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

 **Input**: Data supplied to the program (e.g. numbers, text, user entries)

 **Process**: Computation or logic that transforms the input (e.g. calculations, loops, )

 **Output**: The result produced by the program (e.g. printed values, displayed data, file output

#include <stdio.h>

void main()

{

int n;

printf("Enter the number of elements: ");

scanf("%d", &n);

int arr[n];

printf("Enter %d integer values:\n", n);

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

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

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

}

printf("\nYou entered:\n");

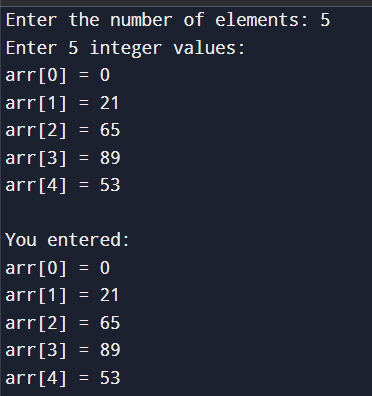
for (int i = 0; i < n; i++)

{

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

}

}



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

 **Input**: User provides the array size (n) and its elements.

 **Process**: Use a loop to accumulate the sum of all elements

 **Output**: Display the final sum to the user.

#include <stdio.h>

void main()

{

int arr[] = {10, 20, 30, 40, 50};

int n = sizeof(arr) / sizeof(arr[0]);

int sum = 0;

for (int 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**: User supplies the number of elements n and the values of the array.

 **Process**: Traverse the array, updating max and min based on comparisons.

 **Output**: Display the maximum and minimum values found.

#include <stdio.h>

void main()

{

int n;

printf("Enter number of elements: ");

scanf("%d", &n);

int arr[n];

printf("Enter %d integer elements:\n", n);

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

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

}

int max = arr[0];

int min = arr[0];

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

{

if (arr[i] > max)

{

max = arr[i];

}

if (arr[i] < min)

{

min = arr[i];

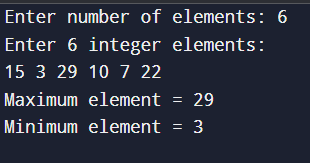
}

}

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

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

}



1. Write a program to reverse an array.

|  |  |  |
| --- | --- | --- |
| **Input** | Read n and array values | scanf("%d", &n);, for ... scanf |

|  |  |  |
| --- | --- | --- |
| **Process** | Swap elements from ends toward middle | while(start < end) { swap } |

|  |  |  |
| --- | --- | --- |
| **Output** | Print reversed array | for (...) printf("%d ", arr[i]); |

#include <stdio.h>

void main()

{

int n;

printf("Enter number of elements: ");

scanf("%d", &n);

int arr[n];

printf("Enter %d integers:\n", n);

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

{

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

}

int start = 0, end = n - 1;

while (start < end)

{

int temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

start++;

end--;

}

printf("Reversed array:\n");

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

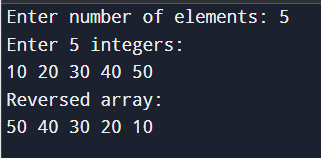
{

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

}

printf("\n");

}



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

 **Input**: User enters the array size, each element, and the search key.

 **Process**: Iterate through the array from start to finish, comparing each element with the search key.

 **Output**: Display the index of the first matching element (0-based) if found, or indicate "not found".

#include <stdio.h>

void main()

{

int n, key, index = -1;

printf("Enter number of elements: ");

scanf("%d", &n);

int arr[n];

printf("Enter %d integer elements:\n", n);

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

{

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

}

printf("Enter element to search: ");

scanf("%d", &key);

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

{

if (arr[i] == key)

{

index = i;

break;

}

}

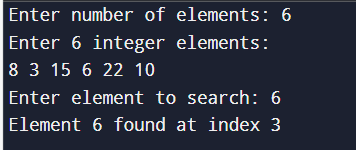
if (index != -1)

printf("Element %d found at index %d\n", key, index);

else

printf("Element %d not found in array\n", key);

}



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

**Input**: User supplies the number of elements (n) and the integer values.

**Process**: Use nested loops to compare adjacent elements and swap them if they’re in the wrong order (bubble sort).

**Output**: Print the sorted array in ascending order.

#include <stdio.h>

void main()

{

int n, temp;

printf("Enter number of elements: ");

scanf("%d", &n);

int arr[n];

printf("Enter %d integer elements:\n", n);

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

{

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

}

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

{

for (int 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 in ascending order:\n");

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

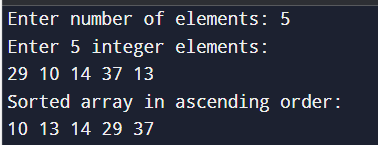
{

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

}

printf("\n");

}



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

 **Input**: Read the current size of the array, its elements, the value to insert, and the desired position.

 **Process**: Shift elements to the right from the insertion index to make space, then place the new element and update the size.

