学习情况表

|  |  |  |  |
| --- | --- | --- | --- |
| **姓名** | 李煦 | **学号** | 2021902007 |
| **学院** | 信息工程学院 | **专业** | 电子信息类 |

（请在下面表格本周学习情况）

|  |
| --- |
| 学习情况简述 |
| 1. **学习内容** 2. **学习了优先队列** 3. **学习了堆的思想** 4. **学习了堆排序的代码表示** 5. **学习了最小堆和最大堆的实现** 6. **学习收获（思考）** 7. **了解了堆排序的思想方法** 8. **掌握了最小堆和最大堆的表示** 9. **了解了优先队列** |
| 本周练习过的代码（例） |
| （本周练习写过的代码，如实填写，疑似抄袭会单独测试）  *代码1：*  *#include<stdio.h>*  *int main()*  *{*  *printf("HelloWorld\n");*  *return 0;*  *}*  *代码2：*  …….. |
| 本周练习过的代码 |
| public class MaxPriorityQueue<T extends Comparable<T>> {     private int N;     private T[] items;         public MaxPriorityQueue(int capacity){         this.N=0;         this.items = (T[]) new Comparable[capacity + 1];     }     private boolean less(int i,int j){         return items[i].compareTo(items[j])<0;     }     private void exch(int i,int j){         T tmp = items[i];         items[i]=items[j];         items[j]=tmp;     }     public T delMax(){         if(N==0){             return null;         }         T max=  items[1];         exch(1,N);         N--;         sink(1);         return max;     }      public void insert(T t){             items[++N]=t;             swim(N);     }     private void swim(int k){         while(k>1){             if(less(k/2,k)){                 exch(k/2,k);             }             k=k/2;         }     }     private void sink(int k){         while (2\*k<=N){             int max;             if(2\*k+1<=N){                 if (less(2\*k,2\*k+1)){                     max=2\*k+1;                 }else{                     max=2\*k;                 }             }else{                 max=2\*k;             }             if(!less(k,max)){//k>=max,                 break;             }             exch(k,max);             k=max;         }     }     public int size(){         return N;     }     public boolean isEmpty(){         return N==0;     } }  package MaxHeap;  import java.util.Scanner;  public class Heap {     private int data[];     private int size;     private int capacity;     private final int maxNum = 999999999;      public Heap()     {         data = new int[10+1];         size = 0;         capacity= 10;         data[0] = maxNum;     }      public Heap(int n)     {         data = new int[n+1];         size = 0;         capacity = n;         data[0] = maxNum;     }      public boolean isFull()     {         return size==capacity;     }      public boolean isEmpty()     {         return size==0;     }      public boolean insertHeap(int e)     {         if (isFull())         {             System.out.println("The heap is full");             return false;         }          int i = ++size;         for (;data[i/2] < e;i = i/2 )         {             data[i] = data[i/2];         }         data[i] = e;         return true;     }      public int deleteMax()     {         if (isEmpty())         {             System.out.println("The heap is empty");             return -1;         }         int maxData = data[1];         int x = data[size--];         int parent,child;         for (parent = 1;parent \* 2<=size;parent = child)         {             child = parent\*2;             if (child!=size && data[child] < data[child+1])             {                 child++;             }             if (x >= data[child]) break;             else data[parent] = data[child];         }          data[parent] = x;          return maxData;     }      public void percDown(int p)     {         int parent,child;         int x = data[p];         for (parent = p;parent\*2<=size;parent = child)         {             child = 2\*parent;             if (child !=size && data[child] <data[child+1])             {                 child++;             }             if (x >= data[child]) break;             else data[parent] = data[child];         }         data[parent] = x;      }      public void buildHeap(int n)     {         Scanner sc = new Scanner;         for (int i = 1;i<=n;i++)         {             data[i] = sc.nextInt();             size++;         }         for (int i = size/2;i>0;i--)         {             percDown(i);         }     }  import java.util.Scanner;  public class Heap {     private int data[];     private int size;     private int capacity;     private final int minNum = -999999999;      public Heap()     {         data = new int[10+1];         size = 0;         capacity = 10;         data[0] = minNum;     }       public Heap(int n)     {         data = new int[n+1];         size = 0;         capacity = n;         data[0] = minNum;     }      public boolean isFull()     {         return size==capacity;     }       public boolean isEmpty()     {         return size==0;     }      public boolean insertHeap(int e)     {         if (isFull())         {             System.out.println("The heap is full");             return false;         }          int i = ++size;         for (;data[i/2] > e;i = i/2)         {             data[i] = data[i/2];         }         data[i] = e;         return true;     }      public int deleteMin()     {         if (isEmpty())         {             System.out.println("The heap is empty");             return -1;         }         int parent,child;         int minData = data[1];         int x = data[size--];         for (parent = 1;parent\*2 <= size;parent = child)         {             child = parent\*2;             if (child !=size && data[child] > data[child+1])             {                 child++;             }             if (x <= data[child]) break;             else data[parent] = data[child];         }         data[parent] = x;         return minData;     }       public void percDown(int p)     {         int parent,child;         int x = data[p];         for (parent = p;parent\*2<=size;parent = child)         {             child = 2\*parent;             if (child !=size && data[child] > data[child+1])             {                 child++;             }             if (x <= data[child]) break;             else data[parent] = data[child];         }         data[parent] = x;     }      public void bulidHeap(int n)     {         Scanner sc = new Scanner;         for (int i = 1;i<=n;i++)         {             data[i] = sc.nextInt();             size++;         }          for (int i = size/2;i>0;i--)         {             percDown(i);         }     }  } |

1. 该表信息将会作为你报名申请的重要依据，请认真仔细填写。
2. 培训班有严格的制度，请认真阅读规则并结合自身情况填写该表。
3. 完成该表填写后以“xxx学习情况”命名，及时上传作业。

最后希望大家能够加入我们，一起努力，共同进步！