# ✅ Day : Basic Input/Output and Operators (2-8-2025)

1. Write a C program to add two integers.

Input : Two integers (a, b)

Process : Add the two integers: sum = a + b

Output : Display the sum of a + b

Program:

#include <stdio.h>

void main()

{

int a, b, sum;

scanf("%d", &a);

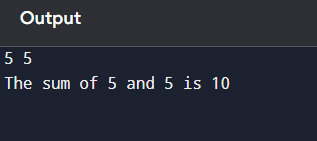
scanf("%d", &b);

sum = a + b;

printf("The sum of %d and %d is %d\n", a, b, sum);

}

Output :



1. Write a program to swap two numbers using a temporary variable.

Input : The program takes two integer values as input from the user, stored in variables a and b.

Process ; swap the values of a and b, a temporary variable temp is used. First, the value of a is assigned to temp. Then, the value of b is assigned to a, and finally, the value stored in temp (original a) is assigned to b.

Output : The program displays the values of a and b after swapping.

Program :

#include <stdio.h>

void main()

{

int a, b, temp;

scanf("%d", &a);

scanf("%d", &b);

temp = a;

a = b;

b = temp;

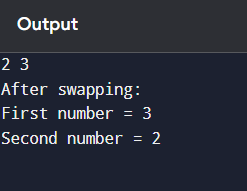
printf("After swapping:\n");

printf("First number = %d\n", a);

printf("Second number = %d\n", b);

}

Output :



.

1. Write a program to find the ASCII value of a character.

Input : The program takes two integer values from the user and stores them in variables a and b.

Process : To swap the values without using a temporary variable, arithmetic operations are used:

* First, a is assigned the sum of a and b.
* Then, b is assigned the difference of the new a and original b (which gives original a).
* Finally, a is assigned the difference of the new a and new b (which gives original b).

Output : The program displays the values of a and b after the swap.

Program :

#include <stdio.h>

void main()

{

int a, b;

scanf("%d", &a);

scanf("%d", &b);

a = a + b;

b = a - b;

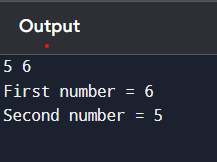
a = a - b;

printf("First number = %d\n", a);

printf("Second number = %d\n", b);

}

Output :



1. Write a program to calculate the area and perimeter of a rectangle.

Input : The program accepts the **length** and **breadth** of a rectangle from the user.

Process : t calculates the **area** using the formula area = length × breadth and the **perimeter** using the formula perimeter = 2 × (length + breadth).

Output : The program displays the **area** and **perimeter** of the rectangle.

Program :

#include <stdio.h>

void main()

{

int length, breadth, area, perimeter;

scanf("%d", &length);

scanf("%d", &breadth);

area = length \* breadth;

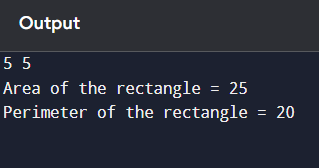
perimeter = 2 \* (length + breadth);

printf("Area of the rectangle = %d\n", area);

printf("Perimeter of the rectangle = %d\n", perimeter);

}

Output :



1. Write a program to compute the simple interest.

Input : The program takes the **principal amount (P)**, **rate of interest (R)**, and **time (T)** in years as input from the user.

Process : It calculates **Simple Interest** using the formula:  
Simple Interest = (P × R × T) / 100.

Output : The calculated **Simple Interest** is displayed to the user.

Program :

#include <stdio.h>

void main()

{

float principal, rate, time, interest;

scanf("%f", &principal);

scanf("%f", &rate);

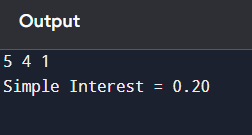
scanf("%f", &time);

interest = (principal \* rate \* time) / 100;

printf("Simple Interest = %.2f\n", interest);

}

Output :



1. Write a program to convert temperature from Celsius to Fahrenheit.

Input : The program accepts the **temperature in Celsius** from the user.

Process : It converts the temperature to Fahrenheit using the formula:  
Fahrenheit = (Celsius × 9/5) + 32.

