

Wednesday, February 5, 2025

DATA STRUCTURE

EXPERIMENT NO. 1

-BY URVASHI PAHUJA , BCA 2B

24/SCA/BCA/087

SUBMITTED TO- MISS STUTI TANDON

Q1. WRITE A PROGRAM IN C TO
IMPLEMENT INSERTION IN 1-D
ARRAYS.

SOL-#include <stdio.h>

```
void insertElement(int arr[], int *size, int pos, int  
value) {
```

```
    // Shift elements to the right to make space for the  
    new element
```

```
    for (int i = *size - 1; i >= pos; i--) {
```

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

```
}
```

```
// Insert the new element at the desired position
```

```
arr[pos] = value;
```

```
// Increment the size of the array
```

```
(*size)++;
```

```
}
```

```
int main() {
```

```
    int arr[100], n, pos, value;
```

```
    // Input the size of the array
```

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

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

```
    // Input the elements of the array
```

```
    printf("Enter the elements of the array:\n");
```

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

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

```
}
```

```
// Input the position and value to insert
```

```
printf("Enter the position to insert the element (0  
to %d): ", n);
```

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

```
printf("Enter the value to insert: ");
```

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

```
// Check if the position is valid
```

```
if (pos < 0 || pos > n) {
```

```
    printf("Invalid position!\n");
```

```
} else {
```

```
    // Perform the insertion
```

```
    insertElement(arr, &n, pos, value);
```

```
// Print the updated array
```

```
printf("Updated array: ");
```

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

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

```

    }

    printf("\n");

}

return 0;

}

```

The screenshot displays the Programiz Online C Compiler interface. The browser address bar shows the URL `programiz.com/c-programming/online-compiler/`. The page header includes the Programiz logo, Google Ads, and a promotional banner for up to ₹60,000 in Ads credit. The main editor area shows a C program in `main.c` with the following code:

```

1  #include <stdio.h>
2
3  void insertElement(int arr[], int *size, int pos, int value) {
4      // Shift elements to the right to make space for the new element
5      for (int i = *size - 1; i >= pos; i--) {
6          arr[i + 1] = arr[i];
7      }
8
9      // Insert the new element at the desired position
10     arr[pos] = value;
11
12     // Increment the size of the array
13     (*size)++;
14 }
15
16 int main() {
17     int arr[100], n, pos, value;
18
19     // Input the size of the array
20     printf("Enter the number of elements: ");
21     scanf("%d", &n);
22

```

The output window on the right shows the program's execution results:

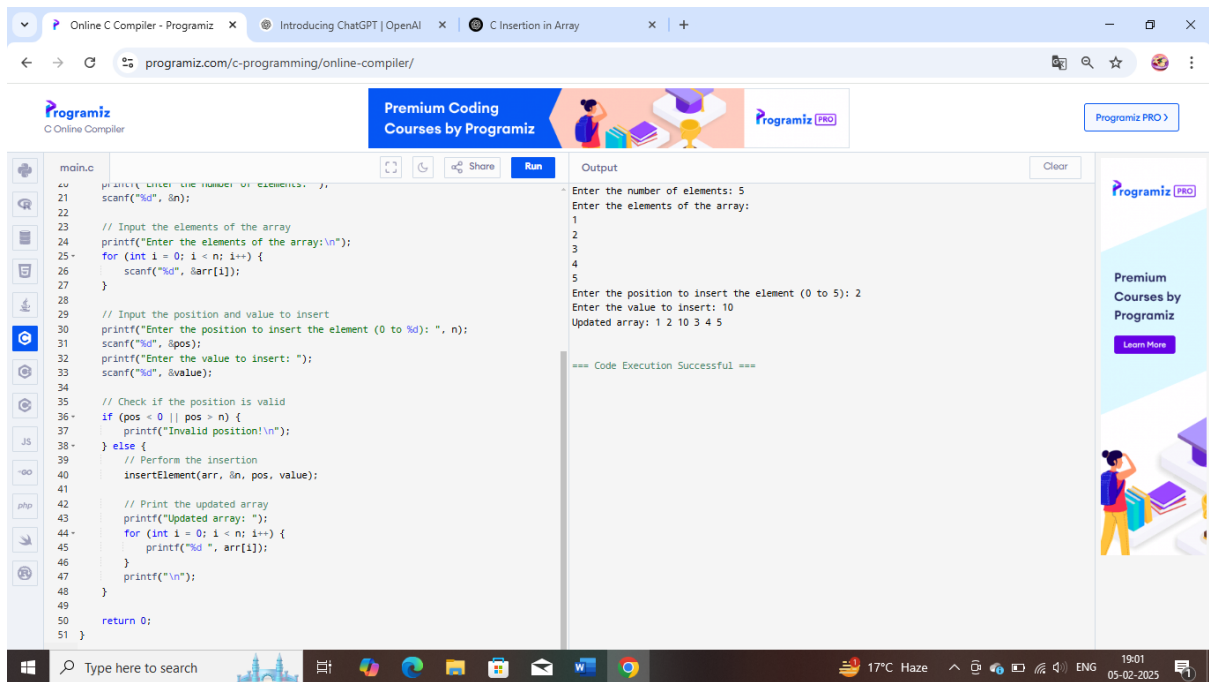
```

Enter the number of elements: 5
Enter the elements of the array:
1
2
3
4
5
Enter the position to insert the element (0 to 5): 2
Enter the value to insert: 10
Updated array: 1 2 10 3 4 5

=== Code Execution Successful ===

