here's a **complete list of important C programs**, grouped by **category and level** — from beginner to advanced — so you can learn and practice step-by-step.

Basic C Programs

1. Hello World program
2. Print your name
3. Add two numbers
4. Subtract two numbers
5. Multiply two numbers
6. Divide two numbers
7. Swap two numbers (with and without third variable)
8. Find ASCII value of a character
9. Check even or odd number
10. Find largest of two numbers
11. Find largest of three numbers
12. Check leap year
13. Check positive, negative, or zero
14. Convert Celsius to Fahrenheit
15. Calculate simple interest

Control Statements (if, switch, loop)

16. Find factorial of a number
17. Check palindrome number
18. Check Armstrong number
19. Reverse a number
20. Print multiplication table
21. Print Fibonacci series
22. Sum of digits of a number

23. Find GCD and LCM of two numbers
24. Count digits in a number
25. Power of a number
26. Check prime number
27. Print all prime numbers between two intervals
28. Check strong number
29. Find HCF and LCM
30. Menu-driven calculator using switch

Pattern Printing Programs

31. Half pyramid using *
32. Inverted pyramid using *
33. Full pyramid using *
34. Floyd's triangle
35. Pascal's triangle
36. Number pyramid
37. Alphabet pyramid
38. Diamond pattern

Array Programs

39. Read and print elements of array
40. Find sum and average of array elements
41. Find maximum and minimum in array
42. Reverse an array
43. Search element in array (linear search)
44. Binary search
45. Sort array (ascending and descending)
46. Merge two arrays

47. Count even and odd numbers in array
48. Copy one array to another

String Programs

49. Find length of string (without using strlen())
50. Reverse a string
51. Check palindrome string
52. Compare two strings
53. Concatenate two strings
54. Count vowels, consonants, digits, and spaces
55. Convert string to uppercase and lowercase
56. Find frequency of characters in string

Functions and Recursion

57. Factorial using recursion
58. Fibonacci series using recursion
59. GCD using recursion
60. Sum of digits using recursion
61. Reverse number using recursion
62. Power of a number using recursion

Pointers

63. Pointer basic example
64. Swap two numbers using pointers
65. Find length of string using pointer
66. Array sum using pointers
67. Call by reference and call by value

Structures and Unions

68. Structure to store student details
69. Add two complex numbers using structure
70. Structure with array
71. Difference between structure and union
72. Nested structure example

File Handling

73. Write to a file
74. Read from a file
75. Copy contents from one file to another
76. Count number of words, lines, and characters in file

Miscellaneous

77. Matrix addition
78. Matrix subtraction
79. Matrix multiplication
80. Transpose of a matrix
81. Store and print student marks using structure and array
82. Dynamic memory allocation example (malloc, calloc, free)

Below are the **first 10 basic C programs with answers**

Hello World Program

```
#include <stdio.h>


int main() {
   printf("Hello, World!");
```

```
    return 0;
}
```

Output:

Hello, World!

Explanation:
printf() is used to print text on the screen.

Print Your Name

```
#include <stdio.h>

int main() {
    printf("My name is Sandip Patil");
    return 0;
}
```

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", sum);
    return 0;
}
```

Input: 10 20
Output: Sum = 30

Swap Two Numbers (Using Third Variable)

```
#include <stdio.h>

int main() {
   int a = 5, b = 10, temp;
   temp = a;
   a = b;
   b = temp;
   printf("After swap: a = %d, b = %d", a, b);
   return 0;
}
```

Output:
After swap: a = 10, b = 5

Swap Without Third Variable

```
#include <stdio.h>

int main() {
   int a = 5, b = 10;
   a = a + b;
   b = a - b;
   a = a - b;
   printf("After swap: a = %d, b = %d", a, b);
   return 0;
}
```

Find ASCII Value of a Character

```
#include <stdio.h>
```

```
int main() {
   char ch;
   printf("Enter a character: ");
   scanf("%c", &ch);
   printf("ASCII value of %c = %d", ch, ch);
   return 0;
}
```

Input: A
Output: ASCII value of A = 65

Check Even or Odd

```
#include <stdio.h>

int main() {
   int num;
   printf("Enter a number: ");
   scanf("%d", &num);
   if (num % 2 == 0)
      printf("%d is even.", num);
   else
      printf("%d is odd.", num);
   return 0;
}
```

Find Largest of Two Numbers

```
#include <stdio.h>
```

```
int main() {
   int a, b;
   printf("Enter two numbers: ");
   scanf("%d %d", &a, &b);
   if (a > b)
      printf("%d is larger.", a);
   else
      printf("%d is larger.", b);
   return 0;
}
```

Check Leap Year