Output : The program displays the **temperature in Fahrenheit**.

Program :

#include <stdio.h>

void main()

{

float celsius, fahrenheit;

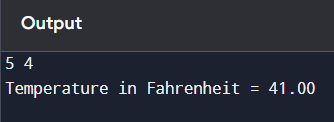
scanf("%f", &celsius);

fahrenheit = (celsius \* 9 / 5) + 32;

printf("Temperature in Fahrenheit = %.2f\n", fahrenheit);

}

Output :



1. Write a program to find the quotient and remainder of two integers.

Input : The program accepts two integer values from the user — the **dividend** and the **divisor**.

Process : It calculates the **quotient** using the division operator (/) and the **remainder** using the modulus operator (%).

Output : The program displays the **quotient** and **remainder**.

Program :

#include <stdio.h>

void main()

{

int dividend, divisor, quotient, remainder;

scanf("%d%d", &dividend,&divisor);

quotient = dividend / divisor;

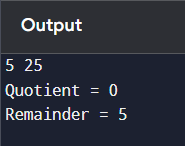
remainder = dividend % divisor;

printf("Quotient = %d\n", quotient);

printf("Remainder = %d\n", remainder);

}

Output :



1. Write a program to check whether a number is even or odd.

Input : The program accepts an integer number from the user.

Process : The program checks the **remainder** when the number is divided by 2 using the modulus operator (%).

Output : The program displays whether the number is **even** or **odd**.

Program :

#include <stdio.h>

int main() {

int number;

// Input

printf("Enter an integer: ");

scanf("%d", &number);

// Process & Output

if (number % 2 == 0) {

printf("%d is even.\n", number);

} else {

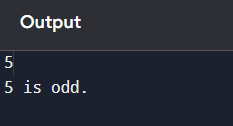
printf("%d is odd.\n", number);

}

return 0;

}

Output :



1. Write a program to calculate the square and cube of a number.

Input : The program takes a single integer number from the user.

Process : It calculates the **square** by multiplying the number by itself (number × number), and the **cube** by multiplying the number three times (number × number × number).

**Output : The program displays the square and cube of the entered number.**

Program :

#include <stdio.h>

void main()

{

int number, square, cube;

scanf("%d", &number);

square = number \* number;

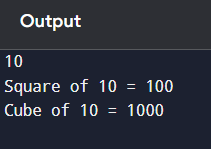
cube = number \* number \* number;

printf("Square of %d = %d\n", number, square);

printf("Cube of %d = %d\n", number, cube);

}

Output :



1. Write a program to swap number using two numbers without using a temporary varaiable

Input : The program takes two integer values from the user, stored in variables a and b.

Process : It swaps the values of a and b using arithmetic operations, without using a third (temporary) variable.

Output : The program displays the values of a and b after the swap.

Program :

#include <stdio.h>

void main()

{

int a, b;

scanf("%d", &a);

scanf("%d", &b);

a = a + b;

b = a - b;

a = a - b;

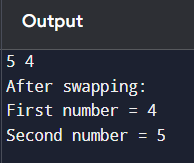
printf("After swapping:\n");

printf("First number = %d\n", a);

printf("Second number = %d\n", b);

}

Output :



# ✅ Day : Conditional Statements (4-8-2025)

1. Write a program to check if a number is positive, negative, or zero.

Input ; A number (integer)

Process :  If the number is greater than 0 → It is positive

* If the number is less than 0 → It is negative
* If the number is equal to 0 → It is zero

Output : Message stating whether the number is positive, negative, or zero

Program :

#include <stdio.h>

void main()

{

int number;

scanf("%d", &number);

if (number > 0) {

printf("The number is positive.\n");

} else if (number < 0) {

printf("The number is negative.\n");

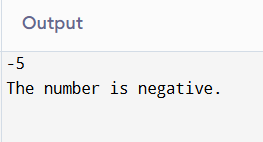
} else {

printf("The number is zero.\n");

}

}

Output :



1. Write a program to find the largest among three numbers.

Input : Three numbers (integers): a, b, c

Process : If a is greater than both b and c, then a is the largest

Else if b is greater than both a and c, then b is the largest

Else, c is the largest.

