

The screenshot shows a code editor interface with a dark theme. On the left, there's a sidebar with icons for various file types: Python, C/C++, C, C++, C, C, C, JS, TS, and Go. The main area has tabs for "main.c" and "Output". The "main.c" tab contains the following C code:

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int arr[100], n, i;
6
7     // Read number of elements
8     printf("Enter the number of elements: ");
9     scanf("%d", &n);
10
11    // Read array elements
12    printf("Enter %d elements:\n", n);
13    for(i = 0; i < n; i++)
14    {
15        scanf("%d", &arr[i]);
16    }
17
18    // Print array elements
19    printf("The elements of the array are:\n");
20    for(i = 0; i < n; i++)
21    {
22        printf("%d ", arr[i]);
23    }
24
25    return 0;
26 }
```

The "Output" tab shows the execution results:

```
Enter the number of elements: 5
Enter 5 elements:
10 20 30 40 50
The elements of the array are:
10 20 30 40 50
--- Code Execution Successful ---
```

At the bottom of the editor, there is a URL: [https://programiz.pro/?utm\\_source=backfill-ads&utm\\_medium=programiz&utm\\_campaign=pubgalaxy-creatives](https://programiz.pro/?utm_source=backfill-ads&utm_medium=programiz&utm_campaign=pubgalaxy-creatives)

The screenshot shows a code editor interface with a dark theme. On the left, there is a vertical toolbar with icons for various languages and tools. The main area displays a C program named 'main.c'. The code prompts the user to enter the number of elements and their values, then calculates and prints the sum. The output window shows the execution results.

main.c

```
1 #include <stdio.h>
2
3 int main() {
4     int arr[100], n, i;
5     int sum = 0;
6
7     printf("Enter number of elements: ");
8     scanf("%d", &n);
9
10    printf("Enter %d elements:\n", n);
11    for(i = 0; i < n; i++) {
12        scanf("%d", &arr[i]);
13    }
14
15    for(i = 0; i < n; i++) {
16        sum = sum + arr[i];
17    }
18
19    printf("Sum of array elements = %d", sum);
20
21    return 0;
22 }
23
```

Output

```
Enter number of elements: 4
Enter 4 elements:
5 10 15 20
Sum of array elements = 50
*** Code Execution Successful ***
```

The screenshot shows a code editor interface with a dark theme. On the left, there is a vertical toolbar with icons for various languages and tools. The main area displays a C program named `main.c`. The code reads the number of elements and the elements themselves from the user, then finds and prints the maximum and minimum values.

```
main.c
8  scanf("%d", &n);
9
10 printf("Enter %d elements:\n", n);
11 for(i = 0; i < n; i++) {
12     scanf("%d", &arr[i]);
13 }
14
15 // Assume first element is max and min
16 max = arr[0];
17 min = arr[0];
18
19 for(i = 1; i < n; i++) {
20     if(arr[i] > max) {
21         max = arr[i];
22     }
23     if(arr[i] < min) {
24         min = arr[i];
25     }
26 }
27
28 printf("Maximum element = %d\n", max);
29 printf("Minimum element = %d", min);
30
31 return 0;
32 }
```

The output window shows the execution results:

```
Enter number of elements: 5
Enter 5 elements:
12 45 7 30 20
Maximum element = 45
Minimum element = 7
== Code Execution Successful ==
```

The screenshot shows a code editor interface with a dark theme. On the left, there's a vertical toolbar with icons for various file types and a search function. The main area contains a C program named 'main.c'.

