# PROGRAMMING WORKSHOP- I
# (C PROGRAMMING)

# LAB MANUAL

# (R24A0586)

# B. TECH II Year - I Sem (R24)

# (2025-2026)


## PREPARED BY

**P.HARIKRISHNA**
**N.PRAMEELA**
**T.SHILPA**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY**
(Autonomous Institution – UGC, Govt. of India)
Sponsored by CMR Educational Society
(Affiliated to JNTU, Hyderabad, Approved by AICTE- Accredited by NBA& NAAC–'A'Grade-ISO9001:2008Certified) Maisammaguda, Dhulapally(PostViaHakimpet), Secunderabad–500100,TelanganaState,India.
Contact Number: 040-23792146/64634237, E-Mail ID: mrcet2004@gmail.com, website: www.mrcet.ac.in

# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

## Vision

To acknowledge quality education and instill high patterns of discipline making the students technologically superior and ethically strong which involves the improvement in the quality of life in human race.

## Mission

To achieve and impart holistic technical education using the best of infrastructure, outstanding technical and teaching expertise to establish the students into competent and confident engineers.

Evolving the center of excellence through creative and innovative teaching learning practices for promoting academic achievement to produce internationally accepted competitive and world class professionals.

## Quality Policy

To pursue continual improvement of teaching learning process of Undergraduate and Post Graduate programs in Engineering & Management vigorously.

To provide state of art infrastructure and expertise to impart the quality education.

# PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

## PEO1 – ANALYTICAL SKILLS :

To facilitate the graduates with the ability to visualize, gather information, articulate, analyze, solve complex problems, and make decisions. These are essential to address the challenges of complex and computation intensive problems increasing their productivity**.**

## PEO2 – TECHNICAL SKILLS :

To facilitate the graduates with the technical skills that prepare them for immediate employment and pursue certification providing a deeper understanding of the technology in advanced areas of computer science and related fields, thus encouraging to pursue higher education and research based on their interest.

## PEO3 – SOFT SKILLS :

To facilitate the graduates with the soft skills that include fulfilling the mission, setting goals, showing self-confidence by communicating effectively, having a positive attitude, get involved in team-work, being a leader, managing their career and their life.

## PEO4 – PROFESSIONAL ETHICS:

To facilitate the graduates with the knowledge of professional and ethical responsibilities by paying attention to grooming, being conservative with style, following dress codes, safety codes, and adapting to technological advancements.

# PROGRAM SPECIFIC OUTCOMES (PSOs)

**1-**Fundamentals and critical knowledge of the Computer System:- Able to Understand the working principles of the computer System and its components , Apply the knowledge to build, asses, and analyze the software and hardware aspects of it.

**2-**The comprehensive and Applicative knowledge of Software Development: Comprehensive skills of Programming Languages, Software process models, methodologies, and able to plan, develop, test, analyze, and manage the software and hardware intensive systems in heterogeneous platforms individually or working in teams.

**3-**Applications of Computing Domain & Research: Able to use the professional, managerial, interdisciplinary skill set, and domain specific tools in development processes, identify the research gaps, and provide innovative solutions to them.

# PROGRAM OUTCOMES (POs)

1-Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2-Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3-Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4-Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5-Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6-The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7-Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8-Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9-Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10-Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11-Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12-Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

# MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY

**Maisammaguda, Dhulapally Post, Via Hakimpet, Secunderabad – 500100**

---

## Department of Computer Science & Engineering

### GENERAL LABORATORY INSTRUCTIONS

1.Students are advised to come to the laboratory at least 5 minutes before (to the starting time), those who come after 5 minutes will not be allowed into the lab.

2. Plan your task properly much before to the commencement, come prepared to the labwith the synopsis / program / experiment details.

3. Student should enter into the laboratory with:
a. Laboratory observation notes with all the details (Problem statement, Aim, Algorithm, Procedure, Program, Expected Output, etc.,) filled in for the lab session.
b. Laboratory Record updated up to the last session experiments and other utensils (if any) needed in the lab.
c. Proper Dress code and Identity card.

4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allotted to you by the faculty.