Output : The largest number among the three

Program :

#include <stdio.h>

void main() {

int a, b, c;

scanf("%d %d %d", &a, &b, &c);

if (a >= b && a >= c) {

printf("The largest number is %d\n", a);

} else if (b >= a && b >= c) {

printf("The largest number is %d\n", b);

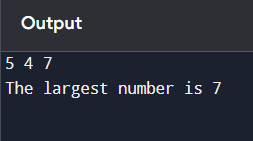
} else {

printf("The largest number is %d\n", c);

}

}

Output :



1. Write a program to check if a year is a leap year.

Input : A year (integer)

Process : It is **divisible by 4** **and** not divisible by 100  
**OR**

It is **divisible by 400**

Output : Message stating whether the year is a leap year or not

Program :

#include <stdio.h>

void main() {

int year;

scanf("%d", &year);

if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {

printf("%d is a leap year.\n", year);

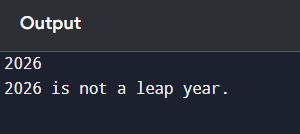
} else {

printf("%d is not a leap year.\n", year);

}

}

Output :



1. Write a program to check whether a character is a vowel or consonant.

Input : A character from the user.

Process : Check if the character is a vowel (a, e, i, o, u) or not. If it is a vowel, print vowel; otherwise, if it's an alphabet, print consonant.

Output : Message saying the character is a vowel or consonant.

Program :

#include <stdio.h>

void main()

{

char ch;

scanf(" %c", &ch);

if ((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z'))

{

if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' ||

ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U') {

printf("%c is a vowel.\n", ch);

} else {

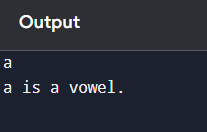
printf("%c is a consonant.\n", ch);

}

}

}

Output :



1. Write a program to assign grades based on marks.

Input : The user enters the marks (integer).

Process : If marks are 90 and above → Grade A

If marks are 80–89 → Grade B

If marks are 70–79 → Grade C

If marks are 60–69 → Grade D

If marks are below 60 → Grade F

Output : A message showing the grade based on the input marks

Program :

#include <stdio.h>

void main()

{

int marks;

scanf("%d", &marks);

if (marks >= 90)

{

printf("Grade A\n");

} else if (marks >= 80)

{

printf("Grade B\n");

} else if (marks >= 70)

{

printf("Grade C\n");

} else if (marks >= 60)

{

printf("Grade D\n");

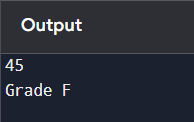
} else {

printf("Grade F\n");

}

}

Output :



1. Write a program to check whether a number is divisible by 5 and 11.

Input : The user enters an integer number

Process : Check if the number is divisible by both 5 and 11 using the modulus operator (%). If number % 5 == 0 and number % 11 == 0, then it is divisible by both; otherwise, it is not.

Output : A message stating whether the number is divisible by both 5 and 11 or not.

Program :

#include <stdio.h>

void main()

{

int number;

scanf("%d", &number);

if (number % 5 == 0 && number % 11 == 0)

{

printf("The number is divisible by both 5 and 11.\n");

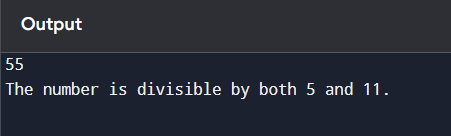
} else {

printf("The number is not divisible by both 5 and 11.\n");

}

}

Output ;



1. Write a program to find the absolute value of a number.

Input : The user enters an integer number.

Process : If the number is less than 0, multiply it by -1 to make it positive. If it is already positive or zero, leave it as it is.

Output : The absolute value of the number.

Program :

#include <stdio.h>

void main()

{

int number, absolute;

scanf("%d", &number);

if (number < 0) {

absolute = -number;

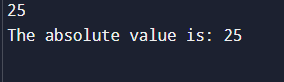
} else {

absolute = number;

}

printf("The absolute value is: %d\n", absolute);

}

.