```
main.c
5 // Input number of elements
6 printf("Enter number of elements: ");
7 scanf("%d", &n);
8
9 // Input array elements
10 printf("Enter %d elements:\n", n);
11 for(i = 0; i < n; i++) {
12     scanf("%d", &arr[i]);
13 }
14
15 // Reversing the array
16 for(i = 0; i < n / 2; i++) {
17     temp = arr[i];
18     arr[i] = arr[n - i - 1];
19     arr[n - i - 1] = temp;
20 }
21
22 // Display reversed array
23 printf("Reversed array:\n");
24 for(i = 0; i < n; i++) {
25     printf("%d ", arr[i]);
26 }
27
28
29 return 0;
30
```

The code implements a simple algorithm to reverse an array. It first prompts the user for the number of elements and the elements themselves. Then, it iterates through the first half of the array, swapping each element with its mirror image across the center. Finally, it prints the reversed array.

At the top right, there are buttons for Run, Share, and Output. The Output panel shows the following terminal session:

```
Enter number of elements: 5
Enter 5 elements:
1 2 3 4 5
Reversed array:
5 4 3 2 1
*** Code Execution Successful ***
```

The screenshot shows a code editor interface with a dark theme. On the left, there is a vertical toolbar with icons for various file types and operations. The main area displays a C program named 'main.c'. The code implements a linear search algorithm to find a specific element in an array. The user interacts with the program by entering the number of elements and the element to search for. The output window shows the program's response, including the entered values and the position where the element was found.

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int arr[100], n, i, key, found = 0;
5
6     // Input number of elements
7     printf("Enter number of elements: ");
8     scanf("%d", &n);
9
10    // Input array elements
11    printf("Enter %d elements:\n", n);
12    for(i = 0; i < n; i++) {
13        scanf("%d", &arr[i]);
14    }
15
16    // Element to search
17    printf("Enter element to search: ");
18    scanf("%d", &key);
19
20    // Linear Search
21    for(i = 0; i < n; i++) {
22        if(arr[i] == key) {
23            found = 1;
24            printf("Element found at position %d\n", i + 1);
25            break;
26        }
27    }
28}
```

Output

```
Enter number of elements: 5
Enter 5 elements:
10 20 30 40 50
Enter element to search: 30
Element found at position 3

== Code Execution Successful ==
```

The screenshot shows a code editor interface with a dark theme. On the left, there is a vertical toolbar with icons for various file types: Python (py), C/C++ (c/cpp), C (c), C# (cs), Java (java), JavaScript (js), TypeScript (ts), and Go (go). The main area displays a C program named 'main.c'.

```
main.c
10 // Input array elements
11 printf("Enter %d elements:\n", n);
12 for(i = 0; i < n; i++) {
13     scanf("%d", &arr[i]);
14 }
15
16 // Sorting in ascending order (Bubble Sort)
17 for(i = 0; i < n - 1; i++) {
18     for(j = 0; j < n - i - 1; j++) {
19         if(arr[j] > arr[j + 1]) {
20             // Swap
21             temp = arr[j];
22             arr[j] = arr[j + 1];
23             arr[j + 1] = temp;
24         }
25     }
26 }
27
28 // Display sorted array
29 printf("Array in ascending order:\n");
30 for(i = 0; i < n; i++) {
31     printf("%d ", arr[i]);
32 }
33
34 return 0;
```

The top bar includes tabs for 'main.c', 'Run', and 'Output'. The 'Run' button is highlighted in blue. To the right of the Run button is a 'Clear' button. The 'Output' section shows the execution results:

```
Enter number of elements: 5
Enter 5 elements:
40 10 30 20 50
Array in ascending order:
10 20 30 40 50
== Code Execution Successful ==
```

The screenshot shows a code editor interface with a dark theme. On the left, there is a vertical toolbar with icons for various languages and tools. The main area displays a C program named 'main.c'. The code implements an array insertion operation. It first asks for the number of elements, then for four elements, and then for an insertion position and element. It shifts elements to the right and inserts the new element at the specified position. Finally, it prints the updated array. The output window shows the interaction with the user and the resulting array.

```
main.c
14 }
15
16 // Input position and element to insert
17 printf("Enter position to insert (1 to %d): ", n + 1);
18 scanf("%d", &pos);
19
20 printf("Enter element to insert: ");
21 scanf("%d", &element);
22
23 // Shift elements to the right
24 for(i = n; i >= pos; i--) {
25     arr[i] = arr[i - 1];
26 }
27
28 // Insert the element
29 arr[pos - 1] = element;
30 n++;
31
32 // Display updated array
33 printf("Array after insertion:\n");
34 for(i = 0; i < n; i++) {
35     printf("%d ", arr[i]);
36 }
37
38 return 0;
```

