EXPERIMENT - 3

Data Structures

Aim

To perform Traversal, Insertion and Deletion operations on an array

Syeda Reeha Quasar
14114902719
3C7

Name - Syeda Reeha Quasar **Roll no. –** 14114802719 **Group –** C7

EXPERIMENT – 3

AIM: To perform Traversal, Insertion and Deletion operations on an array.

THEORY

Traversing an array means accessing each and every element of the array for a specific purpose. Traversing the data elements of an array A can include printing every element, counting the total number of elements, or performing any process on these elements.

An array is a collection of items stored at contiguous memory locations.

PROGRAM 1

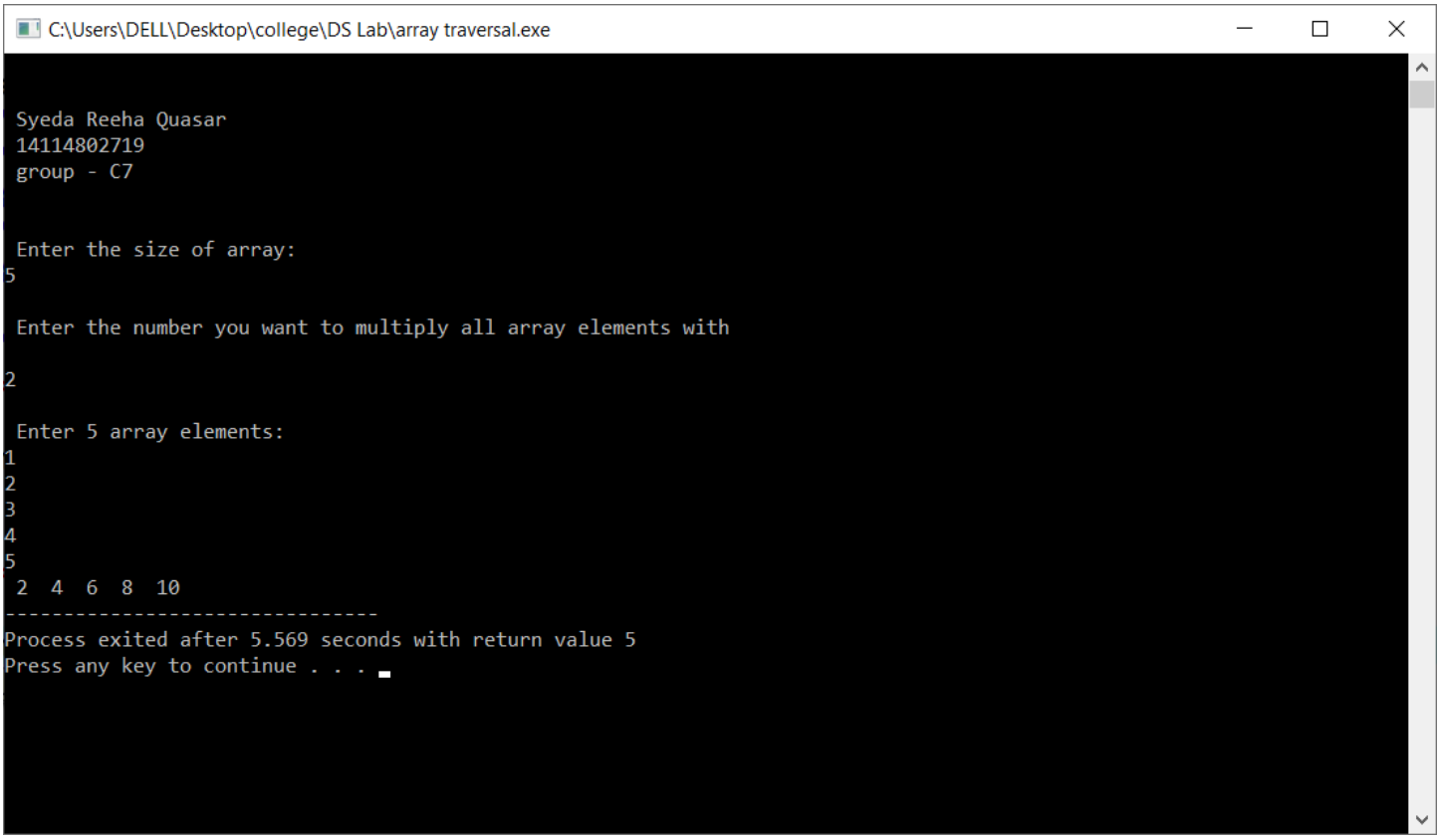
Array Traversal Program

Source code:

```
#include <stdio.h>

int main()
{
	// declaring variables
	int array[100], i, n, m;

	//personal information
	printf("\n\n Syeda Reeha Quasar \n 14114802719\n group - C7 \n\n");