```

The Windows taskbar at the bottom shows the system clock as 17:17 on 05-02-2025, with the language set to ENG.



```
main.c
20 printf("Enter the number of elements: ");
21 scanf("%d", &n);
22
23 // Input the elements of the array
24 printf("Enter the elements of the array:\n");
25 for (int i = 0; i < n; i++) {
26     scanf("%d", &arr[i]);
27 }
28
29 // Input the position and value to insert
30 printf("Enter the position to insert the element (0 to %d): ", n);
31 scanf("%d", &pos);
32 printf("Enter the value to insert: ");
33 scanf("%d", &value);
34
35 // Check if the position is valid
36 if (pos < 0 || pos > n) {
37     printf("Invalid position!\n");
38 } else {
39     // Perform the insertion
40     insertElement(arr, &n, pos, value);
41
42     // Print the updated array
43     printf("Updated array: ");
44     for (int i = 0; i < n; i++) {
45         printf("%d ", arr[i]);
46     }
47     printf("\n");
48 }
49
50 return 0;
51 }
```

Output

```
Enter the number of elements: 5
Enter the elements of the array:
1
2
3
4
5
Enter the position to insert the element (0 to 5): 2
Enter the value to insert: 10
Updated array: 1 2 10 3 4 5

=== Code Execution Successful ===
```

Q2. WRITE A PROGRAM IN C TO IMPLEMENT DELETION IN 1-D ARRAYS.

SOL- #include <stdio.h >

```
void deleteElement(int arr[], int *size, int pos) {  
    // Check if the position is valid
```

```
if (pos < 0 || pos >= *size) {  
    printf("Invalid position!\n");  
    return;  
}
```

```
// Shift elements to the left to fill the gap  
for (int i = pos; i < *size - 1; i++) {  
    arr[i] = arr[i + 1];  
}
```

```
// Decrease the size of the array  
(*size)--;  
}
```

```
int main() {  
    int arr[100], n, pos;  
  
    // Input the size of the array  
    printf("Enter the number of elements: ");  
    scanf("%d", &n);
```

```
// Input the elements of the array
```

```
printf("Enter the elements of the array:\n");
```

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

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

```
}
```

```
// Input the position to delete
```

```
printf("Enter the position to delete the element (0  
to %d): ", n - 1);
```

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

```
// Perform the deletion
```

```
deleteElement(arr, &n, pos);
```

```
// Print the updated array
```

```
printf("Updated array: ");
```

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

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

```
}
```

```
printf("\n");
```

```
return 0;
```

```
}
```