Output

```
Enter number of elements: 4
Enter 4 elements:
10 20 30 40
Enter position to insert (1 to 5): 2 25
Enter element to insert: Array after insertion:
10 25 20 30 40
== Code Execution Successful ==
```

main.c



Share



Output

Clear

```
1 #include <stdio.h>
2
3 int main() {
4     int arr[100], n, i, pos;
5
6     // Input number of elements
7     printf("Enter number of elements: ");
8     scanf("%d", &n);
9
10    // Input array elements
11    printf("Enter %d elements:\n", n);
12    for(i = 0; i < n; i++) {
13        scanf("%d", &arr[i]);
14    }
15
16    // Input position to delete
17    printf("Enter position to delete (1 to %d): ", n);
18    scanf("%d", &pos);
19
20    // Shift elements to the left
21    for(i = pos - 1; i < n - 1; i++) {
22        arr[i] = arr[i + 1];
23    }
24
25    n--; // Reduce array size
```

```
Enter number of elements: 5
Enter 5 elements:
10 20 30 40 50
Enter position to delete (1 to 5): 3
Array after deletion:
10 20 40 50

== Code Execution Successful ==
```

The screenshot shows a code editor interface with a dark theme. On the left, there is a vertical toolbar with icons for various file types and a search function. The main area displays a C program named `main.c`. The code prompts the user to enter the number of elements, the elements themselves, and a key element to find the frequency of. It then iterates through the array to count how many times the key element appears.

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int arr[100], n, i, key, count = 0;
5
6     // Input number of elements
7     printf("Enter number of elements: ");
8     scanf("%d", &n);
9
10    // Input array elements
11    printf("Enter %d elements:\n", n);
12    for(i = 0; i < n; i++) {
13        scanf("%d", &arr[i]);
14    }
15
16    // Element to find frequency
17    printf("Enter element to find frequency: ");
18    scanf("%d", &key);
19
20    // Count frequency
21    for(i = 0; i < n; i++) {
22        if(arr[i] == key) {
23            count++;
24        }
25    }
}
```

The output window on the right shows the execution results:

```
Enter number of elements: 6
Enter 6 elements:
1 2 2 3 1 2
Enter element to find frequency: 3
Frequency of 3 = 1 times

== Code Execution Successful ==
```

The screenshot shows a code editor interface with a dark theme. On the left, there is a vertical toolbar with icons for various languages and tools. The main area displays a C program named 'main.c'. The code reads two arrays from user input and merges them into a third array. The output window shows the execution results.

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int arr1[50], arr2[50], merged[100];
5     int n1, n2, i;
6
7     // Input first array size and elements
8     printf("Enter number of elements in first array: ");
9     scanf("%d", &n1);
10
11    printf("Enter %d elements of first array:\n", n1);
12    for(i = 0; i < n1; i++) {
13        scanf("%d", &arr1[i]);
14    }
15
16    // Input second array size and elements
17    printf("Enter number of elements in second array: ");
18    scanf("%d", &n2);
19
20    printf("Enter %d elements of second array:\n", n2);
21    for(i = 0; i < n2; i++) {
22        scanf("%d", &arr2[i]);
23    }
24
25    // Merge arrays
```

Output

```
Enter number of elements in first array: 3
Enter 3 elements of first array:
1 2 3
Enter number of elements in second array: 3
Enter 3 elements of second array:
4 5 6
Merged array:
1 2 3 4 5 6
== Code Execution Successful ==
```