5. Execute your task in the laboratory, and record the results / output in the lab observation note book, and get certified by the concerned faculty.

6. All the students should be polite and cooperative with the laboratory staff, must maintain the discipline and decency in the laboratory.

7. Computer labs are established with sophisticated and high end branded systems, which should be utilized properly.

8. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions. Misuse of the equipment, misbehaviors with the staff and systems etc., will attract severe punishment.

9. Students must take the permission of the faculty in case of any urgency to go out; if anybody found loitering outside the lab / class without permission during workinghours will be treated seriously and punished appropriately.

10. Students should LOG OFF/ SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects. He/she must ensure the system / seat is kept properly.

**HEAD OF THE DEPARTMENT** **PRINCIPAL**

# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

**SYLLABUS**

**II Year B.Tech CSE-I Sem** **L/T/P/C**
**0/0/2/1**

## (R24A0586) PROGRAMMING WORKSHOP- I (C PROGRAMMING)

### Course Objectives:

1. To write basic C programs using input output statements, datatypes and operators
2. To be able to write programs using conditional statements
3. To decompose a problem into functions and to develop modular, reusable code.
4. To understand the usage of derived datatypes - arrays, pointers and strings ,
5. To be able to use user defined data types like structures and unions
6. To understand need of files in programming

### List of Experiments

| S. No | List of Experiments |
|---|---|
| WEEK 1 | Programs using input and output statements. |
| WEEK 2 | Programs on Operators |
| WEEK 3 | Programs on Conditional Branching Statements. |
| WEEK 4-5 | Programs on Loop statements (while, for, do while loop)<br>Programs on Unconditional Branching statements. |
| WEEK 6 | Programs on Functions |
| WEEK 7-8 | Programs on One Dimensional Arrays.<br>Programs on Two Dimensional Arrays. |
| WEEK 9 | Implementation of String functions with and without built-in functions. |
| WEEK 10-11 | Programs on User defined types (Structure, union and enum) |
| WEEK 12 | Programs on Pointers (DMA functions). |
| WEEK 13-14 | Programs on files. |

### Course Outcomes:

At the end of this course, the student would be able to
1. Write simple programs using input and output functions
2. Design programs on decision and control constructs.
3. Develop programs on code reusability using functions.
4. Implement various concepts of arrays and strings and pointers
5. Create programs using user defined datatypes
6. Implement various concepts of files.

# INDEX

| | | |
|---|---|---|
| | 5. Write a C program with a recursive function to calculate the factorial of a given positive integer.<br>6. Write a recursive function to generate Fibonacci sequence upto n terms? | |
| WEEK 7 | 1. Program to find the sum of all elements in a one-dimensional array.<br>2. Program to find the average of elements in a one-dimensional array.<br>3. Program to find the largest and smallest element in an array.<br>4. Program to sort the elements of the array<br>5. Program to count the frequency of occurrence of a given element in an array | 40 |
| WEEK 8 | 1. Program to read and print elements of a two-dimensional array (matrix).<br>2. Program to find the sum of all elements in a two-dimensional array (matrix).<br>3. Program to add two matrices and display the resulting matrix.<br>4. Program to multiply two matrices and display the resulting matrix<br>5. Program to find the transpose of a given matrix | 45 |
| WEEK 9 | Implement the following using built-in functions and without using built-in functions on strings<br>1. Program to find the length of a string<br>2. Program to copy one string to another string<br>3. Program to concatenate two strings<br>4. Program to compare two strings<br>5. Program that returns the index of a substring in given string<br>6. Program to insert a substring into a string at given position<br>7. Program to delete first occurrence of a substring from a string | 50 |
| WEEK10 | 1. Program to create and display student details using structures<br>2. Program to create and display employee details using structures.<br>3. Program to represent time using structure<br>4. Program to create a bank account and write functions to perform deposit and withdraw operations | 56 |
| WEEK11 | 1. Program to demonstrate the use of union by assigning and printing different data types<br>2. Program to define and use an enum for days of the week**.** | 61 |
| WEEK 12 | 1. Program to dynamically allocate memory using malloc(), store and display elements, then free the memory<br>2. Program to allocate and initialize memory using calloc() and display the elements.<br>3. Program to reallocate memory using realloc() and modify the array size and contents. | 65 |
| WEEK 13 | 1. Program to write text to a file<br>2. Program to read contents from a file character by character<br>3. Program to append Text to a File | 70 |
| WEEK 14 | 1. Program to copy contents from one file to another file<br>2. Program to merge two files to third file<br>3. Program to count the number of occurrences of a string in a text file | 74 |

