**Practical-3**

**AIM:** To implement quick sort in C programming.

**SOFTWARE REQUIRED:** Vs Code

**PSEUDO CODE:**

quickSort(arr, beg, end)

if (beg < end)

pivotIndex = partition(arr,beg, end)

quickSort(arr, beg, pivotIndex)

quickSort(arr, pivotIndex + 1, end)

partition(arr, beg, end)

set end as pivotIndex

pIndex = beg - 1

for i = beg to end-1

if arr[i] < pivot

swap arr[i] and arr[pIndex]

pIndex++

swap pivot and arr[pIndex+1]

return pIndex + 1

**CODE:**

#include <stdio.h>

void swap(int \*a, int \*b) {

    int temp = \*a;

    \*a = \*b;

    \*b = temp;

}

int function(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 pivotIndex = function(arr, low, high);

        quicksort(arr, low, pivotIndex - 1);

        quicksort(arr, pivotIndex + 1, high);

    }

}

int main() {

    int n;

    printf("\nName: Ananta Walli");

    printf("\nEnrollment No: A2305221322");

    printf("\nPlease enter the number of elements that should be in array: ");

    scanf("%d", &n);

    int arr[n];

    printf("Please enter the elements of array:\n");

    for (int i = 0; i < n; i++) {

        scanf("%d", &arr[i]);

    }

    printf("The inputted array will be: ");

    for (int i = 0; i < n; i++) {

        printf("%d ", arr[i]);

    }

    quicksort(arr, 0, n - 1);

    printf("\nThe sorted array will be like: ");

    for (int i = 0; i < n; i++) {

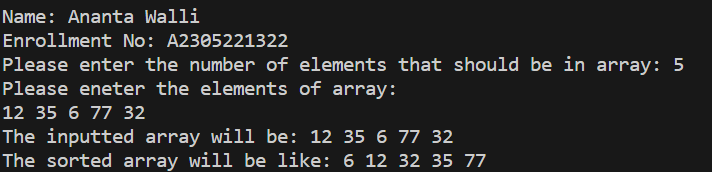
        printf("%d ", arr[i]);

    }

    return 0;

}

**OUTPUT:**

****

**TIME COMPLEXITY:** The time complexity should be:

1. Best case: nlogn
2. Worst case: n^2
3. Average case: nlogn.

**RESULT:** The above code implements the quick sort in C programming.