Write a program that uses pointers to count the number of vowels in a string.<br>8. Write a program that uses pointers to sort an array of integers in ascending order.<br>9. Write a program that uses pointers to implement a stack data structure.<br>10. Write a program that uses pointers to implement a queue data structure.<br>11. Write a program that uses pointers to implement a linked list data structure.<br>12. Write a program that uses pointers to create a dynamic array. | |
| **Exercise –IV :File Management in C and Dynamic Memory Allocation** | **Hours** |
| ▪ **File Management in C**<br>1. Write a program to read data from a file and display it on the screen.<br>2. Write a program to write data to a file.<br>3. Write a program to read a binary file and display its contents on the screen.<br>4. Write a program to write data to a binary file.<br>5. Write a program to append data to an existing file.<br>6. Write a program to copy the contents of one file to another.<br>7. Write a program to count the number of words in a file.<br>8. Write a program to search for a specific string in a file.<br>9. Write a program to encrypt and decrypt a file.<br>▪ **Dynamic Memory Allocation :** | **9 Hrs** |

| Course Content | Hours |
|---|---|
| 1. Write a program to allocate memory dynamically for an integer variable and display itsaddress and value.<br>2. Write a program to allocate memory dynamically for a character variable and display itsaddress and value.<br>3. Write a program to allocate memory dynamically for an array of integers and display its elements.<br>4. Write a program to allocate memory dynamically for an array of characters and displayits elements.<br>5. Write a program to allocate memory dynamically for a structure and display itselements.<br>6. Write a program to allocate memory dynamically for an array of structures and displaytheir elements.<br>7. Write a program to allocate memory dynamically for a 2D array and display its elements.<br>8. Write a program to allocate memory dynamically for a linked list and display itselements.<br>9. Write a program to allocate memory dynamically for a binary tree and display itselements.<br>10. Write a program to allocate memory dynamically for a graph and display its elements. | |

**Text books:**

- C How to Program by Paul Deitel and Harvey Deitel
- Computer Systems: A Programmer's Perspective by Randal E. Bryant and David R.O'Hallaron

**Reference books:**

- The C Programming Language by Brian Kernighan and Dennis Ritchie
- C Programming: A Modern Approach by K.N. King
- C Primer Plus by Stephen Prata
- Expert C Programming: Deep C Secrets" by Peter van der Linden
- Head First C by David Griffiths and Dawn Griffiths

**Pedagogic tools:**

- Chalk and Board
- Power point presentation
- Seminar
- Videos
- Flipped classroom approach
- Peer instruction
- Collaborative learning
- Problem-based learning
- Lectures and demonstrations

**Methods of Assessment & Tools:**
Components of CIA: 40 marks

| Sr. No. | Component | Content | Duration (if any) | Marks | Sub Total |
|---|---|---|---|---|---|
| **A** | Practical Skill Assessment | All Experiments | 3 hours | 30 (Set for 30) | 30 |
| **B** | Observation and Book of Records | All Experiments | - | 10 (10 marks) | 10 |
| | | | | **Grand Total** | **40** |
| **Lab Methods:** | • Hands-on exercises<br>• Programming assignments<br>• Demonstrations<br>• Code reviews | | | | |
| **Assessment Tools:** | • Self-assessments<br>• Peer assessments<br>• Code documentation | | | | |