# **Binary search code**

import java.io.\*;

import java.util.\*;

import java.util.Scanner;

public class BinarySearch{

public static int binarySearch(int *arr*[], int *x*){

int s =0, e = *arr*.length-1;

while(s<=e){

int mid = s + (e-s)/2;

if(*arr*[mid] == *x*){

return mid;

}

if(*arr*[mid]<*x*){

s = mid +1;

}

else{

e = mid-1;

}

}

return -1;

}

public static void main(String *args*[]){

//BinarySearch ob = new BinarySearch();

//Taking array as an input

int n;

int x;

Scanner sc=new Scanner(System.in);

System.out.print("Enter Array Size:");

n=sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter Array elements: ");

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

{

arr[i]=sc.nextInt();

}

System.out.print("Enter the target:");

x = sc.nextInt();

//calling the functions

int result = binarySearch(arr,x);

if(result == -1){

System.out.println("element not present");

}

else{

System.out.println("element present at index :" + result);

}

}

}

# 

# 

# 

# 

# 

# **Bubble sort**

import java.util.\*;

import java.io.\*;

import java.util.Scanner;

public class BubbleSort{

public static void bubbleSort(int *arr*[]){

int n = *arr*.length;

for(int i =1; i< n; i++){

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

if(*arr*[j]> *arr*[j+1]){

int temp = *arr*[j];

*arr*[j] = *arr*[j+1];

*arr*[j+1] = temp;

}

}

}

}

public static void PrintArray(int *arr*[]){

int n = *arr*.length;

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

System.out.print(*arr*[i]+ " ");

}

}

public static void main(String *args*[]){

// BubbleSort ob = new BubbleSort();

int n;

Scanner sc=new Scanner(System.in);

System.out.print("Enter the number of elements you want to store: ");

n=sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array: \n");

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

{

arr[i]=sc.nextInt();

}

bubbleSort(arr);

System.out.println("Sorted array is: ");

PrintArray(arr);

}

}

# **Merge Sort**

import java.util.Scanner;

import java.util.\*;

import java.io.\*;

public class MergeSort {

public static void merge(int[] *arr* , int *s* , int *e*){

int mid = (*s* + *e*)/2;

int len1 = mid - *s* + 1; //length of first half

int len2 = *e* - mid; //length of second half

int[] arr1 = new int[len1];

int[] arr2 = new int[len2];

//copy elements in these arrays;

int originalArrayIndex = *s*;

for(int i = 0; i < len1; i++){

arr1[i] = *arr*[originalArrayIndex++];

}

//originalArrayIndex = mid + 1;

for(int i = 0; i < len2; i++){

arr2[i] = *arr*[originalArrayIndex++];

}

//merge two sorted arrays;

originalArrayIndex = *s*;

int idx1 = 0;

int idx2 = 0;

while(idx1 < len1 && idx2 < len2){

if(arr1[idx1] < arr2[idx2]){

*arr*[originalArrayIndex++] = arr1[idx1++];

}

else {

*arr*[originalArrayIndex++] = arr2[idx2++];

}

}

//if in arr1 elements are remaining

while(idx1 < len1){

*arr*[originalArrayIndex++] = arr1[idx1++];

}

//if in arr2 elements are remaining

while(idx2 < len2){

*arr*[originalArrayIndex++] = arr2[idx2++];

}

}

public static void mergeSort(int[] *arr* , int *s* , int *e*){

//base case

if(*s* >= *e*)

return;

int mid = (*s* + *e*)/2;

//left Part

mergeSort(*arr* , *s* , mid);

//right part

mergeSort(*arr* , mid + 1 , *e*);

merge(*arr* , *s* , *e*);

}

public static void main(String[] *args*) {

int n;

Scanner sc=new Scanner(System.in);

System.out.print("Enter the number of elements you want to store: ");

n=sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array: \n");

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

{

arr[i]=sc.nextInt();

}

mergeSort(arr , 0 , n - 1);

System.out.println("sorted array: ");

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

System.out.print(arr[i] + " ");

}

}

}

# 

# 

# 

# 

# 

# 

# **Quick Sort**

import java.util.Scanner;

public class QuickSort {

public static void quickSort(int[] *arr* , int *s* , int *e*){

//base case

if(*s* >= *e*)

return;

//take the partition

int p = partition(*arr* , *s* , *e*);

//left part sort

quickSort(*arr* , *s* , p - 1);

//right part sort

quickSort(*arr* , p + 1 , *e*);

}

public static int partition(int[] *arr* , int *s* , int *e*){

int pivot = *arr*[*s*];

int count = 0;

//count of elements smaller than pivot element

for(int i = *s* + 1; i <= *e*; i++){

if(*arr*[i] <= pivot)

count++;

}

int pivotIdx = *s* + count;

//put pivot element at correct position

int temp = pivot;

*arr*[*s*] = *arr*[pivotIdx];

*arr*[pivotIdx] = temp;

int sIdx = *s*;

int eIdx = *e*;

//make smaller elements lie before pivot & larger elements after pivot

while(sIdx < pivotIdx && eIdx > pivotIdx){

//move sIdx++ till we find element greater than pivot element in left side

while(*arr*[sIdx] <= pivot){

sIdx++;

}

//move eIdx-- till we find smaller elements than pivot in right side

while(*arr*[eIdx] > pivot){

eIdx--;

}

//swap the values if found

if(sIdx < pivotIdx && eIdx > pivotIdx){

temp = *arr*[sIdx];

*arr*[sIdx] = *arr*[eIdx];

*arr*[eIdx] = temp;

sIdx++;

eIdx--;

}

}

return pivotIdx;

}

public static void main(String[] *args*) {

int n;

Scanner sc=new Scanner(System.in);

System.out.print("Enter the number of elements you want to store: ");

n=sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array: \n");

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

{

arr[i]=sc.nextInt();

}

quickSort(arr , 0 , n - 1);

System.out.println("Sorted array :");

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

System.out.print(arr[i] + " ");

}

}

}