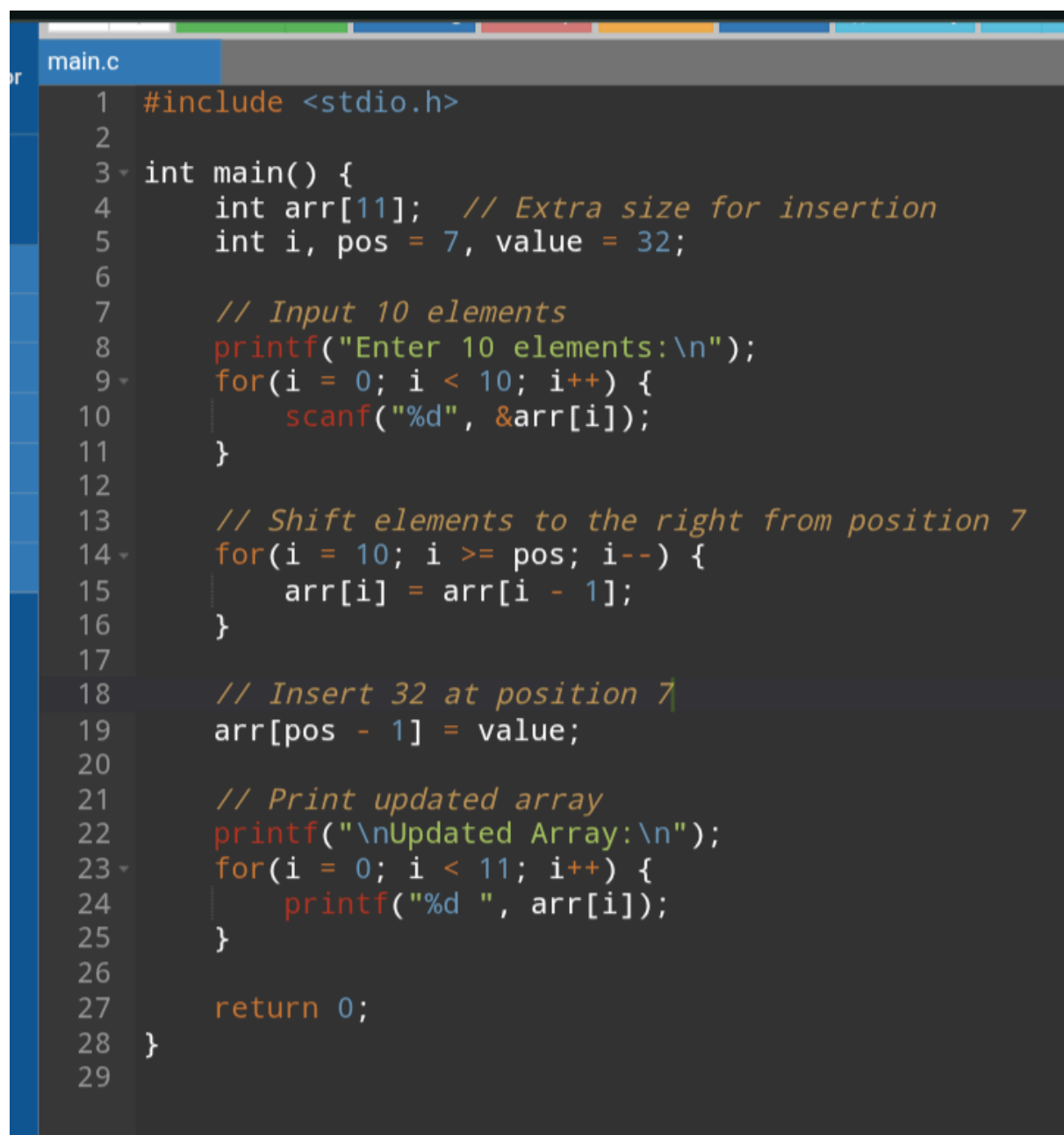


Experiment No.: 03**Experiment Name:** Insertion in Array

Objective: To learn how to insert a new element into an array by shifting elements and updating the array.