```
#include <stdio.h>

int main() {
   int year;
   printf("Enter a year: ");
   scanf("%d", &year);

   if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0))
      printf("%d is a leap year.", year);
   else
      printf("%d is not a leap year.", year);

   return 0;
}
```

Input: 2024
Output: 2024 is a leap year.

Check Positive, Negative, or Zero

```
#include <stdio.h>

int main() {
   int num;
   printf("Enter a number: ");
   scanf("%d", &num);

   if (num > 0)
      printf("Positive");
   else if (num < 0)
      printf("Negative");
   else
      printf("Zero");

   return 0;
}
```

Find Factorial of a Number

```
#include <stdio.h>

int main() {
   int n, i;
   unsigned long long factorial = 1;

   printf("Enter a number: ");
```

```
  scanf("%d", &n);

  if (n < 0)
    printf("Factorial does not exist for negative numbers.");
  else {
    for(i = 1; i <= n; ++i) {
      factorial *= i;
    }
    printf("Factorial of %d = %llu", n, factorial);
  }
  return 0;
}
```

Input: 5
Output: Factorial of 5 = 120

Explanation: Uses for loop to multiply numbers from 1 to n.

Check Palindrome Number

```
#include <stdio.h>

int main() {
  int n, reversed = 0, remainder, original;
  printf("Enter an integer: ");
  scanf("%d", &n);

  original = n;
  while (n != 0) {
    remainder = n % 10;
    reversed = reversed * 10 + remainder;
```

```
    n /= 10;
  }

  if (original == reversed)
    printf("%d is a palindrome.", original);
  else
    printf("%d is not a palindrome.", original);

  return 0;
}
```

Input: 121
Output: 121 is a palindrome.

Check Armstrong Number

```
#include <stdio.h>
#include <math.h>

int main() {
  int num, original, remainder, n = 0;
  float sum = 0;

  printf("Enter an integer: ");
  scanf("%d", &num);

  original = num;

  // Count number of digits
  while (original != 0) {
```

```
    original /= 10;
    ++n;
  }

  original = num;
  while (original != 0) {
    remainder = original % 10;
    sum += pow(remainder, n);
    original /= 10;
  }

  if ((int)sum == num)
    printf("%d is an Armstrong number.", num);
  else
    printf("%d is not an Armstrong number.", num);

  return 0;
}
```

Input: 153
Output: 153 is an Armstrong number.

Reverse a Number

```
#include <stdio.h>

int main() {
  int num, reversed = 0, remainder;
  printf("Enter a number: ");
  scanf("%d", &num);
```

```
  while (num != 0) {
    remainder = num % 10;
    reversed = reversed * 10 + remainder;
    num /= 10;
  }

  printf("Reversed number = %d", reversed);
  return 0;
}
```

Input: 1234
Output: Reversed number = 4321

Print Multiplication Table

```
#include <stdio.h>

int main() {
  int num, i;
  printf("Enter a number: ");
  scanf("%d", &num);

  for(i = 1; i <= 10; ++i) {
    printf("%d x %d = %d\n", num, i, num * i);
  }
  return 0;
}
```

Input: 5
Output:

5 x 1 = 5

5 x 2 = 10

...

5 x 10 = 50

Print Fibonacci Series

```
#include <stdio.h>


int main() {
   int n, t1 = 0, t2 = 1, nextTerm, i;
   printf("Enter number of terms: ");
   scanf("%d", &n);


   printf("Fibonacci Series: ");


   for (i = 1; i <= n; ++i) {
      printf("%d ", t1);
      nextTerm = t1 + t2;
      t1 = t2;
      t2 = nextTerm;
   }


   return 0;
}
```

Input: 7
Output: 0 1 1 2 3 5 8

Check Prime Number

```
#include <stdio.h>
```

```
int main() {
   int n, i, flag = 0;
   printf("Enter a positive integer: ");
   scanf("%d", &n);

   if(n <= 1)
      flag = 1; // Not prime
   else {
      for(i = 2; i <= n/2; ++i) {
         if(n % i == 0) {
            flag = 1;
            break;
         }
      }
   }

   if(flag == 0)
      printf("%d is a prime number.", n);
   else
      printf("%d is not a prime number.", n);

   return 0;
}
```

Input: 13
Output: 13 is a prime number.

Sum of Digits

```c
#include <stdio.h>

int main() {
   int n, sum = 0, remainder;
   printf("Enter a number: ");
   scanf("%d", &n);

   while(n != 0) {
      remainder = n % 10;
      sum += remainder;
      n /= 10;
   }

   printf("Sum of digits = %d", sum);
   return 0;
}
```

Input: 123
Output: Sum of digits = 6

These cover **Control Statements and Loops**.

Let's continue with **Array Programs** (reading, sum, max/min, search, sort, etc.) with **code + explanation + sample output**.

Read and Print Elements of an Array