1. Write a menu-driven program to perform +, -, \*, / operations.

Input : The user enters two numbers and selects an operation from a menu (1 for addition, 2 for subtraction, 3 for multiplication, 4 for division).

Process : Based on the user's choice, perform the selected operation on the two numbers using +, -, \*, or /.

Output : The result of the selected arithmetic operation

Program :

#include <stdio.h>

void main()

{

int choice;

float num1, num2, result;

scanf("%f %f %d", &num1, &num2,&choice);

switch (choice)

{

case 1:

result = num1 + num2;

printf("Result: %.2f\n", result);

break;

case 2:

result = num1 - num2;

printf("Result: %.2f\n", result);

break;

case 3:

result = num1 \* num2;

printf("Result: %.2f\n", result);

break;

case 4:

if (num2 != 0)

result = num1 / num2;

else

{

printf("Error: Division by zero is not allowed.\n");

}

printf("Result: %.2f\n", result);

break;

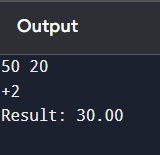
default:

printf("Invalid choice.\n");

}

}

Output :



1. Write a program to find roots of a quadratic equation

Input : The user enters three coefficients a, b, and c of a quadratic equation in the form ax² + bx + c = 0.

Process : Calculate the discriminant D = b² - 4ac.

If D > 0, the roots are real and different.

If D == 0, the roots are real and equal.

If D < 0, the roots are complex and imaginary.

Output : The roots of the equation, either real or complex.

Program :

#include <stdio.h>

#include <math.h>

void main()

{

float a, b, c, d, r1, r2;

scanf("%f %f %f", &a, &b, &c);

d = b \* b - 4 \* a \* c;

if (d > 0) {

r1 = (-b + sqrt(d)) / (2 \* a);

r2 = (-b - sqrt(d)) / (2 \* a);

printf("Real and different roots: %.2f and %.2f\n", r1, r2);

} else if (d == 0) {

r1 = -b / (2 \* a);

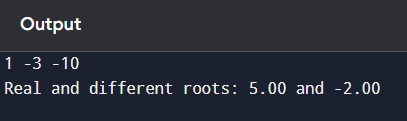
printf("Real and equal roots: %.2f\n", r1);

} else {

printf("Complex roots\n");

}

}



1. Write a program to find the number of digits in a number.

Input : The user enters an integer number.

Process : Divide the number by 10 repeatedly in a loop, and count how many times it takes until the number becomes 0.

Output : The total count, which represents the number of digits in the input number.

Program :

#include <stdio.h>

void main()

{

int num, count = 0;

scanf("%d", &num);

if (num == 0)

{

count = 1;

} else

{

while (num != 0)

{

num = num / 10;

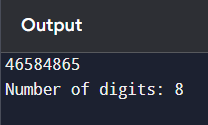
count++;

}

}

printf("Number of digits: %d\n", count);

}



# ✅ Day : Loops and Iterations (5-8-2025)

1. Write a program to print numbers from 1 to 100.

Input : None

Process: Use a loop to print numbers from 1 to 100

Output: Numbers from 1 to 100 printed one by one

Program :

#include <stdio.h>

void main()

{

int i;

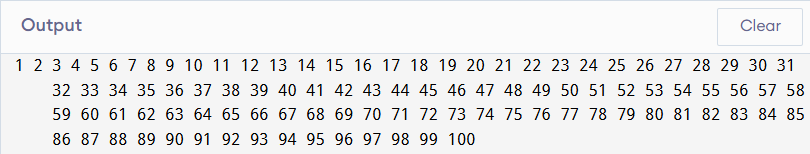
for(i = 1; i <= 100; i++)

{

printf(" %d", i);

}

}



1. Write a program to print even numbers from 1 to 50.

Input: None

Process: Use a loop to check and print numbers divisible by 2 (even numbers) from 1 to 50

Output: Even numbers from 1 to 50

Program : #include <stdio.h>

void main()

{

int i;

for(i = 1; i <= 50; i++)

{

if(i % 2 == 0)

