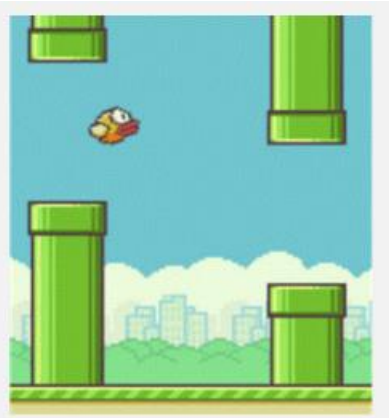
知识精炼（四）

主讲人：邓哲也

NOIP2014 Day1 飞扬的小鸟

Flappy Bird是一款风靡一时的休闲手机游戏。玩家需要不断控制点击手机屏幕的频率来调节小鸟的飞行高度，让小鸟顺利通过画面右方的管道缝隙。如果小鸟一不小心撞到了水管或者掉在地上的话，便宣告失败。



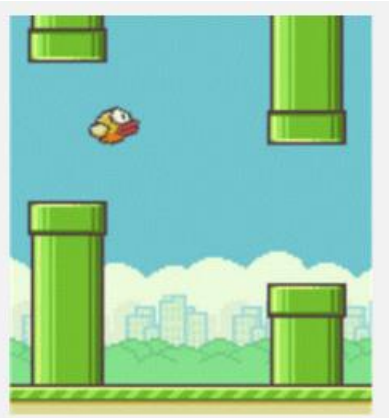
为了简化问题，我们对游戏规则进行了简化和改编：

1.游戏界面是一个长为n，高为m的二维平面，其中有k个管道（忽略管道的宽度）

2.小鸟始终在游戏界面内移动。小鸟从游戏界面最左边任意整数高度位置出发，到达游戏界面最右边时，游戏完成。

NOIP2014 Day1 飞扬的小鸟

3.小鸟每个单位时间沿横坐标方向右移的距离为1.竖直移动的距离由玩家控制。如果点击屏幕，小鸟就会上升一定高度X，每个单位可以点击多次，效果叠加：如果不点击幕，小鸟就会下降一定高度。小鸟位于横坐标方向不同位置时，上升的高度X和下降的高度y可能互不相同



4.小鸟高度等于0或者小鸟碰到管道时，游戏失败。小鸟高度为m时无法再上升。

现在，请你判断是否可以完成游戏。如果可以，输出最少点屏幕数；否则输出小鸟最多可以通过多少个管道缝隙。

NOIP2014 Day1 飞扬的小鸟

输入文件名为bird.in.

第1行有3个整数n,m,k分别表示游戏界面的长度，高度和水管的数量，每两个整数之间用一个空格隔开；

接下来的n行，每行2个用一个空格隔开的整数X和Y，依次表示在横坐标位置0~n-1上玩家点击屏幕后，小鸟在下一位置下降的高度Y。

接下来k行，每行3个整数P,L,H，每两个整数之间用一个空格隔开。每行表示一个管道，其中P表示管道的横坐标，L表示此管道缝隙的下边沿的高度为L，H表示管道缝隙上边沿的高度（输入数据保证P各不相同，但不保证按照大小顺序给出）。

NOIP2014 Day1 飞扬的小鸟

输出共两行。

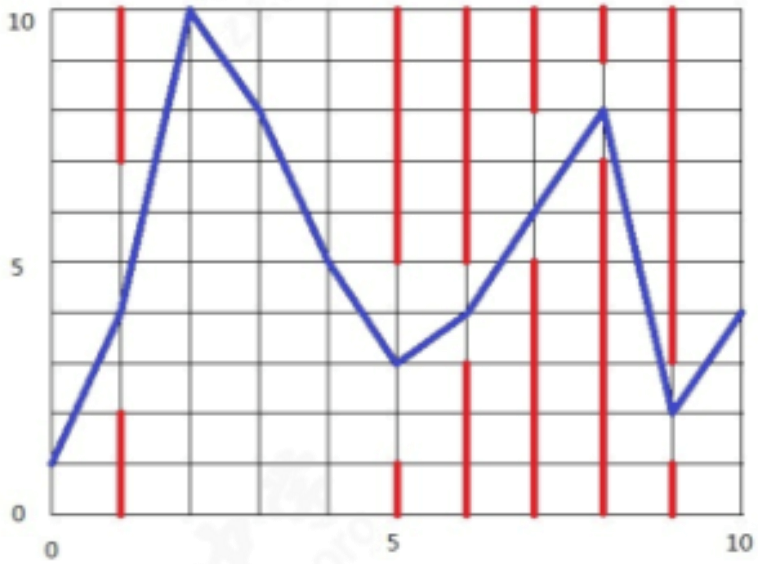
第一行，包含一个整数，如果可以成功完成游戏，则输出1，否则输出0。

第二行，包含一个整数，如果第一行为1，则输出成功完成游戏需要最少点击屏幕数，否则输出小鸟最多可以通过多少个管道缝隙。

NOIP2014 Day1 飞扬的小鸟

【输入输出样例 1】

bird.in	bird.out
10 10 6 3 9 9 9 1 2 1 3 1 2 1 1 2 1 2 1 1 6 2 2 1 2 7 5 1 5 6 3 5 7 5 8 8 7 9 9 1 3	1 6

