public class BinHeap {

private static final int d= 2;

private int[] heap;

private int heapSize;

/\*\*

\* This will initialize our heap with default size.

\*/

public BinHeap(int capacity){

heapSize = 0;

heap = new int[ capacity+1];

Arrays.fill(heap, -1);

}

/\*\*

\* This will check if the heap is empty or not

\* Complexity: O(1)

\*/

public boolean isEmpty(){

return heapSize==0;

}

public boolean isFull(){

return heapSize == heap.length;

}

private int parent(int i){

return (i-1)/d;

}

private int kthChild(int i,int k){

return d\*i +k;

}

public void insert(int x){

if(isFull())

throw new NoSuchElementException("Heap is full, No space to insert new element");

heap[heapSize++] = x;

heapifyUp(heapSize-1);

}

public int delete(int x){

if(isEmpty())

throw new NoSuchElementException("Heap is empty, No element to delete");

int key = heap[x];

heap[x] = heap[heapSize -1];

heapSize--;

heapifyDown(x);

return key;

}

private void heapifyUp(int i) {

int temp = heap[i];

while(i>0 && temp > heap[parent(i)]){

heap[i] = heap[parent(i)];

i = parent(i);

}

heap[i] = temp;

}

private void heapifyDown(int i){

int child;

int temp = heap[i];

while(kthChild(i, 1) < heapSize){

child = maxChild(i);

if(temp < heap[child]){ heap[i] = heap[child]; }else break; i = child; } heap[i] = temp; } private int maxChild(int i) { int leftChild = kthChild(i, 1); int rightChild = kthChild(i, 2); return heap[leftChild]>heap[rightChild]?leftChild:rightChild;

}

public void printHeap()

{

System.out.print("nHeap = ");

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

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

System.out.println();

}

public int findMax(){

if(isEmpty())

throw new NoSuchElementException("Heap is empty.");

return heap[0];

}

public static void main(String[] args){

BinHeap maxHeap = new BinHeap(10);

maxHeap.insert(10);

maxHeap.insert(4);

maxHeap.insert(9);

maxHeap.insert(1);

maxHeap.insert(7);

maxHeap.insert(5);

maxHeap.insert(3);

maxHeap.printHeap();

maxHeap.delete(5);

maxHeap.printHeap();

}

}