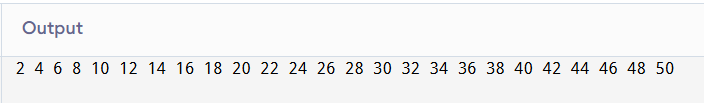
{

printf(" %d", i);

}

}

}



1. Write a program to find the factorial of a number.

Input: A number n

Process: Multiply all numbers from 1 to n (i.e., factorial = 1 × 2 × 3 × ... × n)

Output: Factorial of the given number

Program: #include <stdio.h>

void main()

{

int n, i;

unsigned long long factorial = 1;

scanf("%d", &n);

if(n < 0)

{

printf("Factorial is not defined for negative numbers.\n");

} else {

for(i = 1; i <= n; i++)

{

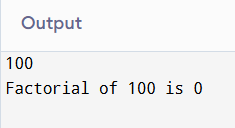
factorial \*= i;

}

printf("Factorial of %d is %llu\n", n, factorial);

}

}



1. Write a program to calculate the sum of digits of a number.

**Input:** A number n

**Process:** Extract each digit using modulus (%) and division (/) operations and add them

**Output:** Sum of the digits of the number

Program : #include <stdio.h>

void main()

{

int n, digit, sum = 0;

scanf("%d", &n);

while(n != 0)

{

digit = n % 10;

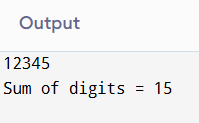
sum += digit;

n = n / 10;

}

printf("Sum of digits = %d\n", sum);

}



1. Write a program to reverse a number.

Input: A number n

Process:

* + Extract the last digit using % 10
  + Add it to the reverse number after multiplying reverse by 10
  + Remove the last digit using / 10

Output: Reversed number

Program: #include <stdio.h>

void main()

{

int n, digit, reverse = 0;

scanf("%d", &n);

while(n != 0)

{

digit = n % 10;

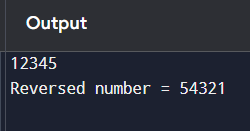
reverse = reverse \* 10 + digit;

n = n / 10;

}

printf("Reversed number = %d\n", reverse);

}



1. Write a program to check whether a number is a palindrome.

Input: A number n

Process:

* + Reverse the number using the same method as the reverse number program
  + Compare the reversed number with the original number

Output:

* + If the original number is equal to the reversed number, it's a palindrome
  + Otherwise, it's not a palindrome

Program :

#include <stdio.h>

int main()

{

int n, original, digit, reverse = 0;

scanf("%d", &n);

original = n;

while(n != 0)

{

digit = n % 10;

reverse = reverse \* 10 + digit;

n = n / 10;

}

if(original == reverse)

{

printf("%d is a palindrome.\n", original);

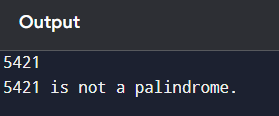
} else

{

printf("%d is not a palindrome.\n", original);

}

}



1. Write a program to print multiplication table of a number.

Input: A number n

Process: Multiply the number n with values from 1 to 10 and print the result in the form n \* i = result

Output: Multiplication table of the number

Program:

#include <stdio.h>

void main()

{

int n, i;

scanf("%d", &n);

printf("Multiplication Table of %d:\n", n);

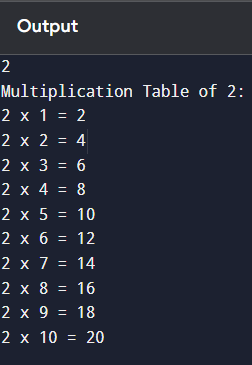
for(i = 1; i <= 10; i++)

{

printf("%d x %d = %d\n", n, i, n \* i);

}

}



1. Write a program to count the number of digits in a number.

Input: A number n

Process:

* + Use a loop to divide the number by 10 repeatedly
  + Each time a digit is removed, increase the count

Output: Total number of digits in the number

Program :

#include <stdio.h>

void main()

{

int n, count = 0;

scanf("%d", &n);

if(n == 0)

{

count = 1;

} else {

while(n != 0)