```c
#include <stdio.h>

int main() {
   int n, i;
   printf("Enter number of elements: ");
```

```
    scanf("%d", &n);

    int arr[n];
    printf("Enter %d elements:\n", n);
    for(i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    printf("Array elements are: ");
    for(i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }

    return 0;
}
```

Input: 5 → 10 20 30 40 50
Output: Array elements are: 10 20 30 40 50

Sum and Average of Array Elements

```
#include <stdio.h>

int main() {
    int n, i, sum = 0;
    float avg;
    printf("Enter number of elements: ");
    scanf("%d", &n);

    int arr[n];
```

```
    printf("Enter %d elements:\n", n);
    for(i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
        sum += arr[i];
    }

    avg = (float)sum / n;
    printf("Sum = %d\nAverage = %.2f", sum, avg);

    return 0;
}
```

Input: 3 → 10 20 30
Output:

Sum = 60

Average = 20.00

Find Maximum and Minimum in Array

```
#include <stdio.h>

int main() {
    int n, i;
    printf("Enter number of elements: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter elements:\n");
    for(i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
```

```
    }

    int max = arr[0];
    int min = arr[0];

    for(i = 1; i < n; i++) {
        if(arr[i] > max) max = arr[i];
        if(arr[i] < min) min = arr[i];
    }

    printf("Maximum = %d\nMinimum = %d", max, min);
    return 0;
}
```

Input: 5 → 12 5 8 20 1
Output:

Maximum = 20

Minimum = 1

Reverse an Array

```
#include <stdio.h>

int main() {
    int n, i;
    printf("Enter number of elements: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter elements:\n");
```

```
    for(i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    printf("Reversed array: ");
    for(i = n-1; i >= 0; i--) {
        printf("%d ", arr[i]);
    }

    return 0;
}
```

Input: 5 → 1 2 3 4 5
Output: Reversed array: 5 4 3 2 1

Linear Search in Array

```
#include <stdio.h>

int main() {
    int n, i, key, flag = 0;
    printf("Enter number of elements: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter elements:\n");
    for(i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
```

```
    printf("Enter element to search: ");
    scanf("%d", &key);

    for(i = 0; i < n; i++) {
        if(arr[i] == key) {
            flag = 1;
            break;
        }
    }

    if(flag)
        printf("%d found at position %d", key, i+1);
    else
        printf("%d not found", key);

    return 0;
}
```

Input: 5 → 10 20 30 40 50 → search 30
Output: 30 found at position 3

Binary Search (Array Must Be Sorted)

```
#include <stdio.h>

int main() {
    int n, i, key, low, high, mid;
    printf("Enter number of elements: ");
    scanf("%d", &n);
```

```
int arr[n];
printf("Enter sorted elements:\n");
for(i = 0; i < n; i++) {
   scanf("%d", &arr[i]);
}

printf("Enter element to search: ");
scanf("%d", &key);

low = 0;
high = n - 1;
int found = 0;

while(low <= high) {
   mid = (low + high) / 2;
   if(arr[mid] == key) {
      found = 1;
      break;
   } else if(arr[mid] < key)
      low = mid + 1;
   else
      high = mid - 1;
}

if(found)
   printf("%d found at position %d", key, mid+1);
else
   printf("%d not found", key);
```

```
    return 0;
}
```

Input: 5 → 10 20 30 40 50 → search 40
Output: 40 found at position 4

Sort Array in Ascending Order

```
#include <stdio.h>

int main() {
    int n, i, j, temp;
    printf("Enter number of elements: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter elements:\n");
    for(i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    // Bubble Sort
    for(i = 0; i < n-1; i++) {
        for(j = 0; j < n-i-1; j++) {
            if(arr[j] > arr[j+1]) {
                temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
```

```
    }
  }

  printf("Sorted array: ");
  for(i = 0; i < n; i++) {
    printf("%d ", arr[i]);
  }

  return 0;
}
```

Input: 5 → 50 20 40 10 30
Output: Sorted array: 10 20 30 40 50

We've now covered **basic array programs**.

Next, we can move to **String Programs** (length, reverse, palindrome, concatenate, etc.) with full code + explanation.

Find Length of a String (Without strlen())

```
#include <stdio.h>

int main() {
  char str[100];
  int length = 0;

  printf("Enter a string: ");
  gets(str); // or use fgets(str, sizeof(str), stdin);

  while(str[length] != '\0') {
    length++;
```

```
    }

    printf("Length of string = %d", length);
    return 0;
}
```

Input: Hello
Output: Length of string = 5

Reverse a String

