

题目描述

AI识别到面板上有N ($1 \leq N \leq 100$) 个指示灯，灯大小一样，任意两个之间无重叠。

由于AI识别误差，每次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)，

请输出先后行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。两个灯高低偏差超过灯半径算同一行（即两个灯坐标的差 \leq 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 2 y2
```

- 编号全局唯一
- $1 \leq \text{编号} \leq 100$
- $0 \leq x1 < x2 \leq 1000$
- $0 \leq y1 < y2 \leq 1000$

输出描述

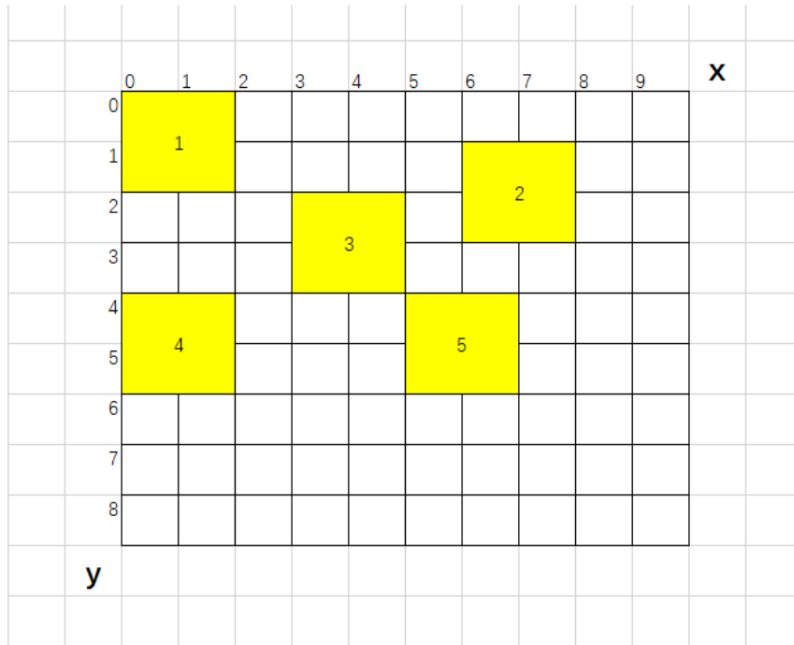
排序后的编号列表，编号之间以空格分隔

用例

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	

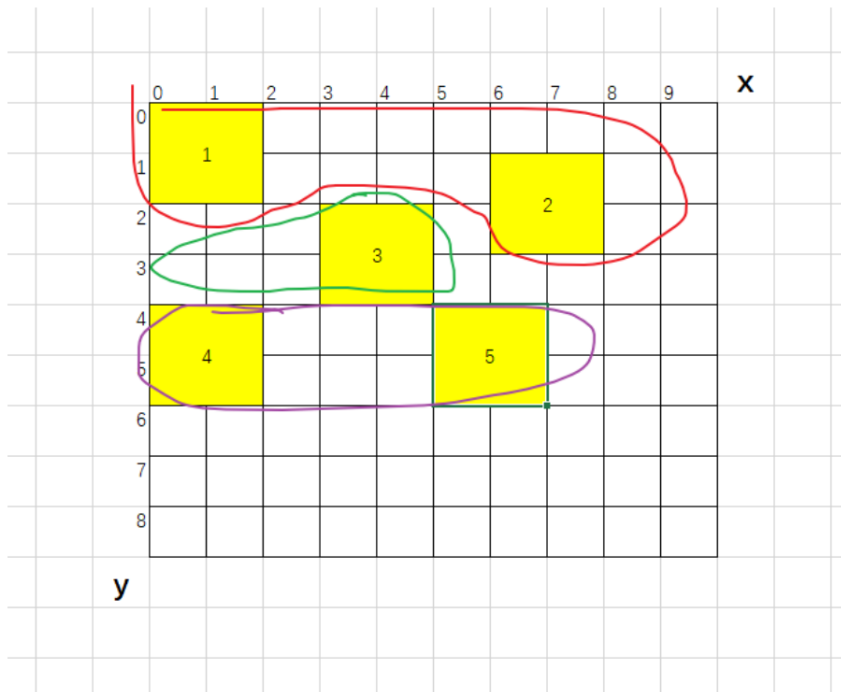
题目解析

题目给的图好像有问题，我画出来是



按照题目意思，每次取出一个最高的灯（y坐标最小的），然后找出和最高灯坐标相差小于等于灯半径的，作为同一行，然后按照x轴坐标进行排序

即存在如下用圈框住的三行，因此打印顺序是：1, 2, 3, 4, 5



```
1 import java.util.ArrayList;
2 import java.util.Arrays;
3 import java.util.Scanner;
4 import java.util.StringJoiner;
5
6 public class Main {
```

```
7 public static void main(String[] args) {
8     Scanner sc = new Scanner(System.in);
9
10    int n = sc.nextInt();
11
12    Light[] lights = new Light[n];
13    for (int i = 0; i < n; i++) {
14        int id = sc.nextInt();
15        int x1 = sc.nextInt();
16        int y1 = sc.nextInt();
17        int x2 = sc.nextInt();
18        int y2 = sc.nextInt();
19        lights[i] = new Light(id, (x1 + x2) / 2, (y1 + y2) / 2, (x2 - x1) / 2);
20    }
21
22    System.out.println(getResult(lights));
23 }
24
25 public static String getResult(Light[] lights) {
26     // 按照圆心y坐标升序
27     Arrays.sort(lights, (a, b) -> a.y - b.y);
28
29     // ans记录题解
30     StringJoiner ans = new StringJoiner(" ");
31
32     // sameRowLights记录同一行的灯
33     ArrayList<Light> sameRowLights = new ArrayList<>();
34     Light base = lights[0];
35     sameRowLights.add(base);
36
37     for (int i = 1; i < lights.length; i++) {
38         Light light = lights[i];
39
40         // 如果lights[i]的纵坐标和base的纵坐标相差不超过半径，则视为同一行
41         if (light.y - base.y <= base.r) {
42             sameRowLights.add(light);
43         } else {
44             // 否则，不是同一行
45             // 针对同一行的灯，再按照横坐标升序
46             sameRowLights.sort((a, b) -> a.x - b.x);
```

```
47         sameRowLights.forEach(a -> ans.add(a.id + ""));
48         sameRowLights.clear();
49
50         // 开始新的一行记录
51         base = light;
52         sameRowLights.add(base);
53     }
54 }
55
56 // 注意不要漏了最后一行
57 if (sameRowLights.size() > 0) {
58     sameRowLights.sort((a, b) -> a.x - b.x);
59     sameRowLights.forEach(a -> ans.add(a.id + ""));
60 }
61
62 return ans.toString();
63 }
64 }
65
66 class Light {
67     int id; // 编号
68     int x; // 圆心横坐标
69     int y; // 圆心纵坐标
70     int r; // 圆半径
71
72     public Light(int id, int x, int y, int r) {
73         this.id = id;
74         this.x = x;
75         this.y = y;
76         this.r = r;
77     }
78 }
```