{

n = n / 10; // Remove last digit

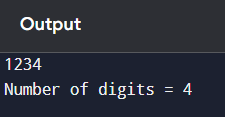
count++;

}

}

printf("Number of digits = %d\n", count);

}



1. Write a program to print the Fibonacci series up to n terms.

Input: An integer n (number of terms)

Process:

* + Initialize first two terms of Fibonacci series: 0 and 1
  + Use a loop to generate next terms by summing the previous two

Output: Fibonacci series up to n terms

Program:

#include <stdio.h>

void main()

{

int n, i;

int a = 0, b = 1, next;

scanf("%d", &n);

printf("Fibonacci Series: ");

for(i = 1; i <= n; i++)

{

printf("%d ", a);

next = a + b;

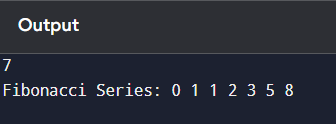
a = b;

b = next;

}

printf("\n");

}



1. Write a program to calculate the sum of the first n natural numbers.

Input: A number n

Process: Calculate the sum using either a loop or formula

* + Loop: Add all numbers from 1 to n
  + OR use the formula: sum = n × (n + 1) / 2

Output: Sum of the first n natural numbers

Program :

#include <stdio.h>

void main()

{

int n, i, sum = 0;

scanf("%d", &n);

for(i = 1; i <= n; i++)

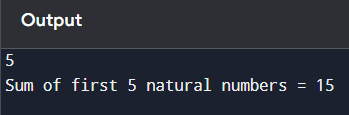
{

sum += i;

}

printf("Sum of first %d natural numbers = %d\n", n, sum);

}



# ✅ Day : Arrays (6-8-2025)

1. Write a program to read and print elements of an array.

|  |  |
| --- | --- |
| Input | Size of the array (n), elements of the array |

|  |  |
| --- | --- |
| Process | Read n elements into the array, then print them one by one |

|  |  |
| --- | --- |
| Output | : Elements of the array printed on the screen |
| Program :    #include <stdio.h>  void main()  {  int arr[100], n, i;  scanf("%d", &n);  for(i = 0; i < n; i++)  {  scanf("%d", &arr[i]);  }  printf("The elements of the array are:\n");  for(i = 0; i < n; i++)  {  printf("%d ", arr[i]);  }  } |  |

1. Write a program to find the sum of elements of an array.

**🔹 Input**

* Accept the size of the array (n)
* Read n integer elements into the array

**🔹 Process**

* Initialize a variable sum = 0
* Traverse the array using a loop
* Add each element to sum

**🔹 Output**

* Display the final value of sum (i.e., the sum of all elements in the array)

Program :

#include <stdio.h>

void main()

{

int arr[100], n, i, sum = 0;

scanf("%d", &n);

for(i = 0; i < n; i++)

{

scanf("%d", &arr[i]);

}

for(i = 0; i < n; i++)

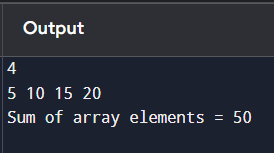
{

sum += arr[i];

}

printf("Sum of array elements = %d\n", sum);

}



1. Write a program to find the maximum and minimum element in an array.

**Input**

* Read size of array n
* Read n elements into the array

**Process**

* Initialize max and min with first element
* Compare each element:
  + Update max if element is greater
  + Update min if element is smaller

**Output**

* Print maximum and minimum elements

Program :

#include <stdio.h>

int main()

{

int arr[100], n, i;

int max, min;

scanf("%d", &n);

for(i = 0; i < n; i++)

{

scanf("%d", &arr[i]);

}

max = min = arr[0];

for(i = 1; i < n; i++)

{

if(arr[i] > max)

max = arr[i];

if(arr[i] < min)

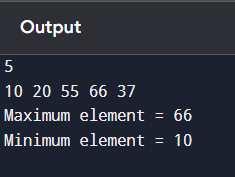
min = arr[i];

}

printf("Maximum element = %d\n", max);

printf("Minimum element = %d\n", min);

}



1. Write a program to reverse an array.

**Input**

* Read size of the array n
* Read n elements into the array

**Process**

* Swap elements from start and end moving towards the center  
  (i.e., swap arr[0] with arr[n-1], arr[1] with arr[n-2], etc.)