	//declaring and asking for size of array
	printf("\n Enter the size of array: \n");
	scanf("%d", &n);

	//asking the number which you want to multiply the elements with
	printf("\n Enter the number you want to multiply all array elements with\n\n");
	scanf("%d", &m);

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

	// printing the array after traversal
	for (i = 0; i < n; i++)
```

```
        {
                printf(" %d ", array[i]*m);
        }
}
```

OUTPUT



PROGRAM 2

INSERTION

Approach

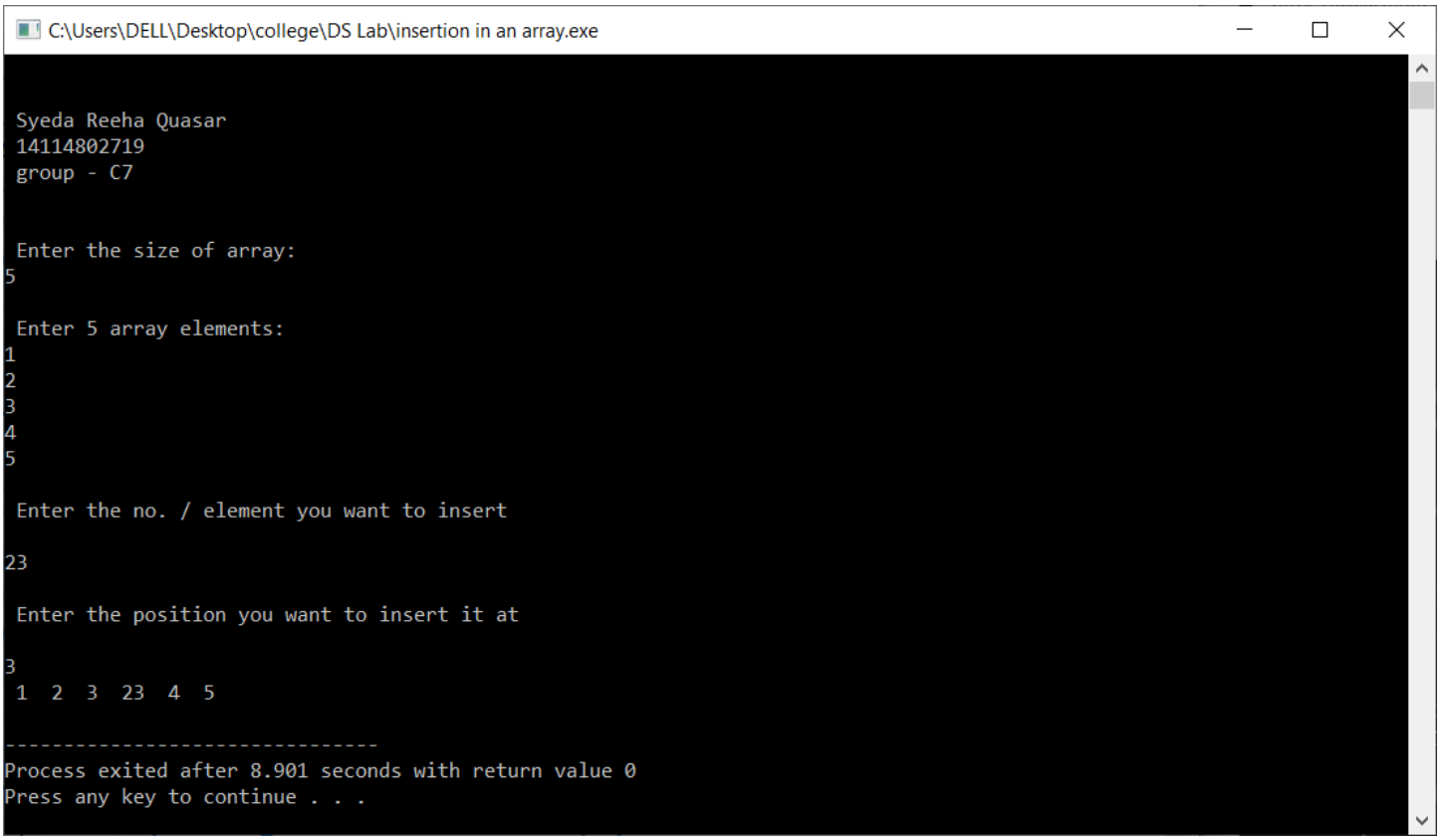
1. First get the element to be inserted, say x
2. Then get the position at which this element is to be inserted, say pos
3. Then shift the array elements from this position to one position forward, and do this for all the other elements next to pos.
4. Insert the element x now at the position pos, as this is now empty.