```
#include <stdio.h>
#include <string.h>

int main() {
    char str[100], temp;
    int i, j;

    printf("Enter a string: ");
    gets(str);

    i = 0;
    j = strlen(str) - 1;

    while(i < j) {
        temp = str[i];
        str[i] = str[j];
        str[j] = temp;
        i++;
        j--;
```

```
    }

    printf("Reversed string: %s", str);
    return 0;
}
```

Input: Hello
Output: Reversed string: olleH

Check Palindrome String

```
#include <stdio.h>
#include <string.h>

int main() {
    char str[100];
    int i, len, flag = 0;

    printf("Enter a string: ");
    gets(str);

    len = strlen(str);
    for(i = 0; i < len/2; i++) {
        if(str[i] != str[len-i-1]) {
            flag = 1;
            break;
        }
    }

    if(flag == 0)
```

```
    printf("%s is a palindrome.", str);
  else
    printf("%s is not a palindrome.", str);

  return 0;
}
```

Input: madam
Output: madam is a palindrome.

Compare Two Strings

```
#include <stdio.h>
#include <string.h>

int main() {
  char str1[100], str2[100];

  printf("Enter first string: ");
  gets(str1);
  printf("Enter second string: ");
  gets(str2);

  if(strcmp(str1, str2) == 0)
    printf("Strings are equal.");
  else
    printf("Strings are not equal.");

  return 0;
```

```
}
```

Input: Hello & Hello
Output: Strings are equal.

Concatenate Two Strings

```
#include <stdio.h>

#include <string.h>


int main() {
   char str1[100], str2[100];


   printf("Enter first string: ");
   gets(str1);
   printf("Enter second string: ");
   gets(str2);


   strcat(str1, str2);
   printf("Concatenated string: %s", str1);


   return 0;
}
```

Input: Hello & World
Output: Concatenated string: HelloWorld

Count Vowels, Consonants, Digits, and Spaces

```
#include <stdio.h>

#include <ctype.h>


int main() {
```

```
  char str[100];
  int i, vowels=0, consonants=0, digits=0, spaces=0;

  printf("Enter a string: ");
  gets(str);

  for(i = 0; str[i] != '\0'; i++) {
    char ch = str[i];
    if(isalpha(ch)) {
      ch = tolower(ch);
      if(ch=='a'||ch=='e'||ch=='i'||ch=='o'||ch=='u')
        vowels++;
      else
        consonants++;
    } else if(isdigit(ch)) {
      digits++;
    } else if(ch==' ') {
      spaces++;
    }
  }

  printf("Vowels = %d\nConsonants = %d\nDigits = %d\nSpaces = %d", vowels,
consonants, digits, spaces);
  return 0;
}
```

Input: Hello World 123
Output:

Vowels = 3

Consonants = 7

Digits = 3

Spaces = 2

Convert String to Uppercase and Lowercase

```
#include <stdio.h>
#include <ctype.h>

int main() {
   char str[100];
   int i;

   printf("Enter a string: ");
   gets(str);

   // Uppercase
   for(i = 0; str[i] != '\0'; i++)
      str[i] = toupper(str[i]);
   printf("Uppercase: %s\n", str);

   // Lowercase
   for(i = 0; str[i] != '\0'; i++)
      str[i] = tolower(str[i]);
   printf("Lowercase: %s", str);

   return 0;
}
```

Input: Hello World
Output:

Uppercase: HELLO WORLD

Lowercase: hello world

That covers the **most important string programs**.

Next, we can move to **Functions & Recursion Programs** (factorial, Fibonacci, GCD, sum of digits, reverse number using recursion).

Factorial Using Recursion

```
#include <stdio.h>

int factorial(int n) {
   if(n == 0 || n == 1)
      return 1;
   else
      return n * factorial(n-1);
}

int main() {
   int num;
   printf("Enter a number: ");
   scanf("%d", &num);

   printf("Factorial of %d = %d", num, factorial(num));
   return 0;
}
```

Input: 5
Output: Factorial of 5 = 120

Explanation: Function calls itself until n reaches 1.

Fibonacci Series Using Recursion

```
#include <stdio.h>

int fibonacci(int n) {
   if(n == 0)
      return 0;
   else if(n == 1)
      return 1;
   else
      return fibonacci(n-1) + fibonacci(n-2);
}

int main() {
   int n, i;
   printf("Enter number of terms: ");
   scanf("%d", &n);

   printf("Fibonacci Series: ");
   for(i = 0; i < n; i++) {
      printf("%d ", fibonacci(i));
   }

   return 0;
}
```

Input: 7
Output: 0 1 1 2 3 5 8

GCD Using Recursion (Euclidean Algorithm)

