**Title: Implementation of Quick Sort.**

**Abstract:** Quick sort is a highly efficient sorting algorithm and is based on partitioning of array of data into smaller arrays. A large array is partitioned into two arrays one of which holds values smaller than the specified value, say pivot, based on which the partition is made and another array holds values greater than the pivot value. Quick sort is the fastest internal sorting algorithm with the time complexity O (n log n). The basic algorithm to sort an array a[ ] of n elements can be described [recursively](https://www.thecrazyprogrammer.com/2013/10/recursive-permutation-in-c.html) as follows:

## Methodology: Quick Sort Algorithm

1. If n < = 1, then return.

2. Pick any element V in a[]. This is called the pivot.

3. Rearrange elements of the array by moving all elements xi > V right of V and all elements x­i < = V left of V. If the place of the V after re-arrangement is j, all elements with value less than V, appear in a[0], a[1] . . . . a[j – 1] and all those with value greater than V appear in a[j + 1] . . . . a[n – 1].

4. Apply quick sort recursively to a[0] . . . . a[j – 1] and to a[j + 1] . . . . a[n – 1].

void quick\_sort(l,h)

{

if(l<u)

{

j=partition(a,l,u);

quick\_sort(a,l,j-1);

quick\_sort(a,j+1,u);

}

}

  partition(l,b)

{

v=a[l];

i=l;

j=u+1;

do

{

do

i++;

while(a[i]<v&&i<=u);

do

j--;

while(v<a[j]);

Swapa([l];a[j])

}while(i<j);

a[l]=a[j];

a[j]=v;

return(j);

}

**Discussion:**

* Quicksort is a divide and conquer algorithm.
* Pick an element from the array, this element is called as pivot element.
* Divide the unsorted array of elements in two arrays with values less than the pivot come in the first sub array.
* Elements with values greater than the pivot come in the second sub-array (equal values can go either way).
* This step is called the partition operation.
* Recursively repeat the step 2(until the sub-arrays are sorted).
* The sub-array of elements with smaller values and separately to the sub-array of elements with greater values.

**Source Code:**

|  |
| --- |
| #include <stdio.h>    void quick\_sort(int[],int,int);  int partition(int[],int,int);    int main()  {  int a[50],n,i;  printf("How many elements?");  scanf("%d",&n);  printf("\nEnter array elements:");    for(i=0;i<n;i++)  scanf("%d",&a[i]);    quick\_sort(a,0,n-1);  printf("\nArray after sorting:");    for(i=0;i<n;i++)  printf("%d ",a[i]);    return 0;  }    void quick\_sort(int a[],int l,int u)  {  int j;  if(l<u)  {  j=partition(a,l,u);  quick\_sort(a,l,j-1);  quick\_sort(a,j+1,u);  }  }    int partition(int a[],int l,int u)  {  int v,i,j,temp;  v=a[l];  i=l;  j=u+1;    do  {  do  i++;    while(a[i]<v&&i<=u);    do  j--;  while(v<a[j]);    if(i<j)  {  temp=a[i];  a[i]=a[j];  a[j]=temp;  }  }while(i<j);    a[l]=a[j];  a[j]=v;    return(j);  } |

