**package** mbatch;

**class** MaxIntHeap

{

**int** arr[]=**new** **int**[10];

**int** size=0;

**int** getParentIndex(**int** i)

{

**return** (i-1)/2;

}

**int** getLeftChildIndex(**int** i)

{

**return** (2\*i+1);

}

**int** getRightChildIndex(**int** i)

{

**return** (2\*i+2);

}

**boolean** hasParent(**int** i)

{

// if(getParentIndex(i)<0) return false;

// return true;

**return** getParentIndex(i)>=0;

}

**boolean** hasLeftChild(**int** i)

{

**return** getLeftChildIndex(i)<size;

}

**boolean** hasRightChild(**int** i)

{

**return** getRightChildIndex(i)<size;

}

**int** parent(**int** i)

{

**return** arr[getParentIndex(i)];

}

**int** rightChild(**int** i)

{

**return** arr[getRightChildIndex(i)];

}

**int** leftChild(**int** i)

{

**return** arr[getLeftChildIndex(i)];

}

**int** size()

{

**return** size;

}

**boolean** isEmpty()

{

**return** size<=0; //or size==0

}

**int** peek()

{

**return** arr[0];

}

**void** insert(**int** val)

{

arr[size]=val;

size++;

HeapifyUp();

}

**void** HeapifyUp()

{

**int** i=size-1;

**while**(hasParent(i) && parent(i)<arr[i])

{

swap(i,getParentIndex(i));

i=getParentIndex(i);

}

}

**void** swap(**int** a,**int** b)

{

**int** temp=arr[a];

arr[a]=arr[b];

arr[b]=temp;

}

**void** print()

{

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

{

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

}

System.***out***.println();

}

**int** poll()

{

**int** val=arr[0];

arr[0]=arr[size-1];

size--;

HeapifyDown();

**return** val;

}

**void** HeapifyDown()

{

**int** i=0;

**while**(hasLeftChild(i))

{

**int** greaterChildIndex=getLeftChildIndex(i);

**if**(hasRightChild(i) && rightChild(i)>leftChild(i))

{

greaterChildIndex=getRightChildIndex(i);

}

**if**(arr[i]<arr[greaterChildIndex])

{

swap(i,greaterChildIndex);

}

**else**

{

**break**;

}

i=greaterChildIndex;

}

}

}

**public** **class** MaxxHeap {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

MaxIntHeap mh=**new** MaxIntHeap();

mh.insert(10);

mh.insert(5);

mh.insert(3);

mh.insert(2);

mh.insert(7);

mh.print();

mh.poll();

mh.print();

}

}