# Practical: Insertion in Array

## Aim:

To write a C program to insert a new element at the end as well as at a given position in an array.

## Apparatus / Software Required:

- Computer System  
- GCC Compiler / Turbo C  
- Text Editor / IDE

## Theory:

An array is a collection of elements stored in contiguous memory locations.  
Insertion is the process of adding a new element at a desired location.  
  
Two cases:  
1. Insertion at End → Simply add element at the last index (increment size).  
2. Insertion at Position → Shift elements from the desired position to the right, then place the new element.

## Algorithm:

1. Start

2. Input size of array n

3. Input n elements

4. Input element to insert at end → place at arr[n], increment n

5. Input element and position to insert → shift elements right from that position, insert new element

6. Print updated array

7. Stop

## C Program:

#include <stdio.h>  
  
int main() {  
 int arr[100], n, i, element, pos;  
  
 // Step 1: Input size of array  
 printf("Enter the number of elements: ");  
 scanf("%d", &n);  
  
 // Step 2: Input array elements  
 printf("Enter %d elements:\n", n);  
 for(i = 0; i < n; i++) {  
 scanf("%d", &arr[i]);  
 }  
  
 // Insertion at End  
 printf("\nEnter element to insert at the end: ");  
 scanf("%d", &element);  
 arr[n] = element; // Insert at end  
 n++; // Increase size  
  
 printf("Array after inserting at end:\n");  
 for(i = 0; i < n; i++) {  
 printf("%d ", arr[i]);  
 }  
  
 // Insertion at a Given Position  
 printf("\n\nEnter element to insert: ");  
 scanf("%d", &element);  
 printf("Enter position (1 to %d): ", n+1);  
 scanf("%d", &pos);  
  
 if(pos < 1 || pos > n+1) {  
 printf("Invalid position!\n");  
 } else {  
 // Shift elements to the right  
 for(i = n; i >= pos; i--) {  
 arr[i] = arr[i-1];  
 }  
 arr[pos-1] = element; // Insert at given position  
 n++; // Increase size  
  
 printf("Array after inserting at position %d:\n", pos);  
 for(i = 0; i < n; i++) {  
 printf("%d ", arr[i]);  
 }  
 }  
  
 return 0;  
}

## Sample Output:

Enter the number of elements: 5  
Enter 5 elements:  
10 20 30 40 50  
  
Enter element to insert at the end: 60  
Array after inserting at end:  
10 20 30 40 50 60   
  
Enter element to insert: 25  
Enter position (1 to 7): 3  
Array after inserting at position 3:  
10 20 25 30 40 50 60

## Conclusion:

Insertion at the end is simple (O(1)) as it requires no shifting.  
Insertion at a given position requires shifting elements to the right, leading to O(n) time complexity.