**Algorithm Analysis and Data Structures**

**CS 5343.502(Spring 2020)**

**Assignment 3**

**Divya Birla**

**2021514344**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**QUESTION:**

Implement heap sort.

1. Make an array of 15 numbers. Make sure the array is not sorted AND the numbers do not make a heap.

    You may hard code this array in your main program.

2. Convert the array into a heap, using Floyd's algorithm.  Print the new array

3. Sort the array into descending order using heap sort method. Print the array

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**SOURCE CODE:**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Course: CS 5343.502 – Spring 2020

\* Assignment <3>

\* Name: Divya Birla

\* E-mail: dxb190021@utdallas.edu

\* Submitted: <03/1/20>

This program performs heapsort on an array of 15 elements.

Here we first build a min heap using Floyd's method and then perform heapsort to arrange elements in the descending order.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<iostream>

#include<cstddef>

#include<stdlib.h>

#include<time.h>

using namespace std;

//Build a min heap

void heapify(int a[], int size, int index) {

int smallest = index;

int lchild = (2 \* index) + 1;

int rchild = (2 \* index) + 2;

if ((lchild < size) && (a[lchild] < a[smallest]))

smallest = lchild;

if ((rchild < size) && (a[rchild] < a[smallest]))

smallest = rchild;

if (smallest != index) {

int temp = a[index];

a[index] = a[smallest];

a[smallest] = temp;

heapify(a, size, smallest);

}

}

//Identifying last parent for Floyd's method

void FloydBuildHeap(int a[], int size) {

int pindex = size / 2;

for (int i = pindex -1; i >= 0; --i) {

heapify(a, size, i);

}

}

// Heapsort by replacing the leaf node with root node

void heapsort(int a[], int size) {

for (int i = size - 1; i >= 0; i--) {

int temp = a[0];

a[0] = a[i];

a[i] = temp;

heapify(a, i, 0);

}

}

//Display array of elements

void disparray(int a[], int size) {

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

cout << a[i] << " ";

}

int main() {

cout << "\n--------PROGRAM TO BUILD HEAP USING FLOYDs AND THEN PERFORMING HEAPSORT-------\n";

int a[] = { 12,5,11,3,10,6,9,4,8,1,7,2,13,21,25 };

int size = sizeof(a) / sizeof(int);

// Bulding Floyd's heap

FloydBuildHeap(a, size);

cout << "\n Heap built using Floyd's Method is as follows:"<<endl;

cout << "\n ";

disparray(a,size);

cout << "\n" ;

// Performing heap sort

heapsort(a, size);

cout << "\n Array after heapsort is as follows:" << endl;

cout << "\n ";

disparray(a, size);

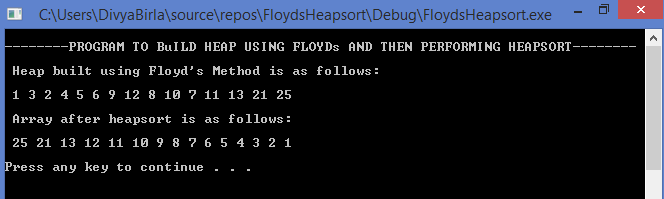
cout << "\n \n";

system("pause");

}

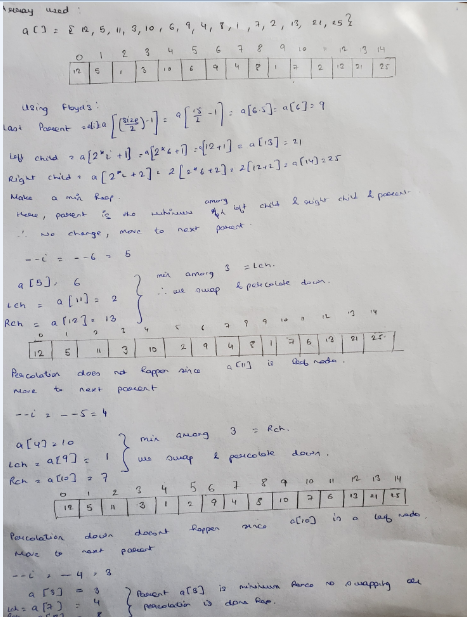
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

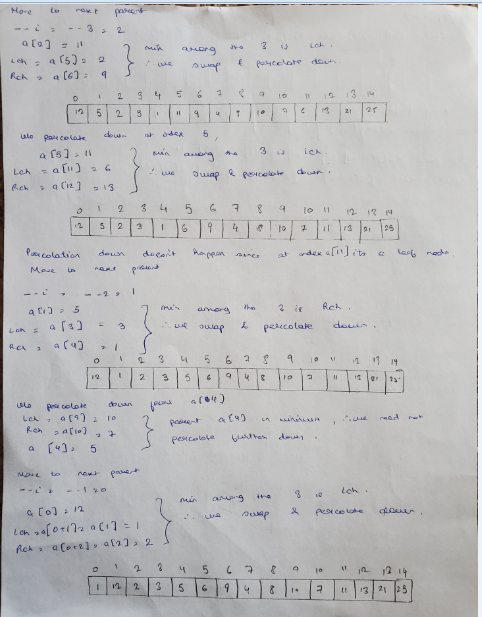
**OUTPUT:**

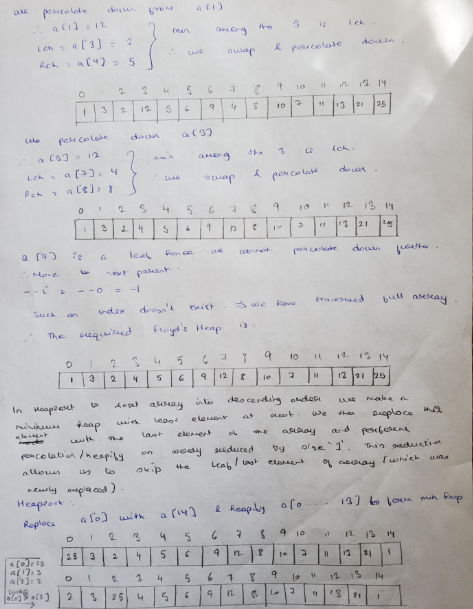


**IMAGES:**

**Min heap creation for Floyd**







**Heapsort using the min heapify.**