 **Output**: Display the array after insertion.

#include <stdio.h>

void main()

{

int n, pos, val;

printf("Enter number of elements (max 50): ");

scanf("%d", &n);

int arr[50];

printf("Enter %d integers:\n", n);

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

{

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

}

printf("Enter value to insert: ");

scanf("%d", &val);

printf("Enter position to insert (1‑based): ");

scanf("%d", &pos);

if (pos < 1 || pos > n + 1)

{

printf("Invalid position!\n");

}

for (int i = n - 1; i >= pos - 1; i--)

{

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

}

arr[pos - 1] = val;

n++;

printf("Array after insertion:\n");

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

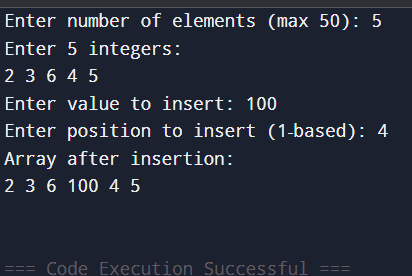
{

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

}

printf("\n");

}



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

 **Input**: User provides the number of elements (n), the array elements, and the element to delete.

 **Process**: Search for the element in the array. If found, shift all subsequent elements one position to the left to overwrite the deleted element.

 **Output**: Display the array after deletion.

#include <stdio.h>

void main()

{

int n, key, i, pos = -1;

printf("Enter number of elements: ");

scanf("%d", &n);

int arr[n];

printf("Enter %d integers:\n", n);

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

{

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

}

printf("Enter element to delete: ");

scanf("%d", &key);

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

{

if (arr[i] == key)

{

pos = i;

break;

}

}

if (pos != -1)

{

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

{

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

}

n--;

}

if (pos != -1)

{

printf("Array after deletion:\n");

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

{

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

}

printf("\n");

}

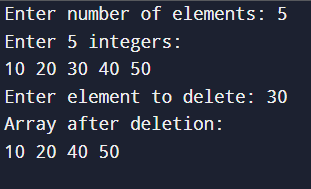
else

{

printf("Element %d not found in array\n", key);

}

}



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

**Input**: User provides the number of elements (n), the array elements.

**Process**: Count the occurrences of each element using nested loops.

**Output**: Display the frequency of each element.

#include <stdio.h>

Void main()

{

int n, i, j, count;

printf("Enter number of elements: ");

scanf("%d", &n);

int arr[n];

printf("Enter %d integers:\n", n);

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

{

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

}

printf("Frequency of each element:\n");

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

{

count = 1;

if (arr[i] != -1)

{

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

{

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

{

count++;

arr[j] = -1; // Mark as counted

}

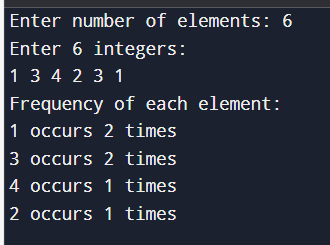
}

printf("%d occurs %d times\n", arr[i], count);

}

}

}



1. Write a program to merge two arrays.

**Input**: User provides the sizes and elements of two arrays.

**Process**: Copy elements from both arrays into a new array.

**Output**: Display the merged array.

#include <stdio.h>

void main()

{

int size1, size2, i, j;

printf("Enter the number of elements for the first array: ");

scanf("%d", &size1);

int arr1[size1];

printf("Enter elements for the first array:\n");

for (i = 0; i < size1; i++)

{

scanf("%d", &arr1[i]);

}

printf("Enter the number of elements for the second array: ");

scanf("%d", &size2);

int arr2[size2];

printf("Enter elements for the second array:\n");

for (i = 0; i < size2; i++)

{

scanf("%d", &arr2[i]);

}

int merged[size1 + size2];

for (i = 0; i < size1; i++)

{

merged[i] = arr1[i];

}

for (i = 0, j = size1; i < size2; i++, j++)

{

merged[j] = arr2[i];

}

printf("Merged array:\n");

for (i = 0; i < size1 + size2; i++)

{

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

}

printf("\n");

}