```
#include <stdio.h>

int gcd(int a, int b) {
   if(b == 0)
      return a;
   else
      return gcd(b, a % b);
}

int main() {
   int a, b;
   printf("Enter two numbers: ");
   scanf("%d %d", &a, &b);

   printf("GCD of %d and %d = %d", a, b, gcd(a, b));
   return 0;
}
```

Input: 24 36
Output: GCD of 24 and 36 = 12

Sum of Digits Using Recursion

```
#include <stdio.h>

int sumDigits(int n) {
   if(n == 0)
      return 0;
   else
      return (n % 10) + sumDigits(n / 10);
```

```
}

int main() {
   int num;
   printf("Enter a number: ");
   scanf("%d", &num);

   printf("Sum of digits of %d = %d", num, sumDigits(num));
   return 0;
}
```

Input: 1234
Output: Sum of digits of 1234 = 10

Reverse Number Using Recursion

```
#include <stdio.h>

void reverseNumber(int n) {
   if(n < 10) {
      printf("%d", n);
      return;
   }
   printf("%d", n % 10);
   reverseNumber(n / 10);
}

int main() {
   int num;
   printf("Enter a number: ");
```

```
    scanf("%d", &num);

    printf("Reversed number: ");
    reverseNumber(num);
    return 0;
}
```

Input: 1234
Output: Reversed number: 4321

Power of a Number Using Recursion

```
#include <stdio.h>

int power(int base, int exp) {
    if(exp == 0)
        return 1;
    else
        return base * power(base, exp-1);
}

int main() {
    int base, exp;
    printf("Enter base and exponent: ");
    scanf("%d %d", &base, &exp);

    printf("%d^%d = %d", base, exp, power(base, exp));
    return 0;
}
```

Input: 2 5
Output: 2^5 = 32

These cover the **most common Functions & Recursion programs**.

Next, we can move to **Pointers Programs** (basic pointers, swap using pointers, array sum using pointers, call by reference).

Basic Pointer Example

```
#include <stdio.h>


int main() {
   int a = 10;
   int *ptr = &a; // Pointer stores the address of a


   printf("Value of a = %d\n", a);
   printf("Address of a = %p\n", &a);
   printf("Value stored in ptr = %p\n", ptr);
   printf("Value pointed by ptr = %d\n", *ptr);


   return 0;
}
```

Output (example):

```
Value of a = 10
Address of a = 0x7ffeefbff5ac
Value stored in ptr = 0x7ffeefbff5ac
Value pointed by ptr = 10
```

Explanation: *ptr gives the value at the address stored in the pointer.

Swap Two Numbers Using Pointers

```
#include <stdio.h>
```

```
void swap(int *x, int *y) {
   int temp;
   temp = *x;
   *x = *y;
   *y = temp;
}

int main() {
   int a = 5, b = 10;
   printf("Before swap: a = %d, b = %d\n", a, b);

   swap(&a, &b);

   printf("After swap: a = %d, b = %d", a, b);
   return 0;
}
```

Output:

Before swap: a = 5, b = 10

After swap: a = 10, b = 5

Explanation: Uses call by reference to swap values.

Find Length of String Using Pointer

```
#include <stdio.h>

int main() {
   char str[100];
   char *ptr;
```

```
    int length = 0;

    printf("Enter a string: ");
    gets(str);

    ptr = str;
    while(*ptr != '\0') {
        length++;
        ptr++;
    }

    printf("Length of string = %d", length);
    return 0;
}
```

Input: Hello
Output: Length of string = 5

Array Sum Using Pointer

```
#include <stdio.h>

int main() {
    int arr[5] = {10, 20, 30, 40, 50};
    int *ptr = arr;
    int sum = 0, i;

    for(i = 0; i < 5; i++) {
        sum += *(ptr + i);
    }
```

```
    printf("Sum of array elements = %d", sum);
    return 0;
}
```

Output: Sum of array elements = 150

Call by Reference vs Call by Value

```
#include <stdio.h>

void callByValue(int a) {
    a = a + 10;
}

void callByReference(int *a) {
    *a = *a + 10;
}

int main() {
    int num = 20;

    callByValue(num);
    printf("After call by value: %d\n", num); // num unchanged

    callByReference(&num);
    printf("After call by reference: %d", num); // num changed

    return 0;
}
```

Output:

After call by value: 20

After call by reference: 30

Explanation: Call by reference modifies the original variable.

These cover the **basic and important pointers programs**.

Next, we can move to **Structures and Unions Programs** (student details, complex number addition, nested structure).

Structure to Store Student Details