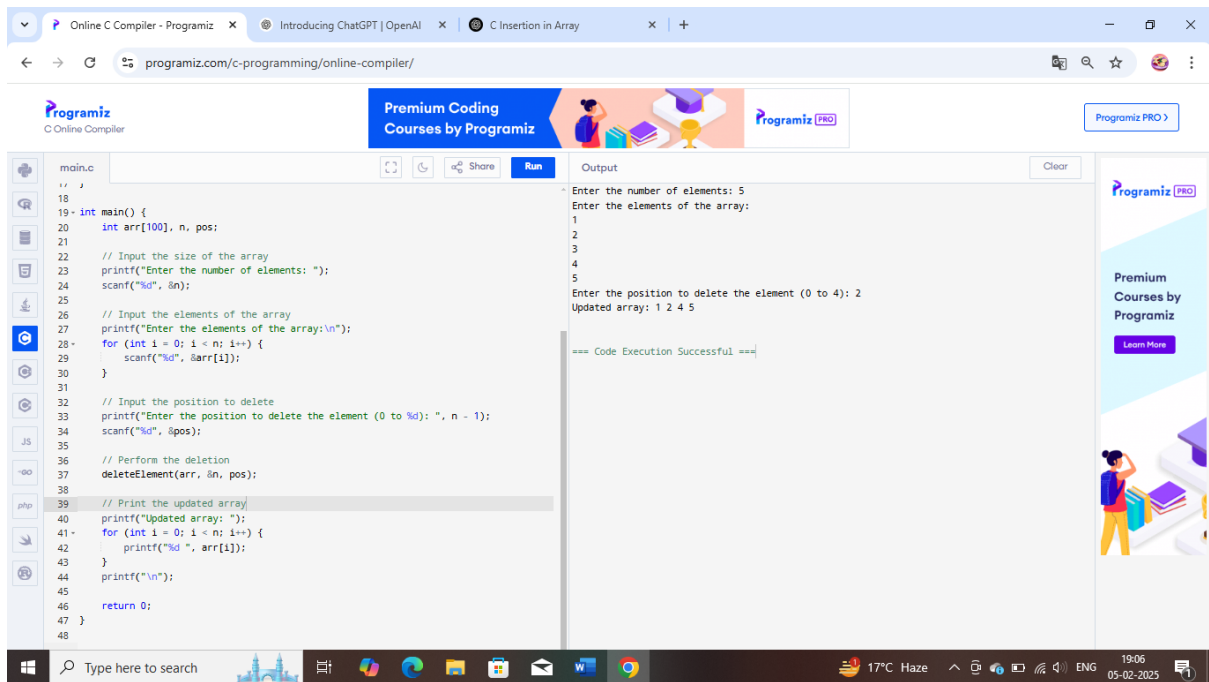
The screenshot displays the Programiz Online C Compiler web application. The browser's address bar shows the URL `programiz.com/c-programming/online-compiler/`. The page header includes the Programiz logo, a banner for 'Premium Coding Courses by Programiz', and a 'Programiz PRO' button. The main interface is divided into two panels: a code editor on the left and an output console on the right. The code editor, titled 'main.c', contains a C program designed to delete an element from an array. The program includes a `deleteElement` function that checks for valid positions, shifts elements to the left, and decreases the array size. The `main` function prompts the user for the number of elements, the elements themselves, and the position to delete, then calls the `deleteElement` function. The output console on the right shows the program's execution: it prompts for the number of elements (5), lists the elements (1, 2, 3, 4, 5), prompts for the position to delete (2), and displays the updated array (1 2 4 5). A green message at the bottom of the output console states '=== Code Execution Successful ==='. The Windows taskbar at the bottom of the screen shows the time as 17:36 on 09-02-2025.

```
main.c
1 #include <stdio.h>
2
3- void deleteElement(int arr[], int *size, int pos) {
4     // Check if the position is valid
5-     if (pos < 0 || pos >= *size) {
6         printf("Invalid position!\n");
7         return;
8     }
9
10    // Shift elements to the left to fill the gap
11-    for (int i = pos; i < *size - 1; i++) {
12        arr[i] = arr[i + 1];
13    }
14
15    // Decrease the size of the array
16    (*size)--;
17 }
18
19- int main() {
20     int arr[100], n, pos;
21
22     // Input the size of the array
```

Output

```
Enter the number of elements: 5
Enter the elements of the array:
1
2
3
4
5
Enter the position to delete the element (0 to 4): 2
Updated array: 1 2 4 5

=== Code Execution Successful ===
```

Q3. WRITE A PROGRAM IN C TO IMPLEMENT LINEAR SEARCHING IN 1-D ARRAYS.

