**DAA PRACTICAL**

**Name:Virendra Kashinath Bagul Roll no. 05**

**QUICK SORT PROGRAM**

#include<iostream>

using namespace std;

void quick(int a[], int l, int up); // Function to perform quick sort.

int partition(int a[], int l, int up); // Function to partition the array.

int main()

{

int n;

cout<<"enter the size of an array"<<endl; // Prompting user to enter the size of the array.

cin>>n;

int arr[n];

cout<<"enter array elements"<<endl;

for(int i=0;i<n;i++)

{

cin>>arr[i]; // Reading array elements from the user.

}

int low = 0;

int up = n-1;

quick(arr, low, up); // Calling quick sort function to sort the array.

cout << "sorted elements are" << endl;

for (int i = 0; i <= up; i++) {

cout << arr[i] << " "; // Printing sorted elements

}

cout << endl;

return 0;

}

void quick(int a[], int l, int up) {

if (l >= up) {

return;

}

int pvtloc = partition(a, l, up); // Partitioning the array and obtaining pivot location.

quick(a, l, pvtloc - 1); //left sublist

quick(a, pvtloc + 1, up); //right sublist

}

int partition(int a[], int l, int up) {

if (l >= up) {

return l;

}

int temp, pvt;

int i = l + 1;

int j = up;

pvt = a[l];

while (i <= j) {

while (a[i] < pvt) {

i++; // Moving towards right until element greater than or equal to pivot is found.

}

while (a[j] > pvt) {

j--;

}

if (i < j) {

temp = a[i];

a[i] = a[j];

a[j] = temp; // Swapping elements if they are out of place.

i++;

j--;

} else {

i++;

}

}

// Swap pivot with element at position j

temp = a[l];

a[l] = a[j];

a[j] = temp; // Placing pivot at its correct position.

return j;

}

//OUTPUT