```
#include <stdio.h>

struct Student {
   char name[50];
   int roll;
   float marks;
};

int main() {
   struct Student s;

   printf("Enter name: ");
   gets(s.name);
   printf("Enter roll number: ");
   scanf("%d", &s.roll);
   printf("Enter marks: ");
   scanf("%f", &s.marks);
```

```
    printf("\nStudent Details:\n");
    printf("Name: %s\n", s.name);
    printf("Roll: %d\n", s.roll);
    printf("Marks: %.2f", s.marks);

    return 0;
}
```

Input: Sandip 101 95.5
Output:

Student Details:

Name: Sandip

Roll: 101

Marks: 95.50

Add Two Complex Numbers Using Structure

```
#include <stdio.h>

struct Complex {
    float real;
    float imag;
};

int main() {
    struct Complex c1, c2, sum;

    printf("Enter first complex number (real imag): ");
    scanf("%f %f", &c1.real, &c1.imag);
```

```
    printf("Enter second complex number (real imag): ");
    scanf("%f %f", &c2.real, &c2.imag);

    sum.real = c1.real + c2.real;
    sum.imag = c1.imag + c2.imag;

    printf("Sum = %.2f + %.2fi", sum.real, sum.imag);
    return 0;
}
```

Input: 2 3 & 4 5
Output: Sum = 6.00 + 8.00i

Structure with Array

```
#include <stdio.h>

struct Student {
    char name[50];
    int marks[3];
};

int main() {
    struct Student s;
    int i, total = 0;

    printf("Enter name: ");
    gets(s.name);

    printf("Enter marks for 3 subjects: ");
```

```
    for(i = 0; i < 3; i++) {
        scanf("%d", &s.marks[i]);
        total += s.marks[i];
    }

    printf("\nStudent: %s\n", s.name);
    printf("Total Marks = %d", total);

    return 0;
}
```

Input: Sandip 80 90 85
Output:

Student: Sandip

Total Marks = 255

Difference Between Structure and Union

```
#include <stdio.h>

struct ExampleStruct {
    int x;
    char y;
};

union ExampleUnion {
    int x;
    char y;
};
```

```
int main() {
   struct ExampleStruct s;
   union ExampleUnion u;

   printf("Size of struct = %zu\n", sizeof(s));
   printf("Size of union = %zu", sizeof(u));

   return 0;
}
```

Output (example on 32-bit system):

Size of struct = 8

Size of union = 4

Explanation: Structure allocates memory for all members; union shares memory among members.

Nested Structure Example

```
#include <stdio.h>

struct Date {
   int day;
   int month;
   int year;
};

struct Student {
   char name[50];
   struct Date dob;
};
```

```
int main() {
   struct Student s;

   printf("Enter name: ");
   gets(s.name);

   printf("Enter date of birth (day month year): ");
   scanf("%d %d %d", &s.dob.day, &s.dob.month, &s.dob.year);

   printf("\nStudent: %s\nDOB: %d/%d/%d", s.name, s.dob.day, s.dob.month,
s.dob.year);

   return 0;
}
```

Input: Sandip 8 10 1995
Output:

Student: Sandip

DOB: 8/10/1995

These cover the **basic and nested structure & union programs**. ✅

Next, we can move to **File Handling Programs** (read/write files, copy files, count

---

Write to a File

```
#include <stdio.h>

int main() {
   FILE *fp;
```

```
    char str[100];

    printf("Enter text to write to file: ");
    gets(str);

    fp = fopen("file.txt", "w"); // Open file in write mode
    if(fp == NULL) {
        printf("Error opening file!");
        return 1;
    }

    fprintf(fp, "%s", str);
    fclose(fp);

    printf("Data written to file successfully.");
    return 0;
}
```

Input: Hello World
Output: Data written to file successfully.
File Content: Hello World

Read from a File

```
#include <stdio.h>

int main() {
    FILE *fp;
    char ch;

    fp = fopen("file.txt", "r"); // Open file in read mode
```

```
    if(fp == NULL) {
        printf("Error opening file!");
        return 1;
    }

    printf("File content:\n");
    while((ch = fgetc(fp)) != EOF) {
        printf("%c", ch);
    }

    fclose(fp);
    return 0;
}
```

Output (for above file):

File content:

Hello World

Copy Contents from One File to Another

```
#include <stdio.h>

int main() {
    FILE *source, *dest;
    char ch;

    source = fopen("file.txt", "r");
    dest = fopen("copy.txt", "w");

    if(source == NULL || dest == NULL) {
```

```
    printf("Error opening file!");
    return 1;
  }

  while((ch = fgetc(source)) != EOF) {
    fputc(ch, dest);
  }

  fclose(source);
  fclose(dest);

  printf("File copied successfully.");
  return 0;
}
```

Output: File copied successfully.
File copy.txt content: Same as file.txt

Count Words, Lines, and Characters in a File