Task No: 01

Problem Statement: Take an array of size 10 from user input. Insert 32 at position 7 and print the updated array.

Source Code (C):The image shows a screenshot of a code editor with a dark background and light-colored text. The editor has a tab at the top labeled 'main.c'. The code is written in C and implements an array insertion. It starts with a preprocessor directive to include <stdio.h>. The main function declares an array 'arr' of size 11, a position 'pos' set to 7, and a value 'value' set to 32. It then prompts the user to enter 10 elements into the array. After input, it shifts elements from index 10 down to index 7 one position to the right. Finally, it inserts the value 32 at index 7 and prints the updated array of 11 elements.

```
1  #include <stdio.h>
2
3  int main() {
4      int arr[11]; // Extra size for insertion
5      int i, pos = 7, value = 32;
6
7      // Input 10 elements
8      printf("Enter 10 elements:\n");
9      for(i = 0; i < 10; i++) {
10         scanf("%d", &arr[i]);
11     }
12
13     // Shift elements to the right from position 7
14     for(i = 10; i >= pos; i--) {
15         arr[i] = arr[i - 1];
16     }
17
18     // Insert 32 at position 7
19     arr[pos - 1] = value;
20
21     // Print updated array
22     printf("\nUpdated Array:\n");
23     for(i = 0; i < 11; i++) {
24         printf("%d ", arr[i]);
25     }
26
27     return 0;
28 }
29
```

Figure 01: Input Array & Insert 32 at Position 32

Output:

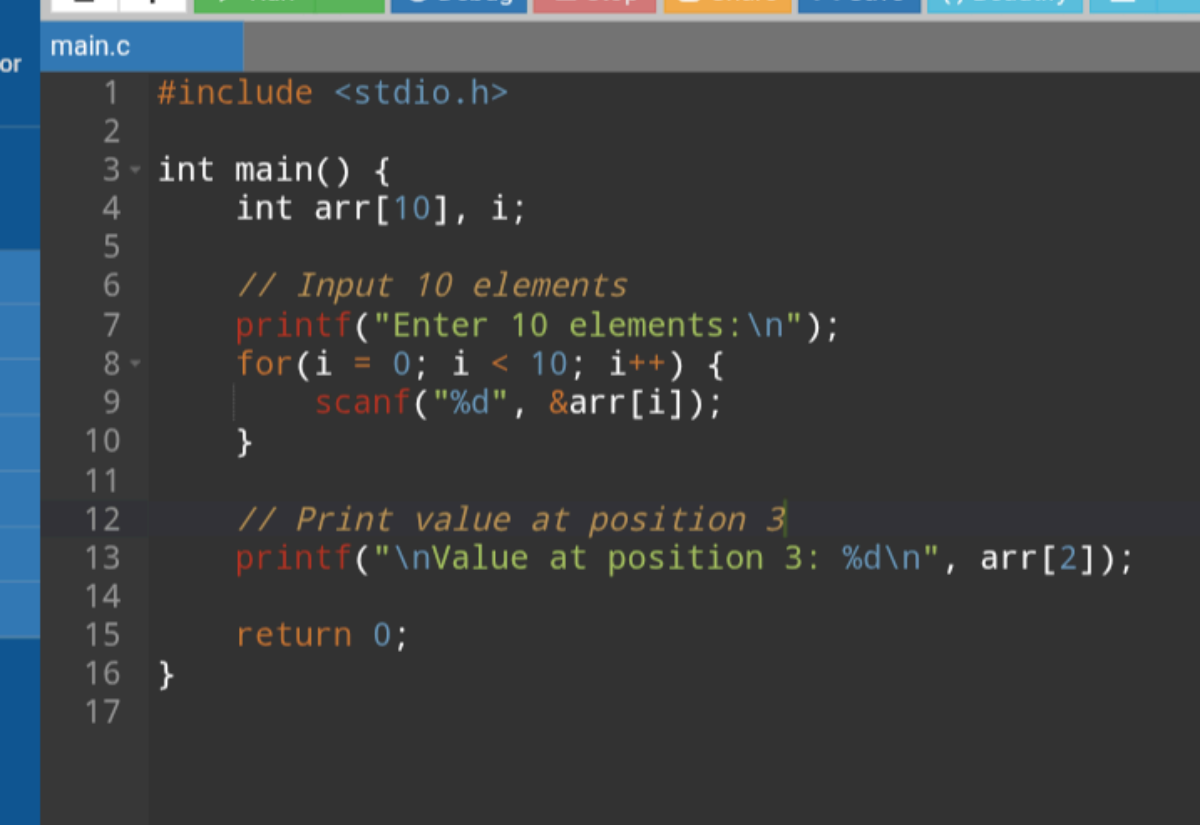

```

input
Enter 10 elements:
1 2 3 4 5 6 7 8 9 10

Updated Array:
1 2 3 4 5 6 32 7 8 9 10

...Program finished with exit code 0
Press ENTER to exit console.

