题目描述：

AI 识别到面板上有 N（1<= N <= 100）个指示灯，灯大小一样，任意两个灯之间无重叠。 由于 AI 识别误差，每次识别到的指示灯位置可能有差异，以 4 个坐标值描述 AI 识别的指示灯的大小和位置(左上角 x1,y1，右下角 x2,y2)，

请输出先行后列排序的指示灯的编号，排序规则：

1、每次在尚未排序的灯中挑选最高的灯作为的基准灯，

2、找出和基准灯属于同一行所有的灯进行排序。两个灯高低偏差不超过灯半径算同一行（即两个灯 y 坐标的差 <= 灯高度的一半）。

输入描述：

第一行为 N，表示灯的个数

接下来 N 行，每行为 1 个灯的坐标信息，格式为：编号 x1 y1 x2 y2，编号全局唯一，

1<= 编号 <= 100，0 <= x1 < x2 <= 1000，0 <= y1 < y2 <= 1000

输出描述：

排序后的编号列表，编号之间以空格分隔

示例 1

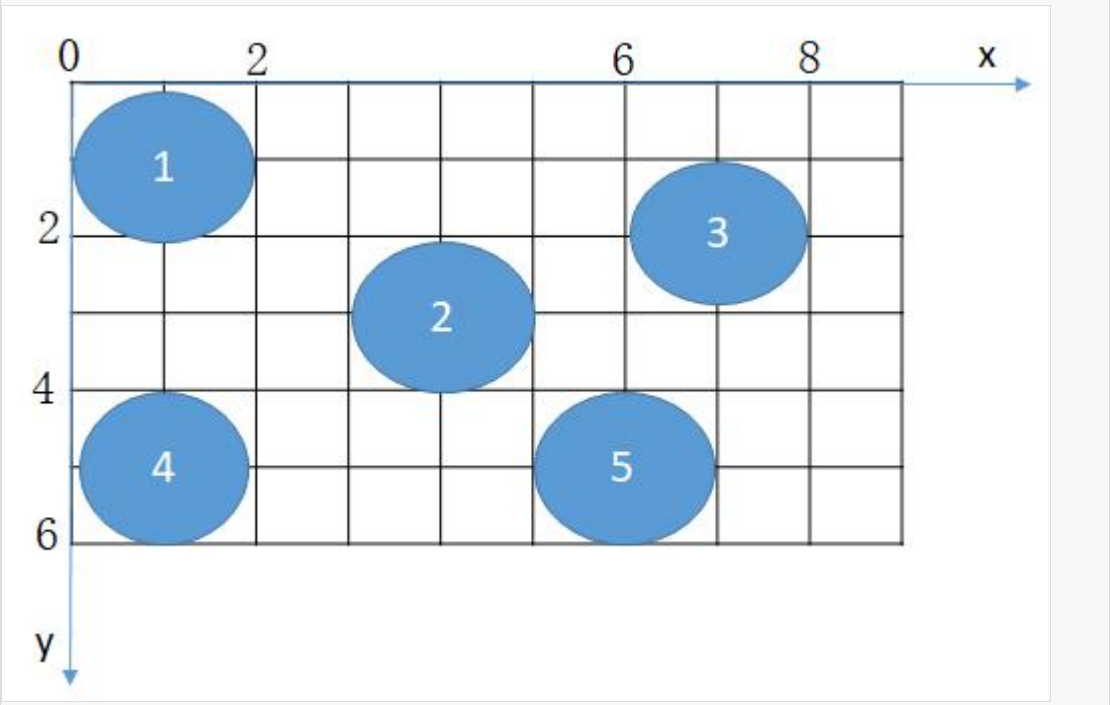
输入：

```
5
1 0 0 2 2
2 6 1 8 3
3 3 2 5 4
5 5 4 7 6
4 0 4 2 6
```

输出：

```
1 2 3 4 5
```

说明：




```
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        PriorityQueue<int[]> q = new PriorityQueue<>(new Comparator<int[]>() {
            @Override
            public int compare(int[] o1, int[] o2) {
               return o1[2] - o2[2];
            }
        });
        int height = 0;
        for (int i = 0; i < n; i++) {
            int no = in.nextInt();
            int x1 = in.nextInt();
            int y1 = in.nextInt();
            int x2 = in.nextInt();
            int y2 = in.nextInt();
            height = y2 - y1;
            q.add(new int[]{no, x1, y1, x2, y2});
        }
        List<String> res = new LinkedList<>();
        while(!q.isEmpty()){
            int[] base = q.peek();
            List<int[]> lineNodes = new ArrayList<>();
            while(!q.isEmpty() && q.peek()[2] - base[2] <= height / 2){
```

```
                    lineNodes.add(q.poll());
                }
                lineNodes.sort(new Comparator<int[]>() {
                    @Override
                    public int compare(int[] o1, int[] o2) {
                        return o1[1] - o2[1];
                    }
                });
                for (int i = 0; i < lineNodes.size(); i++) {
                    res.add(String.valueOf(lineNodes.get(i)[0]));
                }
            }

            System.out.println(String.join(" ", res));
        }


}
```