Algorithm of Quicksort

Suppose there is an array of size n.

Steps :-

1. Take middle element as pivot element.
2. Two index variable, say, p and q are taken which are positioned at 0th and (n-1)th location respectively. The p searches for equal or greater value than pivot where q search for smaller or equal value than pivot. The p searches by moving forward and q searches by moving backward.
3. Whenever p & q find desired values they interchange them.
4. P = p+1
5. q = q-1
6. Repeat steps 3 to 5 until P<q
7. Now array is divided into Two Sub-Arrays - Left sub array from 0 to q and Right sub Array from p to (n-1)th location.
8. Repeat steps 1 to 7 for each sub array until each sub-array is left with only one element.
9. END

```
#define n 10
#include <stdio.h>
#include<conio.h>
int main()
{
void quicksort(int a, int lower, int upper);
int a[n]= {7,1,5,2,4,3,14,18,2,10};

int lower, upper;
lower = 0;
upper = n-1;
quicksort( a, lower, upper);
// print result
for(int i=0; i<n; i++)
printf("%d\n", a[i]);
getch();
return 0;
}
void quicksort(int *a, int lower, int upper)
{
int p, q, pivot, temp;
p= lower;
q= upper;
pivot = a[(p+q)/2];
while(p<q)
{
while(a[p] > pivot)
p++;
while(a[q] < pivot)
q--;
if(p<=q)
{
temp= a[p];
a[p]= a[q];
a[q]= temp;
```

p++;

q--;

}

}

if(q>lower)

quicksort(a, lower, q);

if (p <upper)

quicksort(a, p, upper);

}