MergeSort

#include<iostream>

#include<vector>

using namespace std;

void merge(int arr[],int si,int mid, int ei){

vector<int> temp;

int i=si;

int j=mid+1;

while(i<=mid && j<=ei){

if(arr[i]<=arr[j]){

temp.push\_back(arr[i++]);

}else{

temp.push\_back(arr[j++]);

}

}

while(i<=mid) temp.push\_back(arr[i++]);

while(j<=ei) temp.push\_back(arr[j++]);

for(int idx=si,x=0;idx<=ei;idx++){

arr[idx]=temp[x++];

}

}

void mergeSort(int arr[],int si, int ei){

if(si>=ei){

return ;

}

int mid = si +(ei-si)/2;

mergeSort(arr,si,mid);//left half

mergeSort(arr,mid+1,ei);//right half

merge(arr,si,mid,ei);

}

void Print(int arr[],int n){

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

cout<<arr[i]<<" ";

}

cout<<endl;

}

int main() {

// code here

cout<<"Enter the no of elements in the array"<<endl;

int n; cin>>n;

int arr[n];

cout<<"Enter the elements"<<endl;

for(int i=0;i<n;i++) cin>>arr[i];

mergeSort(arr,0,n-1);

Print(arr,n);

return 0;

}

Quick Sort

#include<iostream>

using namespace std;

int partition(int arr[],int si,int ei){

int i=si-1,pivot=arr[ei];

for(int j=si;j<ei;j++){

if(arr[j]<=pivot)

{

i++;

swap(arr[i],arr[j]);

}

}

i++;

swap(arr[i],arr[ei]);

return i;

}

void QiuckSort(int arr[],int si,int ei){

if(si>=ei) return ;

int partition\_idx=partition(arr,si,ei);

QiuckSort(arr,si,partition\_idx-1);

QiuckSort(arr,partition\_idx+1,ei);

}

void Print(int arr[],int n){

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

cout<<arr[i]<<" ";

}

cout<<endl;

}

int main() {

// code here

cout<<"Enter the no of elements in the array"<<endl;

int n; cin>>n;

int arr[n];

cout<<"Enter the elements"<<endl;

for(int i=0;i<n;i++) cin>>arr[i];

QiuckSort(arr,0,n-1);

Print(arr,n);

return 0;

}