**WEEK – 1: <u>Programs Using Input and Output Statements</u>**

**Write a C Program to Add Two Numbers**

```
#include <stdio.h>
int main()
{
   int a, b, sum;
   printf("Enter two numbers: ");
   scanf("%d %d", &a, &b);
   sum = a + b;
   printf("Sum = %d\n", sum);
   return 0;
}
```

**OUTPUT:** Enter two numbers: 10 20
Sum = 30

## Exercises:

1. **Write a C program to find Area of a Circle**
2. **Write a C program to calculate the area of a triangle.**
3. **Write a C program to calculate the Simple Interest given the principal, rate of interest, and time.**
4. **Write a simple C program to calculate the Compound Interest using the principal, rate of interest, and time.**
5. **Write a C program to swap two numbers using a temporary variable**
6. **Write a C program to swap two numbers without using a temporary variable.**

**Signature of the Faculty**

**WEEK-2: Programs on Operators**

**Write a C program to show how arithmetic operators work.**

```
#include <stdio.h>
int main()
{
   int a, b;
   printf("Enter two integers: ");
   scanf("%d %d", &a, &b);
   printf("Addition: %d + %d = %d\n", a, b, a + b);
   printf("Subtraction: %d - %d = %d\n", a, b, a - b);
   printf("Multiplication: %d * %d = %d\n", a, b, a * b);
   printf("Division: %d / %d = %d\n", a, b, a / b);
   printf("Modulus: %d %% %d = %d\n", a, b, a % b);
   return 0;
}
```

**OUTPUT:**

```
Enter two integers: 15 4
Addition: 15 + 4 = 19
Subtraction: 15 - 4 = 11
Multiplication: 15 * 4 = 60
Division: 15 / 4 = 3
Modulus: 15 % 4 = 3
```

## Exercises:

1. **Write a C Program to show usage of relational, logical and assignment operators.**
2. **Write a C Program to show how Increment and Decrement Operators work (++, --)**
3. **Write a C Program to perform Bitwise Operations on two numbers**
4. **Write a C Program to show operator precedence in evaluation of mathematical expressions**

**Signature of the Faculty**

**WEEK 3 : <u>Programs on Conditional Branching Statements</u>**

**Write a C program to check if a given number is positive**.

```
#include <stdio.h>
int main()
 {
   int num;
   printf("Enter a number: ");
   scanf("%d", &num);
   if (num > 0)
  {
      printf("%d is positive.\n", num);
   }
   return 0;
}
```

**OUTPUT:**

```
Enter a number: 7
7 is positive.
```

## Exercises:

1. **Write a C program to check whether a given integer is even or odd**.
2. **Write a C program to find the largest of two numbers?**
3. **Write a C program to find the largest of three numbers entered by the user.**
4. **Write a C program to check whether given year is a leap year or not?**
5. **Write a C program to print the day of the week based on a number (1 to 7) entered by the user using a switch statement.**

**Signature of the Faculty**

**WEEK 4: <u>Programs on Loop statements (while, for, do while loop)</u>**

**Write a C program using a while loop to print all numbers from 1 to n, where n is input by the user.**

```
#include <stdio.h>
int main()
{
   int n, i = 1;
   printf("Enter a positive integer: ");
   scanf("%d", &n);
   while (i <= n)
  {
      printf("%d ", i);
      i++;
   }
   printf("\n");
   return 0;
}
```