SOL-

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

```
int main() {
```

```
    int arr[8] = {1, 2, 3, 4, 5, 6, 7, 8};
```

```
int search;

printf("Enter element you want to search: ");

scanf("%d", &search);

int found = 0;


for (int i = 0; i < 8; i++) {
    if (arr[i] == search) {
        found = 1;
        break; // Exit the loop once the element is
found
    }
}


if (found == 1) {
    printf("Your searched element was found");
} else {
    printf("not found---->");
}


return 0;
```

}

Q4. WRITE A PROGRAM TO IMPLEMENT SORTING IN 1-D ARRAYS.

SOL- #include <stdio.h>

```
int main() {
```

```
    int arr[6] = {34, 12, 54, 2, 89, 4214};
```

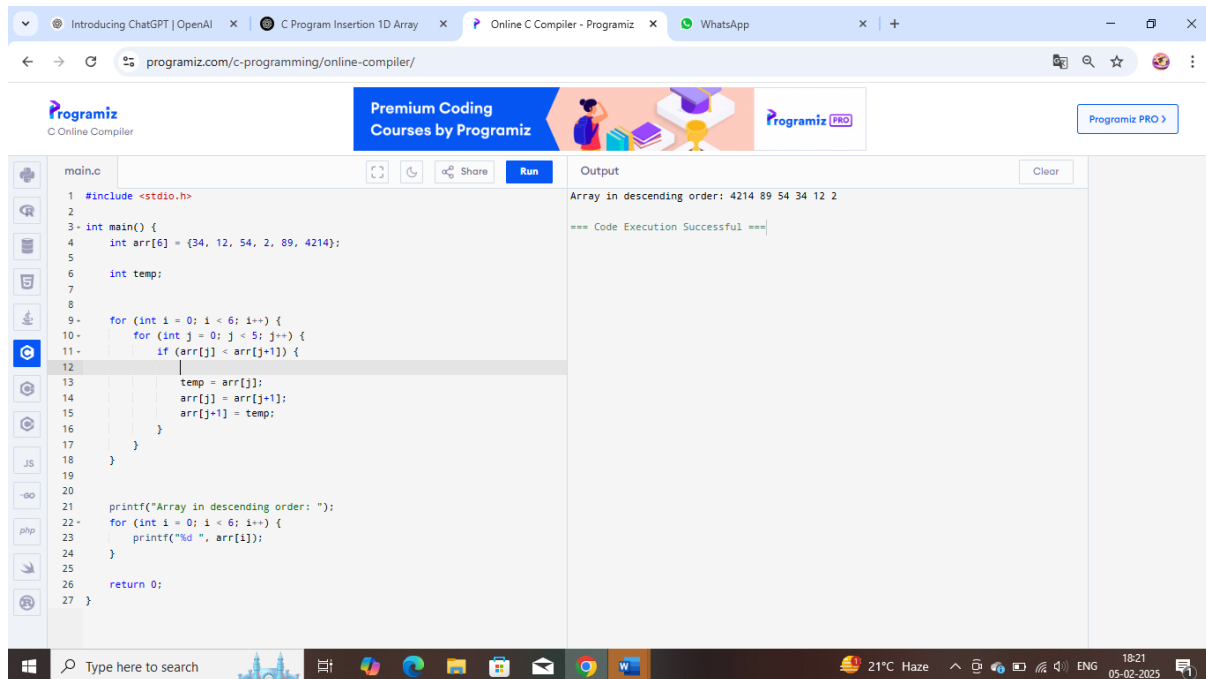
```
int temp;
```

```
for (int i = 0; i < 6; i++) {  
    for (int j = 0; j < 5; j++) {  
        if (arr[j] < arr[j+1]) {  
  
            temp = arr[j];  
            arr[j] = arr[j+1];  
            arr[j+1] = temp;  
        }  
    }  
}
```

```
printf("Array in descending order: ");  
for (int i = 0; i < 6; i++) {  
    printf("%d ", arr[i]);  
}
```

```
return 0;

}
```