```

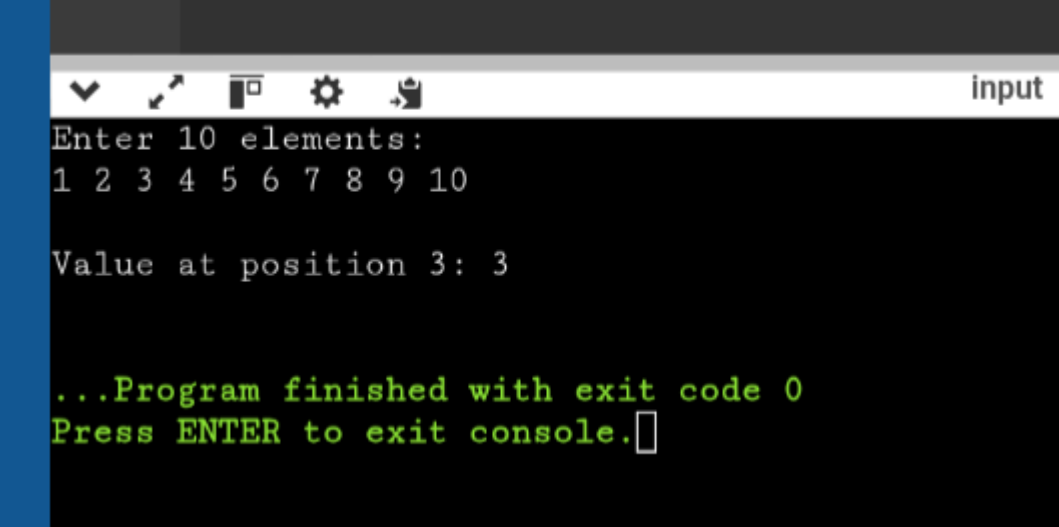
Figure 02: Output of Task 01**Task No:** 02**Problem Statement:** Print the value from the array at position 3.**Source Code (C):**


```

main.c
1  #include <stdio.h>
2
3  int main() {
4      int arr[10], i;
5
6      // Input 10 elements
7      printf("Enter 10 elements:\n");
8      for(i = 0; i < 10; i++) {
9          scanf("%d", &arr[i]);
10     }
11
12     // Print value at position 3
13     printf("\nValue at position 3: %d\n", arr[2]);
14
15     return 0;
16 }
17

```

Figure 03: Input Array & Print Position 3

Output:A screenshot of a terminal window with a dark background and a blue vertical bar on the left. The terminal has a title bar with icons and the word 'input' on the right. The output text is as follows:

```
Enter 10 elements:  
1 2 3 4 5 6 7 8 9 10  
  
Value at position 3: 3  
  
...Program finished with exit code 0  
Press ENTER to exit console.█
```

Figure 04: Output of Task 02

Discussion: This experiment has given me a clear and practical understanding of how insertion works in arrays. By carefully shifting the elements to the right, I was able to create space for the new value 32 at position 7 without losing any data. This highlighted the importance of indexing and the structured nature of arrays. Furthermore, accessing the value at position 3 strengthened my confidence in working with both zero-based and one-based indexing. I also realized how crucial these basic operations are, as they form the foundation for advanced data structures. Overall, this task not only improved my coding skills but also enhanced my logical thinking and efficiency in solving array-related problems.