**问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201403-2 |
| 试题名称： | 窗口 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　在某图形操作系统中,有 N 个窗口,每个窗口都是一个两边与坐标轴分别平行的矩形区域。窗口的边界上的点也属于该窗口。窗口之间有层次的区别,在多于一个窗口重叠的区域里,只会显示位于顶层的窗口里的内容。 　　当你点击屏幕上一个点的时候,你就选择了处于被点击位置的最顶层窗口,并且这个窗口就会被移到所有窗口的最顶层,而剩余的窗口的层次顺序不变。如果你点击的位置不属于任何窗口,则系统会忽略你这次点击。 　　现在我们希望你写一个程序模拟点击窗口的过程。  **输入格式**  　　输入的第一行有两个正整数,即 N 和 M。(1 ≤ N ≤ 10,1 ≤ M ≤ 10) 　　接下来 N 行按照从最下层到最顶层的顺序给出 N 个窗口的位置。 每行包含四个非负整数 x1, y1, x2, y2,表示该窗口的一对顶点坐标分别为 (x1, y1) 和 (x2, y2)。保证 x1 < x2,y1 2。 　　接下来 M 行每行包含两个非负整数 x, y,表示一次鼠标点击的坐标。 　　题目中涉及到的所有点和矩形的顶点的 x, y 坐标分别不超过 2559 和　　1439。  **输出格式**  　　输出包括 M 行,每一行表示一次鼠标点击的结果。如果该次鼠标点击选择了一个窗口,则输出这个窗口的编号(窗口按照输入中的顺序从 1 编号到 N);如果没有,则输出"IGNORED"(不含双引号)。  **样例输入**  3 4 0 0 4 4 1 1 5 5 2 2 6 6 1 1 0 0 4 4 0 5  **样例输出**  2 1 1 IGNORED  **样例说明**  　　第一次点击的位置同时属于第 1 和第 2 个窗口,但是由于第 2 个窗口在上面,它被选择并且被置于顶层。 　　第二次点击的位置只属于第 1 个窗口,因此该次点击选择了此窗口并将其置于顶层。现在的三个窗口的层次关系与初始状态恰好相反了。 　　第三次点击的位置同时属于三个窗口的范围,但是由于现在第 1 个窗口处于顶层,它被选择。 　　最后点击的 (0, 5) 不属于任何窗口。 |
| 代码程序： | #include<stdio.h>  typedef struct{  int x;  int y;  }Point;  typedef struct{  Point leftDown;  Point rightUp;  int id;  }Window;  bool isClick(Window w, Point p);  int main()  {  int N,M;  Window window[15];  scanf("%d %d",&N,&M);  // 输入窗口  for(int i = 0; i<N;i++) // 编号i+1;  {  scanf("%d %d %d %d",&window[i].leftDown.x,&window[i].leftDown.y,  &window[i].rightUp.x,&window[i].rightUp.y);  window[i].id = i+1;  }  // 处理点击  Point temp;  for(int i = 0; i<M;i++)  {  scanf("%d %d",&temp.x,&temp.y);  int j;  for(j = N-1;j>=0;j--){  if(isClick(window[j],temp))  {  printf("%d\n",window[j].id);  Window wtemp = window[j];  // 更新最顶层的窗口  for(int k = j;k<N-1;k++)  {  window[k] = window[k+1];  }  window[N-1] = wtemp;  break;  }  }  if(j<0)  {  printf("IGNORED\n");  }  }  return 0;  }  bool isClick(Window w, Point p)  {  if(p.x>=w.leftDown.x&&p.x<=w.rightUp.x && p.y>=w.leftDown.y && p.y<=w.rightUp.y)  {  return true;  }  return false;  } |