The screenshot shows a web browser with multiple tabs, including 'Online C Compiler - Programiz'. The main content area displays a C program in a text editor. The program is a selection sort algorithm that sorts an array of integers in descending order. The array initially contains {34, 12, 54, 2, 89, 4214}. The output window shows the sorted array: 'Array in descending order: 4214 89 54 34 12 2'. Below the output, it states '=== Code Execution Successful ==='. The interface includes a 'Run' button and a 'Clear' button for the output. The bottom of the browser shows a Windows taskbar with various icons and system information like '21°C Haze' and '05-02-2025'.

```
main.c
1 #include <stdio.h>
2
3 int main() {
4     int arr[6] = {34, 12, 54, 2, 89, 4214};
5
6     int temp;
7
8
9     for (int i = 0; i < 6; i++) {
10        for (int j = 0; j < 5; j++) {
11            if (arr[j] < arr[j+1]) {
12
13                temp = arr[j];
14                arr[j] = arr[j+1];
15                arr[j+1] = temp;
16            }
17        }
18    }
19
20
21    printf("Array in descending order: ");
22    for (int i = 0; i < 6; i++) {
23        printf("%d ", arr[i]);
24    }
25
26    return 0;
27 }
```

Output

Array in descending order: 4214 89 54 34 12 2

=== Code Execution Successful ===

Q5. WRITE A PROGRAM IN C TO IMPLEMENT UPDATION IN 1-D ARRAYS.

SOL- #include <stdio.h>

// Function to update an element at a given position

```
void updateElement(int arr[], int size, int pos, int
new_value) {
    if (pos < 0 || pos >= size) {
        printf("Invalid position!\n");
    } else {
        arr[pos] = new_value; // Update the element at
the given position
        printf("Element at index %d has been updated to
%d.\n", pos, new_value);
    }
}
```

```
// Function to print the array
void printArray(int arr[], int size) {
    printf("Updated array: ");
    for (int i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}
```

```
int main() {  
    int arr[100], n, pos, new_value;  
  
    // Input the number of elements  
    printf("Enter the number of elements: ");  
    scanf("%d", &n);  
  
    // Input the elements of the array  
    printf("Enter the elements of the array:\n");  
    for (int i = 0; i < n; i++) {  
        scanf("%d", &arr[i]);  
    }  
  
    // Ask for the position and new value to update  
    printf("Enter the position to update (0 to %d): ", n -  
1);  
    scanf("%d", &pos);  
    printf("Enter the new value: ");  
    scanf("%d", &new_value);  
}
```

```
// Update the element at the specified position
```

