**MERGE SORT**

#include<bits/stdc++.h>

#include<cmath>

using namespace std;

int A[10000];

void Merge(int arr[], int l, int m, int h)

{

int n1 = m - l + 1;

int n2 = h - m;

int L[n1], R[n2];

for(int i = 0; i < n1; i++)

L[i] = arr[l + i];

for(int j = 0; j < n2; j++)

R[j] = arr[m + 1 + j];

int i = 0;

int j = 0;

int k = l;

while (i < n1 && j < n2)

{

if (L[i] <= R[j])

{

arr[k] = L[i];

i++;

}

else

{

arr[k] = R[j];

j++;

}

k++;

}

while (i < n1)

{

arr[k] = L[i];

i++;

k++;

}

while (j < n2)

{

arr[k] = R[j];

j++;

k++;

}

}

void Merge\_sort(int arr[], int low, int high)

{

if (low < high)

{

int mid = (low + high )/ 2;

Merge\_sort(arr, low, mid);

Merge\_sort(arr, mid + 1, low);

Merge(arr, low, mid, high);

}

}

int main()

{

int n;

cout << "Enter the size of array : " << endl;

cin >> n;

cout<<"\nEnter data : "<<endl;

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

cin >> A[i];

}

Merge\_sort(A, 0, n-1);

cout<<"\n\nSorted array : \n";

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

{

cout<<A[i]<<endl;

}

cout<<endl;

return 0;

}

**BINARY SEARCH**

#include <iostream>

using namespace std;

int binarySearch(int arr[], int l, int h, int val)

{

if (h>= l) {

int mid = (l + h) / 2;

if (arr[mid] == val)

return mid;

if (arr[mid] > val)

return binarySearch(arr, l, mid - 1, val);

return binarySearch(arr, mid + 1, h, val);

}

return -1;

}

void Merge(int arr[], int l, int m, int h)

{

int n1 = m - l + 1;

int n2 = h - m;

int L[n1], R[n2];

for(int i = 0; i < n1; i++)

L[i] = arr[l + i];

for(int j = 0; j < n2; j++)

R[j] = arr[m + 1 + j];

int i = 0;

int j = 0;

int k = l;

while (i < n1 && j < n2)

{

if (L[i] <= R[j])

{

arr[k] = L[i];

i++;

}

else

{

arr[k] = R[j];

j++;

}

k++;

}

while (i < n1)

{

arr[k] = L[i];

i++;

k++;

}

while (j < n2)

{

arr[k] = R[j];

j++;

k++;

}

}

void Merge\_sort(int arr[], int low, int high)

{

if (low < high)

{

int mid = (low + high )/ 2;

Merge\_sort(arr, low, mid);

Merge\_sort(arr, mid + 1, low);

Merge(arr, low, mid, high);

}

}

int main()

{

int s,i,x,f;

cout << "\nEnter the size of the Array: ";

cin >> s;

int A[s];

cout << "\nEnter the values of the array" << endl;

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

cin >> A[i];

cout << "\nEnter the value of X to search: " << endl;

cin >> x;

Merge\_sort(A, 0, s-1);

f=binarySearch(A, 0, s, x);

if(f==-1)

cout<<"\nNot found";

else

cout<<"\nElement found";

}

**QUICK SORT**

#include<iostream>

#include<cstdlib>

using namespace std;

void swap(int \*a, int \*b) {

int temp;

temp = \*a;

\*a = \*b;

\*b = temp;

}

int Partition(int a[], int l, int h) {

int pivot, index, i;

index = l;

pivot = h;

for(i = l; i < h; i++) {

if(a[i] < a[pivot]) {

swap(&a[i], &a[index]);

index++;

}

}

swap(&a[pivot], &a[index]);

return index;

}

int RandomPivotPartition(int a[], int l, int h) {

int pvt, n, temp;

n = rand();

pvt = l + n%(h-l+1);

swap(&a[h], &a[pvt]);

return Partition(a, l, h);

}

int QuickSort(int a[], int l, int h) {

int pindex;

if(l < h) {

pindex = RandomPivotPartition(a, l, h);

QuickSort(a, l, pindex-1);

QuickSort(a, pindex+1, h);

}

return 0;

}

int main() {

int n, i;

cout<<"\nEnter the number of data element to be sorted: ";

cin>>n;

int arr[n];

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

cout<<"\nEnter element "<<i+1<<": ";

cin>>arr[i];

}

QuickSort(arr, 0, n-1);

cout<<"\nSorted Data : ";

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

{

if(i==n-1)

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

else

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

}

return 0;

}