**OUTPUT:** Enter a positive integer: 5
1 2 3 4 5

**Exercises:**

1. **Write a C program using a for loop to calculate and display the sum of the first n natural numbers.**
2. **Write a program to calculate the factorial of given number**
3. **Write a program to generate Fibonacci sequence upto n terms**
4. **Write a C program using a do-while loop to print numbers from n to 1(descending order), where n is entered by the user.**
5. **Write a C program to print the following patterns upto n rows using for loop**

**ii.**
```
*
* *
* * *
```

**ii.**
```
1
1 2
1 2 3
```

**iii.**
```
 1
 2 2
 3 3 3
```

**Signature of the Faculty**

**WEEK5: <u>Programs on Unconditional Branching statements</u>**

In C, **unconditional branching statements** are mainly:

- goto
- break
- continue

**Write a C program that uses a goto statement to print an error message if the user enters a non-positive number.**

```
#include <stdio.h>
int main()
 {
   int num;
   printf("Enter a positive number: ");
   scanf("%d", &num);
   if (num <= 0)
  {
      goto error;
   }
   printf("You entered: %d\n", num);
   return 0;
error:
   printf("Error: Number must be positive.\n");
   return 1;
}
```

**OUTPUT:**

```
Enter a positive number: 10
You entered: 10
```

**Exercises:**

1. **Write a C program using** a **for loop that stops printing numbers when it reaches 5 using the break statement.**
2. **Write a C program using a for loop that skips printing even numbers using the continue statement.**
3. **Write a C program that accepts a set of numbers from keyboard and finds the sum of positive numbers only. Input contains positive and negative numbers. User input should end when number is zero.**

**Signature of the Faculty**

**WEEK 6: Programs on Functions.**

**Write a C program with a function to add two numbers and return the result.**

```
#include <stdio.h>
int add(int a, int b);
int main()
 {
   int x, y, sum;
   printf("Enter two numbers: ");
   scanf("%d %d", &x, &y);
   sum = add(x, y);  // Function call
   printf("Sum = %d\n", sum);
   return 0;
}
int add(int a, int b)
 {
   return a + b;
}
```

**OUTPUT:**

```
Enter two numbers: 5 7
Sum = 12
```

**Exercises:**

**1.Write a function that returns the GCD of any two numbers?**
**2.Write a C Program to swap 2 numbers using call by value.**
**3. Write a C Program to swap 2 numbers using call by reference.**
**4.Write a C program with a recursive function to calculate the factorial of a given positive integer.**
**5.Write a recursive function to generate Fibonacci sequence upto n terms?**

**Signature of the Faculty**

## WEEK 7: Programs on One Dimensional Arrays

**Write a C program to read and print elements of a one-dimensional array.**

```
#include <stdio.h>
int main()
{
   int n;
   printf("Enter number of elements: ");
   scanf("%d", &n);
   int arr[n];
   printf("Enter %d elements:\n", n);
   for (int i = 0; i < n; i++) {
      scanf("%d", &arr[i]);
   }
   printf("You entered: ");
   for (int i = 0; i < n; i++) {
      printf("%d ", arr[i]);
   }
   printf("\n");
   return 0;
}
```

**OUTPUT:**

```
Enter number of elements: 5
Enter 5 elements:
10 20 30 40 50
You entered: 10 20 30 40 50
```

**Exercises:**

1. **Write a C program to find the sum of all elements in a one-dimensional array.**
2. **Write a C program to find the average of elements in a one-dimensional array.**
3. **Write a C program to find the largest and smallest element in an array.**
4. **Write a C program to sort the elements of the array**
5. **Write a C program to count the frequency of occurrence of a given element in an array**

**Signature of the Faculty**

## WEEK 8 : Programs on Two Dimensional Arrays

**Write a C program** to **read and print elements of a two-dimensional array (matrix).**

```
#include <stdio.h>
int main()
{
   int rows, cols;
   printf("Enter number of rows and columns: ");
   scanf("%d %d", &rows, &cols);
   int matrix[rows][cols];
   printf("Enter elements of the matrix:\n");
   for (int i = 0; i < rows; i++)
  {
      for (int j = 0; j < cols; j++)
     {
         scanf("%d", &matrix[i][j]);
      }
   }
   printf("The matrix is:\n");
   for (int i = 0; i < rows; i++)
   {
      for (int j = 0; j < cols; j++)
      {
         printf("%d ", matrix[i][j]);
      }
      printf("\n");
   }
   return 0;
}
```