**Output**

* Print the reversed array

Program :

#include <stdio.h>

void main()

{

int arr[100], n, i, temp;

scanf("%d", &n);

for(i = 0; i < n; i++)

{

scanf("%d", &arr[i]);

}

for(i = 0; i < n / 2; i++)

{

temp = arr[i];

arr[i] = arr[n - 1 - i];

arr[n - 1 - i] = temp;

}

printf("Reversed array:\n");

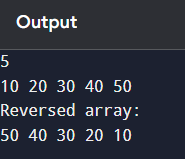
for(i = 0; i < n; i++)

{

printf("%d ", arr[i]);

}

}



1. Write a program to search for an element in an array (linear search)

**Input**

* Read size of array n
* Read n elements into the array
* Read the element to be searched (key)

**Process**

* Traverse the array from start to end
* Compare each element with key
* If found, note the position

**Output**

* Print the position if element is found
* Else print “Element not found”

Program :

#include <stdio.h>

void main()

{

int arr[100], n, i, key, found = 0;

scanf("%d", &n);

for(i = 0; i < n; i++)

{

scanf("%d", &arr[i]);

}

scanf("%d", &key);

for(i = 0; i < n; i++)

{

if(arr[i] == key)

{

found = 1;

break;

}

}

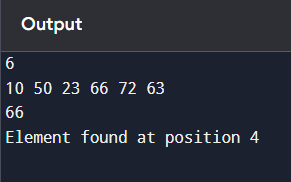
if(found)

printf("Element found at position %d\n", i + 1); // positions start at 1

else

printf("Element not found\n");

}



.

1. Write a program to sort an array in ascending order.

**nput**

* Read size of array n
* Read n elements into the array

**✅ Process**

* Use nested loops to compare and swap elements
* Sort elements in ascending order using Bubble Sort (or any other method)

**✅ Output**

* Print the sorted array in ascending order

Program :

#include <stdio.h>

void main()

{

int arr[100], n, i, j, temp;

scanf("%d", &n);

for(i = 0; i < n; i++)

{

scanf("%d", &arr[i]);

}

for(i = 0; i < n - 1; i++) {

for(j = 0; j < n - 1 - i; j++)

{

if(arr[j] > arr[j + 1])

{

temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

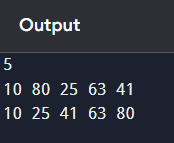
for(i = 0; i < n; i++)

{

printf("%d ", arr[i]);

}

}



1. Write a program to insert an element in an array.

* Input Read size of the array n
* Read n elements into the array
* Read the element to insert (value)
* Read the position (pos) where the element is to be inserted

✅ Process

Shift elements

* from the end to pos one step right
* Insert value at position pos

✅ Output

Program :

#include <stdio.h>

void main()

{

int arr[100], n, pos, val, i;

scanf("%d", &n);

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

scanf("%d %d", &pos, &val);

for(i = n; i >= pos; i--)

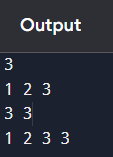
arr[i] = arr[i - 1];

arr[pos - 1] = val;

n++;

for(i = 0; i < n; i++) printf("%d ", arr[i]);

}



1. Write a program to delete an element from an array.

Input: Array size n, elements, position to delete

Process: Shift elements left from position to end

Output: Print updated array

Program :

#include <stdio.h>

int main()

{

int arr[100], n, pos, i;

scanf("%d", &n);

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

scanf("%d", &pos);

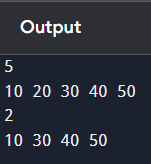
for(i = pos - 1; i < n - 1; i++)

arr[i] = arr[i + 1];

n--;

for(i = 0; i < n; i++) printf("%d ", arr[i]);

}



1. Write a program to find the frequency of elements in an array.

Input: Array size n, read n elements

Process:

* + For each element, count how many times it appears
  + Mark duplicates as visited

Output: Display each unique element and its frequency

Program :

#include <stdio.h>

void main()