```
#include <stdio.h>
#include <ctype.h>

int main() {
  FILE *fp;
  char ch;
  int lines = 0, words = 0, chars = 0;

  fp = fopen("file.txt", "r");
  if(fp == NULL) {
```

```
    printf("Error opening file!");
    return 1;
  }

  while((ch = fgetc(fp)) != EOF) {
    chars++;
    if(ch == '\n') lines++;
    if(isspace(ch)) words++;
  }

  fclose(fp);
  printf("Lines = %d\nWords = %d\nCharacters = %d", lines + 1, words + 1, chars);
  return 0;
}
```

Input (file content):

Hello World

This is C programming

Output:

Lines = 2

Words = 5

Characters = 32

These cover the **basic file handling programs**.

Matrix Addition

```
#include <stdio.h>

int main() {
```

```
int r, c, i, j;
printf("Enter number of rows and columns: ");
scanf("%d %d", &r, &c);

int A[r][c], B[r][c], sum[r][c];

printf("Enter elements of first matrix:\n");
for(i = 0; i < r; i++)
   for(j = 0; j < c; j++)
      scanf("%d", &A[i][j]);

printf("Enter elements of second matrix:\n");
for(i = 0; i < r; i++)
   for(j = 0; j < c; j++)
      scanf("%d", &B[i][j]);

for(i = 0; i < r; i++)
   for(j = 0; j < c; j++)
      sum[i][j] = A[i][j] + B[i][j];

printf("Sum of matrices:\n");
for(i = 0; i < r; i++) {
   for(j = 0; j < c; j++)
      printf("%d ", sum[i][j]);
   printf("\n");
}

return 0;
```

}

Input:

Matrix A: 1 2 3 4

Matrix B: 5 6 7 8

Output:

6 8

10 12

Matrix Multiplication

```
#include <stdio.h>

int main() {
   int r1, c1, r2, c2, i, j, k;
   printf("Enter rows and columns of first matrix: ");
   scanf("%d %d", &r1, &c1);
   printf("Enter rows and columns of second matrix: ");
   scanf("%d %d", &r2, &c2);

   if(c1 != r2) {
      printf("Matrix multiplication not possible!");
      return 1;
   }

   int A[r1][c1], B[r2][c2], mul[r1][c2];

   printf("Enter elements of first matrix:\n");
   for(i = 0; i < r1; i++)
      for(j = 0; j < c1; j++)
```

```
        scanf("%d", &A[i][j]);

    printf("Enter elements of second matrix:\n");
    for(i = 0; i < r2; i++)
      for(j = 0; j < c2; j++)
        scanf("%d", &B[i][j]);

    for(i = 0; i < r1; i++)
      for(j = 0; j < c2; j++) {
        mul[i][j] = 0;
        for(k = 0; k < c1; k++)
          mul[i][j] += A[i][k] * B[k][j];
      }

    printf("Product of matrices:\n");
    for(i = 0; i < r1; i++) {
      for(j = 0; j < c2; j++)
        printf("%d ", mul[i][j]);
      printf("\n");
    }

    return 0;
}
```

Input Example:

Matrix A: 1 2 3 4 (2x2)

Matrix B: 5 6 7 8 (2x2)

Output:

19 22

Transpose of a Matrix

```
#include <stdio.h>

int main() {
   int r, c, i, j;
   printf("Enter rows and columns: ");
   scanf("%d %d", &r, &c);

   int A[r][c], transpose[c][r];

   printf("Enter elements:\n");
   for(i = 0; i < r; i++)
      for(j = 0; j < c; j++)
         scanf("%d", &A[i][j]);

   for(i = 0; i < r; i++)
      for(j = 0; j < c; j++)
         transpose[j][i] = A[i][j];

   printf("Transpose of matrix:\n");
   for(i = 0; i < c; i++) {
      for(j = 0; j < r; j++)
         printf("%d ", transpose[i][j]);
      printf("\n");
   }
```

```
    return 0;
}
```

Dynamic Memory Allocation (malloc)

```
#include <stdio.h>
#include <stdlib.h>

int main() {
    int n, i;
    int *ptr;

    printf("Enter number of elements: ");
    scanf("%d", &n);

    ptr = (int*)malloc(n * sizeof(int));
    if(ptr == NULL) {
        printf("Memory not allocated.");
        return 1;
    }

    printf("Enter elements:\n");
    for(i = 0; i < n; i++)
        scanf("%d", &ptr[i]);

    printf("Array elements are: ");
    for(i = 0; i < n; i++)
        printf("%d ", ptr[i]);
```