**OUTPUT:**

```
Enter number of rows and columns: 2 3
Enter elements of the matrix:
1 2 3
4 5 6
The matrix is:
1 2 3
4 5 6
```

**Exercises:**

1. **Write a C program to find the sum of all elements in a two-dimensional array (matrix).**
2. **Write a C program to add two matrices and display the resulting matrix.**
3. **Write a C program to multiply two matrices and display the resulting matrix**
4. **Write a C program to find the transpose of a given matrix**

**Signature of the Faculty**

## WEEK 9: Implementation of String functions with and without built-in functions

**Write a C program to find the length of a string using built-in function strlen().**

```
#include <stdio.h>
#include <string.h>
int main()
{
   char str[100];
   printf("Enter a string: ");
   fgets(str, sizeof(str), stdin);
   str[strcspn(str, "\n")] = '\0';
   printf("Length of the string is %lu\n", strlen(str));
   return 0;
}
```

Enter a string: Hello
Length of the string is 5
Without built-in function

**Write a C program to find the length of a string without using built-in functions.**

```
#include <stdio.h>
int stringLength(char str[])
{
   int length = 0;
   while (str[length] != '\0')
   {
      length++;
   }
   return length;
}
int main()
{
   char str[100];
   printf("Enter a string: ");
   fgets(str, sizeof(str), stdin);
   int len = 0;
   while (str[len] != '\n' && str[len] != '\0') len++;
   str[len] = '\0';
   printf("Length of the string is %d\n", stringLength(str));
   return 0;
}
```

**OUTPUT**:

Enter a string: MRCET
Length of the string is 5

**Exercises:**

1. **Write a C program to copy one string to another string**
2. **Write a C program to concatenate two strings**
3. **Write a C program to compare two strings**
4. **Write a C program that returns the index of a substring in given string**
5. **Write a C program to insert a substring into a string at given position**
6. **Write a C program to delete first occurrence of a substring from a string**

**Signature of the Faculty**

## WEEK 10: Programs on User defined type Structure

**Structure**
A struct groups related variables under one name.

**Write a C program to create and display student details using structures.**

```
#include <stdio.h>
// Define a structure to hold student data
struct Student
{
   int id;
   char name[50];
   float marks;
};
int main()
 {
   struct Student s1;
   printf("Enter student ID: ");
   scanf("%d", &s1.id);
   printf("Enter student name: ");
   scanf(" %[^\n]", s1.name);  // To read string with spaces
   printf("Enter marks: ");
   scanf("%f", &s1.marks);
   printf("\nStudent Details:\n");
   printf("ID: %d\nName: %s\nMarks: %.2f\n", s1.id, s1.name, s1.marks);
   return 0;
}
```

**OUTPUT:**

```
Enter student ID: 101
Enter student name: Krishna
Enter marks: 87.5

Student Details:
ID: 101
Name: Krishna
Marks: 87.50
```

**Exercises:**

1. **Write a C program to create and display employee details using structures.**
2. **Write a C program to represent time using structure**
3. **Write a C program to create a bank account and write functions to perform deposit and withdraw operations**

**Signature of the Faculty**

## WEEK 11: Programs on User defined types - Union and enum

### Union

A union shares the same memory location for all members — only one member can hold a value at a time.

**Write a C program to demonstrate the use of union by assigning and printing different data types.**

```
#include <stdio.h>
union Data
{
   int i;
   float f;
   char str[20];
};

int main() {
   union Data data;
   data.i = 10;
   printf("data.i = %d\n", data.i);
   data.f = 220.5;
   printf("data.f = %.2f\n", data.f);
   // Assigning to str will overwrite previous values
   // So printing i or f now gives unpredictable results
   strcpy(data.str, "Hello");
   printf("data.str = %s\n", data.str);
   return 0;
}
```

