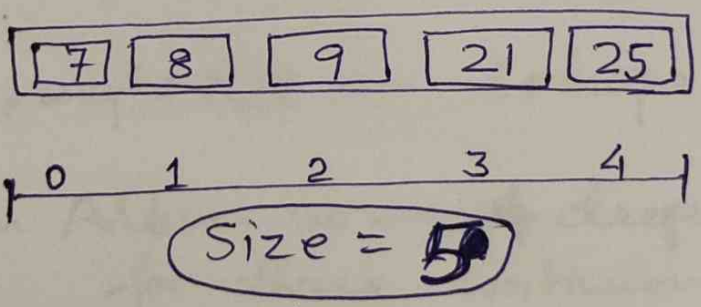
Basic Operations on an Array →

* Traverse – print all the elements one by one.
* Insertion – Add an element at given index.
* Deletion – Delete an element at given index
* Search – search an element using given index or by value.
* Update – Update an element at given index.

Example :-

```
int arr[5] = {7, 8, 9, 21, 25};
```

Here,



→ Here, index starts with 0.

→ Array length is 5 which means it can store 5 elements.

→ Each element can be accessed via its index. For example, we can fetch element at index 3 as 21.

Insertion Operation – This operation allows us to insert one or more data elements into an array. Based on the requirements, new element can be added at the beginning, end or any given index of array.

Let us see practical implementation of insert –

Algorithm –

Let LA is a linear array which is unsorted with N elements and K is a +ve integer such that $K <= N$.

The algorithm to insert ITEM into the K^{th} position of LA is :–

1. Start
2. Set J = N
3. Set N = N + 1
4. Repeat steps 5 and 6 while J >= K
5. Set LA[J+1] = LA[J]
6. Set J = J - 1;
7. Set LA[k] = ITEM
8. Stop

The implementation of this algorithm into program is as follows :–

```
#include <stdio.h>
void main(){
    int LA[] = {1, 7, 9, 21, 25};
    int item = 10, k = 3, n = 5;
    int i=0, j=n;
  printf("The original array elements are :\n");
  for(i=0;  i<n;  i++){
    printf("LA[%d] = %d \n", i, LA[i]);
  }
  n=n+1;
  while(j >= k){
          LA[j+1] = LA[j];
          j = j-1;
        }
    LA[k] = item;
printf("The array elements after insertion :\n");
for(i=0;  i<n;  i++){
 printf("LA[%d] = %d \n", i, LA[i]);
  }
  getch();
}
```

output of the program :-

The original array elements are :-

LA[0] = 1
LA[1] = 7
LA[2] = 9
LA[3] = 21
LA[4] = 25

The array elements after insertion :

LA[0] = 1
LA[1] = 7
LA[2] = 9
LA[3] = 10
LA[4] = 21
LA[5] = 25