```
    free(ptr); // Free allocated memory
    return 0;
}
```

Input: 5 → 10 20 30 40 50
Output: Array elements are: 10 20 30 40 50

Menu-Driven Calculator

```
#include <stdio.h>

int main() {
    int choice;
    float a, b;

    do {
        printf("\nMenu:\n1.Add\n2.Subtract\n3.Multiply\n4.Divide\n5.Exit\nEnter choice: ");
        scanf("%d", &choice);

        if(choice >= 1 && choice <= 4) {
            printf("Enter two numbers: ");
            scanf("%f %f", &a, &b);
        }

        switch(choice) {
            case 1: printf("Result = %.2f", a+b); break;
            case 2: printf("Result = %.2f", a-b); break;
            case 3: printf("Result = %.2f", a*b); break;
            case 4:
                if(b != 0) printf("Result = %.2f", a/b);
```

```
            else printf("Cannot divide by zero");
            break;
        case 5: printf("Exiting..."); break;
        default: printf("Invalid choice");
    }
  } while(choice != 5);

  return 0;
}
```

C PROGRAMS WITH ANSWERS

BASIC PROGRAMS

1. Hello World

```
#include <stdio.h>
int main() {
printf("Hello, World!");
return 0;
}
```

Output: Hello, World!

2. Print Your Name

```
#include <stdio.h>
int main() {
printf("My name is Sandip Patil");
return 0;
}
```

3. Add Two Numbers

```
#include <stdio.h>
```

```
int main() {
int a, b, sum;
scanf("%d %d", &a, &b);
sum = a + b;
printf("Sum = %d", sum);
return 0;
}
```

Output for 10 20: Sum = 30

...

CONTROL STATEMENTS

11. Factorial

```
#include <stdio.h>
int main() {
int n, i;
unsigned long long factorial = 1;
scanf("%d", &n);
for(i = 1; i <= n; ++i) factorial *= i;
printf("Factorial of %d = %llu", n, factorial);
return 0;
}
```

...

ARRAYS

19. Read and Print Array

```
#include <stdio.h>
int main() {
int n, i;
scanf("%d", &n);
int arr[n];
```

```
for(i=0;i<n;i++) scanf("%d", &arr[i]);
for(i=0;i<n;i++) printf("%d ", arr[i]);
return 0;
}
```

...

STRINGS

26. Length of String

```
#include <stdio.h>
int main(){
char str[100]; int len=0;
gets(str);
while(str[len]!='\0') len++;
printf("Length = %d", len);
return 0;
}
```

...

FUNCTIONS & RECURSION

33. Factorial using Recursion

```
#include <stdio.h>
int factorial(int n){
if(n==0||n==1) return 1;
else return n*factorial(n-1);
}
int main(){
int num;
scanf("%d", &num);
printf("Factorial of %d = %d", num, factorial(num));
return 0;
```

}

...

POINTERS

39. Basic Pointer

```
#include <stdio.h>

int main(){

int a=10; int *ptr=&a;

printf("Value = %d, Address = %p, Pointer Value = %p, Value pointed = %d", a, &a, ptr, *ptr);

return 0;

}
```

...

STRUCTURES & UNIONS

44. Student Details

```
#include <stdio.h>

struct Student { char name[50]; int roll; float marks; };

int main(){

struct Student s;

gets(s.name); scanf("%d %f", &s.roll, &s.marks);

printf("Name:%s\nRoll:%d\nMarks:%.2f", s.name, s.roll, s.marks);

return 0;

}
```

...

FILE HANDLING

49. Write to File

```
#include <stdio.h>

int main(){

FILE *fp;
```

```
char str[100];
gets(str);
fp=fopen("file.txt", "w");
fprintf(fp, "%s", str);
fclose(fp);
return 0;
}
```

...

MISCELLANEOUS

53. Matrix Addition

```
#include <stdio.h>
int main(){
int r,c,i,j;
scanf("%d %d", &r,&c);
int A[r][c],B[r][c],sum[r][c];
for(i=0;i<r;i++) for(j=0;j<c;j++) scanf("%d", &A[i][j]);
for(i=0;i<r;i++) for(j=0;j<c;j++) scanf("%d", &B[i][j]);
for(i=0;i<r;i++) for(j=0;j<c;j++) sum[i][j]=A[i][j]+B[i][j];
for(i=0;i<r;i++){for(j=0;j<c;j++) printf("%d ", sum[i][j]); printf("\n");}
return 0;
}
```

...