**OUTPUT:**

```
data.i = 10
data.f = 220.50
data.str = Hello
```

**Enum**

An enum defines named integer constants

**Write a C program to define and use an enum for days of the week.**

```
#include <stdio.h>
// Define enum for days of the week
enum Day { Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday };

int main()
{
   enum Day today;
   today = Wednesday;
   printf("Day number: %d\n", today);
   if (today == Wednesday) {
      printf("It's Wednesday, mid-week!\n");
   }
   return 0;
}
```

**OUTPUT:**

```
Day number: 3
It's Wednesday, mid-week!
```

**Exercises:**

1. **Write a C program to show usage of union data type**
2. **Write a C program to implement enum data type**

**Signature of the Faculty**

## WEEK 12: Programs on Pointers (DMA functions)

**Write a C program to demonstrate pointer usage by accessing and printing variable values and addresses.**

```
#include <stdio.h>
int main() {
   int var = 10;
   int *ptr;
   ptr = &var;  // pointer stores address of var
   printf("Value of var = %d\n", var);
   printf("Value via pointer = %d\n", *ptr);
   printf("Address of var = %p\n", &var);
   printf("Pointer points to address = %p\n", ptr);
   return 0;
}
```

**OUTPUT:**

```
Value of var = 10
Value via pointer = 10
Address of var = 0x7ffc1234abcd
Pointer points to address = 0x7ffc1234abcd
```

**Dynamic Memory Allocation using malloc()**

Allocate memory for an integer array, read values, print them, and free the memory.

**Write a C program to dynamically allocate memory using malloc(), store and display elements, then free the memory.**

```
#include <stdio.h>
#include <stdlib.h>
int main() {
   int *arr;
   int n;
   printf("Enter number of elements: ");
   scanf("%d", &n);
   arr = (int *)malloc(n * sizeof(int));  // Allocate memory
   if (arr == NULL) {
      printf("Memory allocation failed.\n");
      return 1;
   }
   printf("Enter %d elements:\n", n);
   for (int i = 0; i < n; i++) {
      scanf("%d", &arr[i]);
   }
   printf("You entered:\n");
   for (int i = 0; i < n; i++) {
      printf("%d ", arr[i]);
   }
   printf("\n");
   free(arr);  // Free allocated memory
   return 0;
}
```

**OUTPUT:**

```
Enter number of elements: 5
Enter 5 elements:
10 20 30 40 50
You entered:
10 20 30 40 50
```

**Exercises:**

1. **Write a C program to allocate and initialize memory using calloc() and display the elements.**
2. **Write a C program to reallocate memory using realloc() and modify the array size and contents.**

**Signature of the Faculty**

## WEEK 13: Programs on files

**Write a C program to write text into a file.**

```
#include <stdio.h>
int main()
{
   FILE *fp;
   fp = fopen("example.txt", "w");
   if (fp == NULL)
  {
      printf("Error opening file!\n");
      return 1;
   }
   fprintf(fp, "Hello, this is a sample text.\n");
   fprintf(fp, "Writing to a file in C.\n");
   fclose(fp);
   printf("Data written to file successfully.\n");
   return 0;
}
```

**OUTPUT:**

Data written to file successfully.

**Exercises:**

1. **Write a C program to read contents from a file character by character**
2. W**rite a C program to append Text to a File**

**Signature of the Faculty**

## WEEK 14: Programs on files

**Write a C program to copy contents from one file to another using file**

```
#include <stdio.h>
int main()
{
   FILE *src, *dest;
   char ch;
   src = fopen("example.txt", "r");
   if (src == NULL)
   {
      printf("Error opening source file!\n");
      return 1;
   }
   dest = fopen("copy.txt", "w");
   if (dest == NULL)
   {
      printf("Error opening destination file!\n");
      fclose(src);
      return 1;
   }
   while ((ch = fgetc(src)) != EOF) {
      fputc(ch, dest);
   }
   fclose(src);
   fclose(dest);
   printf("File copied successfully.\n");
   return 0;
}
```

**OUTPUT:**
File copied successfully

**Exercises:**

1. **Write a C program to merge two files into a third file**
2. **Write a C program to count the number of occurrences of a string in a text file**

**Signature of the Faculty**