Source Code:

```
#include <stdio.h>


int main()
{
	// variables declaration
	int array[100], i, n, m, p;
	// my details
	printf("\n\n Syeda Reeha Quasar \n 14114802719\n group - C7 \n\n");
	// size of the array
	printf("\n Enter the size of array: \n");
	scanf("%d", &n);

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

```
	//insertion details
	printf("\n Enter the no. / element you want to insert\n\n");
	scanf("%d", &m);
	printf("\n Enter the position you want to insert it at\n\n");
	scanf("%d", &p);

	// increasing the size of the array
	n = n + 1;

	//moving all elements after position entered one place forward
	for (i = n; i >= p; i--)
	{
		array[i] = array[i - 1];
	}

	//placing the lement m at p position
	array[p] = m;

	for (i = 0; i < n; i++)
	{
		printf(" %d ", array[i]);
	}
	printf("\n");

	return 0;
}
```

OUTPUT


C:\Users\DELL\Desktop\college\DS Lab\insertion in an array.exe
Syeda Reeha Quasar
14114802719
group - C7
Enter the size of array:
5
Enter 5 array elements:
1
2
3
4
5
Enter the no. / element you want to insert
23
Enter the position you want to insert it at
3
1 2 3 23 4 5
Process exited after 8.901 seconds with return value 0
Press any key to continue . . .

PROGRAM 2

DELETION

Approach

We first search 'x' in array, then elements that are on right side of x to one position back.

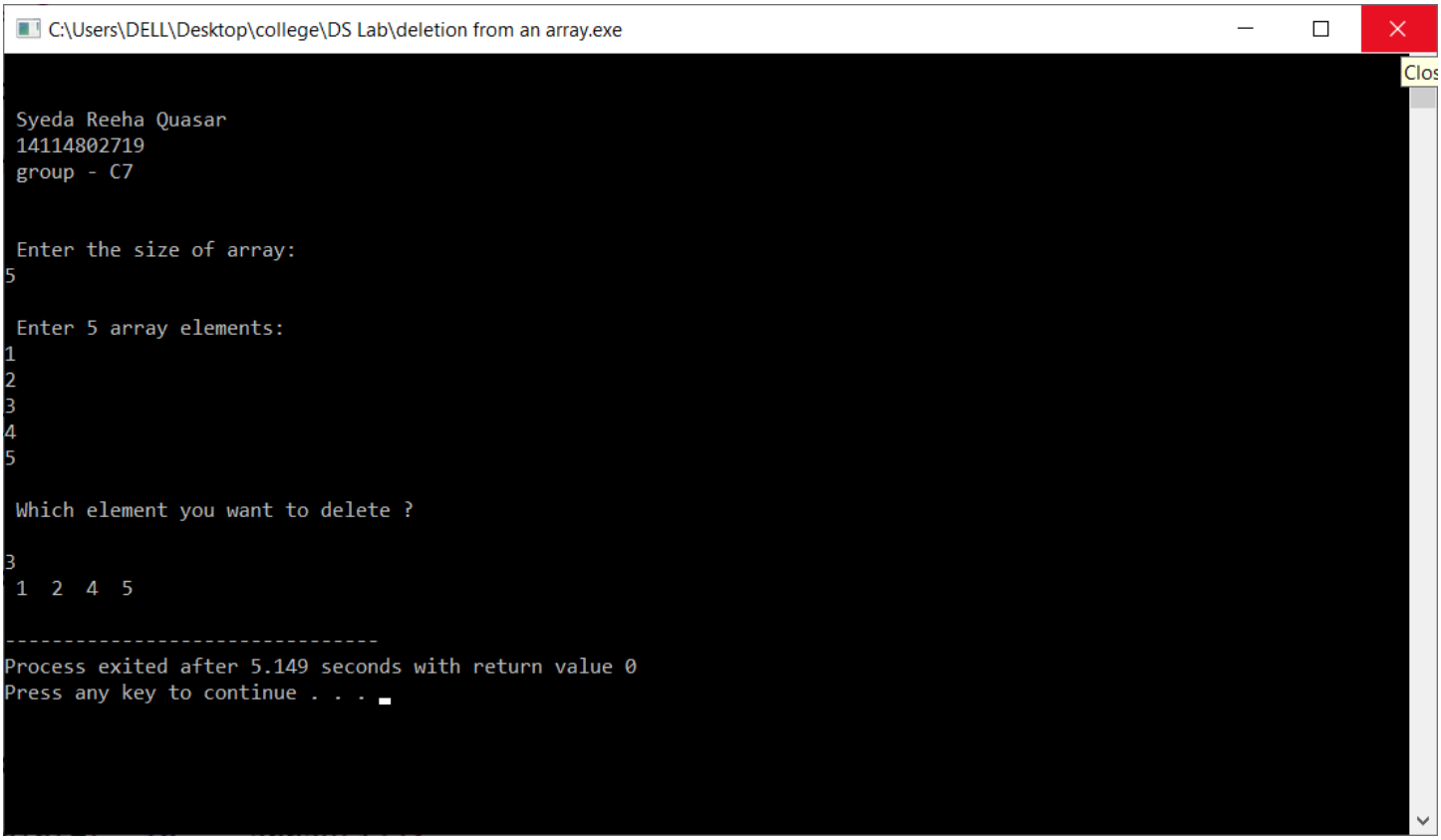
Source Code:

```
#include <stdio.h>

int main()
{
	// variables declaration
	int array[100], i, n, m, p, flag;
	// my details
	printf("\n\n Syeda Reeha Quasar \n 14114802719\n group - C7 \n\n");
	// size of the array
	printf("\n Enter the size of array: \n");
	scanf("%d", &n);

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

	//deletion details
