```
#include <iostream>
#include <stdio.h>
#include <fstream>
#define arrlen 5
using namespace std;
int ww=64; ofstream out;

void quickSort(int arr[],int low,int high);
int partiton(int arr[],int low,int high);
void printsort(int arr[],int low=0,int high=arrlen-1,int check=0)
{if(check==0){cout<<"\n________________________________________________________
_______\n\t\tafter midway partition
arrangement\n_______________________________________________________________
___\n";

out<<"\n_______________________________________________________________\n\t\taft
er midway partition
arrangement\n_______________________________________________________________
___\n";
}else
{
   cout<<"\n-Considering this subarray now-\n";
   out<<"\n-Considering this subarray now-\n";

}
   for(int i=low;i<=high;i++)
     {
        cout<<arr[i]<<"\t";out<<arr[i]<<"\t";
     }
}

int main()
{
int a[10];out.open("quicksort_by_beast.txt");
out<<"\n\t\t\t__________________________\n\t\t\t::QUICKSORT WITH
STEPS::\n\t\t\t__________________________\n";
cout<<"Enter your elements\n";
for(int i=0;i<arrlen;i++)
   cin>>i[a];

   cout<<"Your unsorted array is ==\n";
out<<"Your unsorted array is ==\n";
```

```
for (int i=0;i<arrlen;i++)
  { cout<<a[i]<<"\t";
   out<<a[i]<<"\t";

  }
    cout<<"===================\n\n\n";out<<"===================\n\n\n";

quickSort(a,0,arrlen-1);

    cout<<"\n\nYour sorted array is ==\n";out<<"\n\nYour sorted array is ==\n";

for (int i=0;i<arrlen;i++)
  {
     cout<<i[a]<<"\t"; out<<i[a]<<"\t";
  }
out.close();
}

void quickSort(int arr[], int low, int high)
{
    if(low<high){cout<<endl<<(char)(++ww)<<endl;out<<endl<<"step :"<<(char)(ww)<<endl;
    int pi=partiton(arr,low,high);
    quickSort(arr,low,pi-1);
    quickSort(arr,pi+1,high);

}
}
int partiton(int arr[],int low,int high){
    /*
    this function is :
1. make middle element pivot.as mid=(low+high)/2 and value at mid is assigned to variable pivot

2. make a variable i and initialize with mid+1
3. Now run a loop from low index to high index
    case [a]
    IF any element which is smaller than pivot element and is placed right of the pivot element,

    then make it left of the pivot element to do so we need some swaps.
    first we use a temporary variable to hold the smaller number then we replace smaller number
by mid+1
    element(this element is just right of pivot and it is greater than pivot )
    now we shift our mid element(pivot ) to mid+1 location(so as to make a space )
    NOW THE SMALLER NUMBER IS BROUGHT TO MID LOCATION.(where previously our
mid was .)
    UPDATE THE MID POSITION AS IT IS SHIFTED BY 1 index.


    case [B]
    if any number which is greater than pivot element and it is left of pivot
       then we simply run a loop to find a smaller number right of pivot and swap it with pivot .

*/
    int mid=(low+high)/2;
    int pivot=arr[mid];
    int i=mid+1;
    cout<<"\n++++++++++++++++++++++partition +++++++++++++++\n low="<<low<<"\t
high="<<high<<"\nmiddle element ="<<mid<<" with value="<<pivot<<endl;
out<<"\n++++++++++++++++++++++partition +++++++++++++++\n low="<<low<<"\t
high="<<high<<"\nmiddle element ="<<mid<<" with value="<<pivot<<endl;
printsort(arr,low,high,1);
    for(int j=low;j<=high;j++)


    {
      /// running the looop from low to high
      if(arr[j]<pivot&&j>mid)
      {cout<<" \n a element smaller than mid and right of mid is found value="<<arr[j]<<" at index
="<<j<<endl;
       out<<" \n a element smaller than mid and right of mid is found value="<<arr[j]<<" at index
="<<j<<endl;


    int temp=arr[j];/// holds smaller number right of pivot
    arr[j]=arr[mid+1]; /// swap the smaller number location with item greater than pivot and that is
right of pivot element
    arr[mid+1]=arr[mid];///now the rightmost position of [mid] is swapped with pivot element
    /* making space to hold the smaller number coming to be placed left of mid element*/
    arr[mid]=temp;/// now smaller number is left of the pivot element.

    mid++;/// finally we know our middle element has got a new location after accommodating a
number from right side
printsort(arr,low,high); /// print the mid sorted subarray
      }
     if(arr[j]>pivot&&j<mid)
```

```
       {cout<<" \n a element greater  than mid and left  of mid is found ="<<arr[j]<<" at index
="<<j<<endl;
        out<<" \n a element greater  than mid and left  of mid is found ="<<arr[j]<<" at index
="<<j<<endl;

      int temp=pivot;

      arr[mid]=arr[j];
      arr[j]=pivot;
      mid=j;
printsort(arr,low,high);


       }




    }
return mid;
}
```