Name: Sumit Kalshetty

Sub: DSA(Lab). Roll no: 2125

Div: A

PRACTICAL NO:9

Problem Statement:

Implement Heap sort to sort given set of values using max or min heap.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PROGRAM\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <iostream>

using namespace std;

// A function to heapify the array.

void MaxHeapify(int a[], int i, int n)

{

int j, temp;

temp = a[i];

j = 2\*i;

while (j <= n)

{

if (j < n && a[j+1] > a[j])

j = j+1;

// Break if parent value is already greater than child value.

if (temp > a[j])

break;

// Switching value with the parent node if temp < a[j].

else if (temp <= a[j])

{

a[j/2] = a[j];

j = 2\*j;

}

}

a[j/2] = temp;

return;

}

void Build\_MaxHeap(int a[], int n)

{

int i;

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

MaxHeapify(a, i, n);

}

void Max\_HeapSort(int a[], int n)

{

int i, temp;

for (i = n; i >= 2; i--)

{

// Storing maximum value at the end.

temp = a[i];

a[i] = a[1];

a[1] = temp;

// Building max heap of remaining element.

MaxHeapify(a, 1, i - 1);

}

}

void min\_heapify(int a[],int i,int n)

{

int j, temp;

temp = a[i];

j = 2 \* i;

while (j <= n)

{

if (j < n && a[j+1] < a[j])

j = j + 1;

if (temp < a[j])

break;

else if (temp >= a[j])

{

a[j/2] = a[j];

j = 2 \* j;

}

}

a[j/2] = temp;

return;

}

void build\_minheap(int a[], int n)

{

int i;

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

{

min\_heapify(a,i,n);

}

}

void Min\_HeapSort(int a[], int n)

{

int i, temp;

for (i = n; i >= 2; i--)

{

// Storing minimum value at the end.

temp = a[i];

a[i] = a[1];

a[1] = temp;

// Building max heap of remaining element.

min\_heapify(a, 1, i - 1);

}

}

void print(int arr[], int n)

{

cout<<"\nSorted Data ";

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

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

return;

}

int main()

{

int n, i, ch;

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

cin>>n;

int arr[n];

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

{

cout<<"Enter element "<<i<<": ";

cin>>arr[i];

}

// Building max heap.

do

{

cout<<"\n1. Heap sort using max heap";

cout<<"\n2. Heap sort using min heap";

cout<<"\n 3. exit";

cout<<"\nenter your choice:";

cin>>ch;

switch(ch)

{

case 1: Build\_MaxHeap(arr, n);

Max\_HeapSort(arr, n);

print(arr, n);

break;

case 2: build\_minheap(arr, n);

Min\_HeapSort(arr, n);

print(arr, n);

break;

case 3:return 0;

default:cout<<"\n Invalid choice !! Please enter your choice again."<<endl;

}

}

while(ch!=3);

}