	printf("\n Which element you want to delete ? \n\n");
	scanf("%d", &m);
```

```
flag = 0;

// finding the element
for (i = 0; i < n; i++)
{
        if (array[i] == m)
        {
                flag = i;
                break;
        }
}

// checking if element is found or not
if (flag != 0)
{
        n = n - 1;

        for (p = flag; p < n; p++)
        {
                array[p] = array[p + 1];
        }

        for (i = 0; i < n; i++)
        {
        printf(" %d ", array[i]);
        }

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

```
        // printing not found if element is not there
        else
        {
                printf("\n Element was not found in the array \n");
        }

        return 0;
}
```

OUTPUT



Viva Questions

Q1. What are the operations on an array?

Ans. Array operations are operations that are performed on arrays term by term or element by element.

Basic Operations supported by an array:

1. Traverse – access/print all the array elements one by one.
2. Insertion – Adds an element at the given index.
3. Deletion – Deletes an element/element at the given index.
4. Search – Searches an element using the given index or by the value.

Q2. What is deletion in an array?

Ans. Deletion refers to removing an existing element from the array and re-organizing all elements of an array.

Q3. What is insertion in an array?

Ans. When the insertion happens at any position/index, it causes all the existing data items below it to shift one step downward. Finally, it inserts an element at the required position/index of an array.

Q4. What is traversing in an array?

Ans. In traversing operation of an array, each element of an array is accessed exactly for once for processing. This is also called visiting of an array.