#include<iostream>

Source Code

#include<cmath>

using namespace std;

void swap(int &x,int &y){

    int temp=x;

    x=y;

    y=temp;

}

class Heap{

public:

    int \*a;

    int cap;

    int curr\_size;

    Heap(int s){

        curr\_size=0;

        cap=s;

        a=**new** int[s];

    }

    void PrintArray()

    {

        for(int i=0;i<curr\_size;i++){

            cout<<a[i]<<" ";

        }

        cout<<endl;

    }

    int parent(int i){

        // return floor(i/2);

        return ((i-1)/2);

    }

    int left(int i){

        return (2\*i +1);

    }

    int right(int i){

        return (2\*i +2);

    }

    void insertHeap(int ele){

        if(curr\_size==cap){

            cout<<"Overflow: Heap full"<<endl;

            return;

        }

        curr\_size++;

        int i=curr\_size-1;

        a[i]=ele;

        while(i!=0 and a[parent(i)]<a[i]){

            swap(a[i],a[parent(i)]);

            i=parent(i);

        }

    }

};

int main(){

    int size;

    cout<<"Enter Size of Max Heap"<<endl;

    cin>>size;

    Heap obj(size);

    cout<<"Max heap created"<<endl;

    cout<<"Now Enter the element: "<<endl;

    int x;

    for(int i=0;i<size;i++){

        cin>>x;

        obj.insertHeap(x);

    }

    cout<<"After Insertion: "<<endl;

obj.PrintArray();

return 0;

//12 23 45 67 89 5 8 10 2 15 20

}

Output

#include<iostream>

Source Code

#include<cmath>

using namespace std;

void swap(int &x,int &y){

    int temp=x;

    x=y;

    y=temp;

}

class Heap{

public:

    int \*a;

    int cap;

    int curr\_size;

    Heap(int s){

        curr\_size=0;

        cap=s;

        a=**new** int[s];

    }

    void PrintArray()

    {

        for(int i=0;i<curr\_size;i++){

            cout<<a[i]<<" ";

        }

        cout<<endl;

    }

    int parent(int i){

        // return floor(i/2);

        return ((i-1)/2);

    }

    int left(int i){

        return (2\*i +1);

    }

    int right(int i){

        return (2\*i +2);

    }

    void insertHeap(int ele){

        if(curr\_size==cap){

            cout<<"Overflow: Heap full"<<endl;

            return;

        }

        curr\_size++;

        int i=curr\_size-1;

        a[i]=ele;

        while(i!=0 and a[parent(i)]<a[i]){

            swap(a[i],a[parent(i)]);

            i=parent(i);

        }

    }

    void Heapify(int i){

        int l=left(i);

        int r=right(i);

        int largest=i;

        if(l<curr\_size and a[l]>a[i]){

            largest=l;

        }

        if(r<curr\_size and a[r]>a[l]){

            largest=r;

        }

        if(largest!=i){

            swap(a[largest],a[i]);

            Heapify(largest);

        }

    }

    int DeleteMax(){

        if(curr\_size<=0){

            return -1;

        }

        if(curr\_size==1){

            curr\_size--;

            return a[0];

        }

        int root=a[0];

        a[0]=a[curr\_size-1];

        curr\_size--;

        Heapify(0);

        return root;

    }

};

int main(){

    int size;

    cout<<"Enter Size of Max Heap"<<endl;

    cin>>size;

    Heap obj(size);

    cout<<"Max heap created"<<endl;

    cout<<"Now Enter the element: "<<endl;

    int x;

    for(int i=0;i<size;i++){

        cin>>x;

        obj.insertHeap(x);

    }

    cout<<"After Insertion: "<<endl;

    obj.PrintArray();

    // cout<<"Deleted element is: "<<obj.DeleteMin()<<endl;

    // obj.PrintArray();

    cout<<"Deleted(Descending order) element is: "<<endl;

    for(int i=0;i<size;i++){

        cout<<obj.DeleteMax()<<" ";

    }

    cout<<endl;

    return 0;

//12 23 45 67 89 5 8 10 2 15 20

}

Output