**ASSIGNMENT**

P Siddharth Roy

AP19110010527

CSE-H

#Write a program for the Insertion sort algorithm.

Program:

#include<stdio.h>

int main(){

/\* Here i & j for loop counters, temp for swapping,

\* count for total number of elements, number[] to

\* store the input numbers in array. You can increase

\* or decrease the size of number array as per requirement

\*/

int i, j, count, temp, number[25];

printf("How many numbers u are going to enter?: ");

scanf("%d",&count);

printf("Enter %d elements: ", count);

// This loop would store the input numbers in array

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

scanf("%d",&number[i]);

// Implementation of insertion sort algorithm

for(i=1;i<count;i++){

temp=number[i];

j=i-1;

while((temp<number[j])&&(j>=0)){

number[j+1]=number[j];

j=j-1;

}

number[j+1]=temp;

}

printf("Order of Sorted elements: ");

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

printf(" %d",number[i]);

return 0;

}

#Write a program for the Selection sort algorithm.

Program:

#include<stdio.h>

int main(){

/\* Here i & j for loop counters, temp for swapping,

\* count for total number of elements, number[] to

\* store the input numbers in array. You can increase

\* or decrease the size of number array as per requirement

\*/

int i, j, count, temp, number[25];

printf("How many numbers u are going to enter?: ");

scanf("%d",&count);

printf("Enter %d elements: ", count);

// Loop to get the elements stored in array

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

scanf("%d",&number[i]);

// Logic of selection sort algorithm

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

for(j=i+1;j<count;j++){

if(number[i]>number[j]){

temp=number[i];

number[i]=number[j];

number[j]=temp;

}

}

}

printf("Sorted elements: ");

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

printf(" %d",number[i]);

return 0;

}

#Write a program for Bubble sort algorithm.

Program

/\* Implementing Bubble sort in a C Program

\*/

#include<stdio.h>

int main(){

int count, temp, i, j, number[30];

printf("How many numbers are u going to enter?: ");

scanf("%d",&count);

printf("Enter %d numbers: ",count);

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

scanf("%d",&number[i]);

/\* This is the main logic of bubble sort algorithm

\*/

for(i=count-2;i>=0;i--){

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

if(number[j]>number[j+1]){

temp=number[j];

number[j]=number[j+1];

number[j+1]=temp;

}

}

}

printf("Sorted elements: ");

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

printf(" %d",number[i]);

return 0;

}

#Write a program for the Merge sort algorithm.

Program:

#include <stdio.h>

#define max 10

int a[11] = { 10, 14, 19, 26, 27, 31, 33, 35, 42, 44, 0 };

int b[10];

void merging(int low, int mid, int high) {

int l1, l2, i;

for(l1 = low, l2 = mid + 1, i = low; l1 <= mid && l2 <= high; i++) {

if(a[l1] <= a[l2])

b[i] = a[l1++];

else

b[i] = a[l2++];

}

while(l1 <= mid)

b[i++] = a[l1++];

while(l2 <= high)

b[i++] = a[l2++];

for(i = low; i <= high; i++)

a[i] = b[i];

}

void sort(int low, int high) {

int mid;

if(low < high) {

mid = (low + high) / 2;

sort(low, mid);

sort(mid+1, high);

merging(low, mid, high);

} else {

return;

}

}

int main() {

int i;

printf("List before sorting\n");

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

printf("%d ", a[i]);

sort(0, max);

printf("\nList after sorting\n");

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

printf("%d ", a[i]);

}

#Write a program for the Heap sort algorithm.

Program:

// C# program for implementation of Heap Sort

using System;

public class HeapSort

{

public void sort(int[] arr)

{

int n = arr.Length;

// Build heap (rearrange array)

for (int i = n / 2 - 1; i >= 0; i--)

heapify(arr, n, i);

// One by one extract an element from heap

for (int i=n-1; i>0; i--)

{

// Move current root to end

int temp = arr[0];

arr[0] = arr[i];

arr[i] = temp;

// call max heapify on the reduced heap

heapify(arr, i, 0);

}

}

// To heapify a subtree rooted with node i which is

// an index in arr[]. n is size of heap

void heapify(int[] arr, int n, int i)

{

int largest = i; // Initialize largest as root

int l = 2\*i + 1; // left = 2\*i + 1

int r = 2\*i + 2; // right = 2\*i + 2

// If left child is larger than root

if (l < n && arr[l] > arr[largest])

largest = l;

// If right child is larger than largest so far

if (r < n && arr[r] > arr[largest])

largest = r;

// If largest is not root

if (largest != i)

{

int swap = arr[i];

arr[i] = arr[largest];

arr[largest] = swap;

// Recursively heapify the affected sub-tree

heapify(arr, n, largest);

}

}

/\* A utility function to print array of size n \*/

static void printArray(int[] arr)

{

int n = arr.Length;

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

Console.Write(arr[i]+" ");

Console.Read();

}

// Driver program

public static void Main()

{

int[] arr = {12, 11, 13, 5, 6, 7};

int n = arr.Length;

HeapSort ob = new HeapSort();

ob.sort(arr);

Console.WriteLine("Sorted array is");

printArray(arr);

}

}