LAB 06:

QUICK SORT ARRAY IN C PROGRAMMING

Code:

```
#include <stdio.h>

void swap(int *a, int *b) {
   int temp = *a;
   *a = *b;
   *b = temp;
}

int partition(int arr[], int low, int high) {
   int pivot = arr[high];
   int i = (low - 1);

   for (int j = low; j <= high - 1; j++) {
      if (arr[j] < pivot) {
         i++;
         swap(&arr[i], &arr[j]);
      }
   }
   swap(&arr[i + 1], &arr[high]);
   return (i + 1);
}

void quickSort(int arr[], int low, int high) {
   if (low < high) {
      int pi = partition(arr, low, high);
```

```
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}

int main() {
    int n;
    printf("Enter the number of elements in the array: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter %d integers for the array:\n", n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("Original array: ");
    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");

    quickSort(arr, 0, n - 1);

    printf("Sorted array: ");
    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");

    return 0;
}
```

OUTPUT:

```
Enter the number of elements in the array: 5
Enter 5 integers for the array:
55 22 66 86 45
Original array: 55 22 66 86 45
Sorted array: 22 45 55 66 86

Process returned 0 (0x0)   execution time : 19.781 s
Press any key to continue.
```

```
Enter the number of elements in the array: 4
Enter 4 integers for the array:
43 78 132 635
Original array: 43 78 132 635
Sorted array: 43 78 132 635

Process returned 0 (0x0)   execution time : 16.844 s
Press any key to continue.
```