{

int arr[100], freq[100], n, i, j, count;

scanf("%d", &n);

for(i = 0; i < n; i++)

{

scanf("%d", &arr[i]);

freq[i] = -1;

}

for(i = 0; i < n; i++) {

count = 1;

if(freq[i] != 0) {

for(j = i + 1; j < n; j++)

{

if(arr[i] == arr[j])

{

count++;

freq[j] = 0;

}

}

freq[i] = count;

}

}

for(i = 0; i < n; i++)

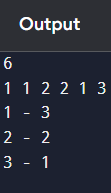
{

if(freq[i] != 0)

{

printf("%d - %d\n", arr[i], freq[i]);

}

}

1. Write a program to merge two arrays.

**Input**:

* + Size and elements of first array (a[])
  + Size and elements of second array (b[])

**Process** :Copy all elements of a[] into merged[]

* + Append all elements of b[] to merged[]

**Output**:

* + Print the merged array

Program :

#include <stdio.h>

void main() {

int a[50], b[50], c[100], n1, n2, i;

scanf("%d %d", &n1, &n2);

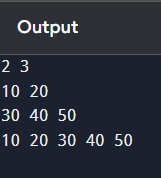
for(i = 0; i < n1; i++) scanf("%d", &a[i]);

for(i = 0; i < n2; i++) scanf("%d", &b[i]);

for(i = 0; i < n1; i++) c[i] = a[i];

for(i = 0; i < n2; i++) c[n1 + i] = b[i];

for(i = 0; i < n1 + n2; i++) printf("%d ", c[i]);

}

# ✅ Day : Strings (7-8-2025)

1. Write a program to find the length of a string without using strlen().
2. Write a program to copy one string to another.
3. Write a program to concatenate two strings.
4. Write a program to compare two strings.
5. Write a program to count vowels and consonants in a string.
6. Write a program to convert lowercase to uppercase and vice versa.
7. Write a program to check if a string is palindrome.
8. Write a program to reverse a string.
9. Write a program to count words in a string.
10. Write a program to find the frequency of each character in a string.

# ✅ Day : Functions (8-8-2025)

1. Write a function to find the factorial of a number.
2. Write a function to check whether a number is prime.
3. Write a function to calculate power using recursion.
4. Write a function to check palindrome number using recursion.
5. Write a function to calculate nCr (combinations).
6. Write a program to demonstrate call by value and call by reference.
7. Write a program using function to swap two numbers.
8. Write a recursive function to find the nth Fibonacci number.
9. Write a program to find GCD and LCM using functions.
10. Write a program to demonstrate global and local variables.

# ✅ Day : Pointers (9-8-2025)

1. Write a program to print the address of a variable using pointer.
2. Write a program to access array elements using pointers.
3. Write a program to swap two numbers using pointers.
4. Write a program to add two numbers using pointers.
5. Write a program to find the length of a string using pointers.
6. Write a program to reverse a string using pointers.
7. Write a program to count vowels using pointer.
8. Write a program to demonstrate pointer to pointer.
9. Write a program to allocate memory using malloc() and free it.
10. Write a program to sort an array using pointer notation.

# ✅ Day : Structures and Unions (11-8-2025)

1. Define a structure for student record and print details.
2. Write a program to store and display employee details using structures.
3. Write a program to pass a structure to a function.
4. Write a program to store multiple student records using array of structures.
5. Write a program to demonstrate nested structures.
6. Write a program to calculate total and average marks using structures.
7. Write a program to find the highest marks among students.
8. Write a program to sort student records by name using structure.
9. Write a program using union to store data of different types.
10. Compare and contrast structure vs union with a sample program.

# ✅ Day : File Handling (12-8-2025)

1. Write a program to create and write to a text file.
2. Write a program to read contents of a file and display.
3. Write a program to count number of lines in a file.
4. Write a program to copy contents from one file to another.
5. Write a program to append text to a file.
6. Write a program to count vowels in a file.
7. Write a program to read integers from a file and find the sum.
8. Write a program to read a structure from a file.
9. Write a program to sort names stored in a file.
10. Write a program to search for a word in a file.