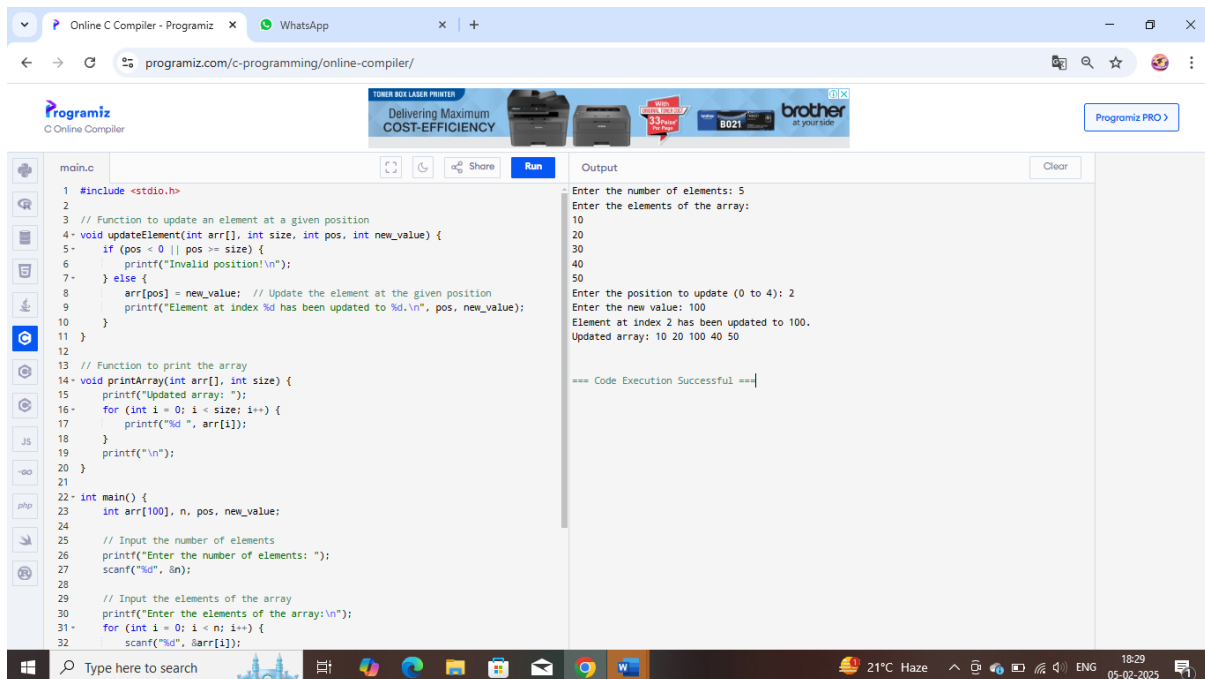
```
updateElement(arr, n, pos, new_value);
```

```
// Print the updated array
```

```
printArray(arr, n);
```

```
return 0;
```

```
}
```



The screenshot displays the Programiz Online C Compiler interface. The left pane shows the C source code for a program that updates an array element. The right pane shows the output of the program's execution.

Source Code (main.c):

```
1 #include <stdio.h>
2
3 // Function to update an element at a given position
4 void updateElement(int arr[], int size, int pos, int new_value) {
5     if (pos < 0 || pos >= size) {
6         printf("Invalid position!\n");
7     } else {
8         arr[pos] = new_value; // Update the element at the given position
9         printf("Element at index %d has been updated to %d.\n", pos, new_value);
10    }
11 }
12
13 // Function to print the array
14 void printArray(int arr[], int size) {
15     printf("Updated array: ");
16     for (int i = 0; i < size; i++) {
17         printf("%d ", arr[i]);
18     }
19     printf("\n");
20 }
21
22 int main() {
23     int arr[100], n, pos, new_value;
24
25     // Input the number of elements
26     printf("Enter the number of elements: ");
27     scanf("%d", &n);
28
29     // Input the elements of the array
30     printf("Enter the elements of the array:\n");
31     for (int i = 0; i < n; i++) {
32         scanf("%d", &arr[i]);
```

Output:

```
Enter the number of elements: 5
Enter the elements of the array:
10
20
30
40
50
Enter the position to update (0 to 4): 2
Enter the new value: 100
Element at index 2 has been updated to 100.
Updated array: 10 20 100 40 50

=== Code Execution Successful ===
```

The browser's address bar shows the URL `programiz.com/c-programming/online-compiler/`. The Windows taskbar at the bottom indicates the system time is 18:29 on 05-02-2025.

Online C Compiler - Programiz x WhatsApp x +

programiz.com/c-programming/online-compiler/

Programiz C Online Compiler

Premium Coding Courses by Programiz

Programiz PRO

Programiz PRO

main.c

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

```
//
printf("\n");
}
int main() {
    int arr[100], n, pos, new_value;

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

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

    // Ask for the position and new value to update
    printf("Enter the position to update (0 to %d): ", n - 1);
    scanf("%d", &pos);
    printf("Enter the new value: ");
    scanf("%d", &new_value);

    // Update the element at the specified position
    updateElement(arr, n, pos, new_value);

    // Print the updated array
    printArray(arr, n);

    return 0;
}
```

Run

Share

Output

Clear

```
Enter the number of elements: 5
Enter the elements of the array:
10
20
30
40
50
Enter the position to update (0 to 4): 2
Enter the new value: 100
Element at index 2 has been updated to 100.
Updated array: 10 20 100 40 50

=== Code Execution Successful ===
```

Type here to search

21°C Haze

18:30

05-02-2025