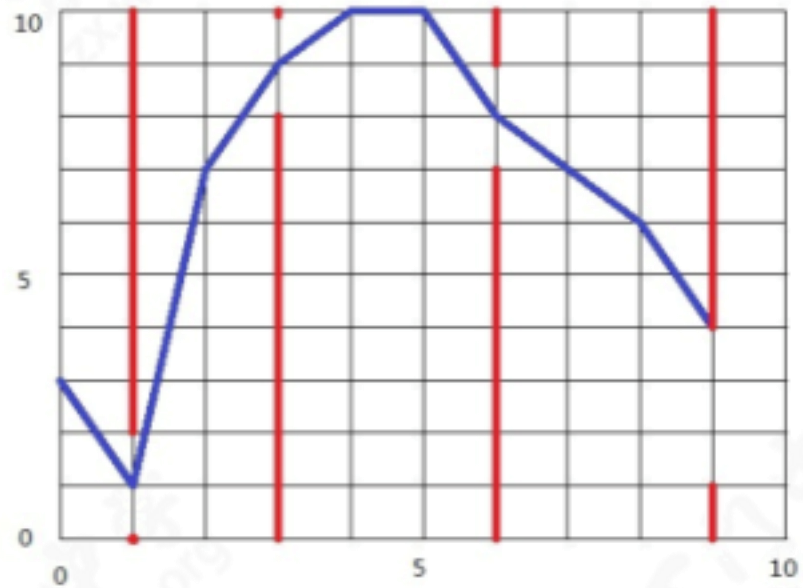


输入输出样例1说明

NOIP2014 Day1 飞扬的小鸟

【输入输出样例 2】

bird.in	bird.out
10 10 4 1 2 3 1 2 2 1 8 1 8 3 2 2 1 2 1 2 2 1 2 1 0 2 6 7 9 9 1 4 3 8 10	0 3



输入输出样例2说明

NOIP2014 Day1 飞扬的小鸟

对于 30%的数据：$5 \leqslant n \leqslant 10, 5 \leqslant m \leqslant 10, k=0$，保证存在一组最优解使得同一单位时间最多点击屏幕 3 次；

对于 50%的数据：$5 \leqslant n \leqslant 20, 5 \leqslant m \leqslant 10$，保证存在一组最优解使得同一单位时间最多点击屏幕 3 次；

对于 70%的数据：$5 \leqslant n \leqslant 1000, 5 \leqslant m \leqslant 100$；

对于 100%的数据：$5 \leqslant n \leqslant 10000, 5 \leqslant m \leqslant 1000$，

$0 \leqslant k < n, 0 < X < m, 0 < Y < m, 0 < P < n, 0 \leqslant L < H \leqslant m, L+1 < H$

NOIP2014 Day1 飞扬的小鸟

一个简单的思路，用 f[i][j] 表示到第 i 列第 j 的高度所要的最小点击次数。

枚举在第 i-1 列点 k 次屏幕，用 f[i-1][j - k * up[i-1]] + k 来更新 f[i][j]。

也就是当成多重背包来做，这样的时间复杂度是 $O(nm^2)$，只能得到 70 分。

需要使用单调队列优化，才能做到 O(nm)

NOIP2014 Day1 飞扬的小鸟

更简单的做法：

点屏幕当成完全背包来做：

```
f[i][j] = min(f[i][j], f[i-1][j - up[i-1]] + 1);
    f[i][j] = min(f[i][j], f[i][j - up[i-1]] + 1);
```

假设第 i 列没有管子，更新完所有的 f[i][j] 之后，把管子对应的部分的 f[i][j] 设置为 INF。

时间复杂度 O(nm)

NOIP2014 Day1 飞扬的小鸟

```
int up[N], down[N], l[N], r[N];
scanf(“%d%d%d”, &n, &m, &k);
for(int i = 0;i < n;i ++) scanf(“%d%d”, &up[i],
&down[i]);
for(int i = 1;i <= n;i ++) l[i] = 0, r[i] = m + 1;
for(int i = 1;i <= k;i ++){
        int x;
        scanf(“%d”, &x);
        scanf(“%d%d”, &l[x], &r[x]);
}
```

NOIP2014 Day1 飞扬的小鸟

```
f[0][0] = 1e9;
for (int i = 1; i <= n;i ++)
    for(int j = 0;j <= m;j ++)
        f[i][j] = 1e9;
```

NOIP2014 Day1 飞扬的小鸟

```
for(int i = 1;i <= n;i ++){
        for(int j = 1;j <= m;j ++){
                if (j >= up[i - 1]){
                        f[i][j] = min(f[i][j], f[i - 1][j - up[i - 1]] + 1);
        // 点一下
                        f[i][j] = min(f[i][j], f[i][j - up[i - 1]] + 1);
        // 点多下
                }
                if (j == m)
                        for(int k = j - up[i - 1];k <= m;k ++){
                                f[i][j] = min(f[i][j], f[i-1][k] + 1);
                                f[i][j] = min(f[i][j], f[i][k] + 1);
                        }
        }
        for(int j = 1;j <= m;j ++)
                if (j + down[i - 1] <= m)
                        f[i][j] = min(f[i][j], f[i - 1][j + down[i-1]]);
        …
```

NOIP2014 Day1 飞扬的小鸟

```
        ...
        for(int j = 0;j <= l[i];j ++) f[i][j] = 1e9;
        for(int j = r[i]; j <= m;j ++) f[i][j] = 1e9;
}
int ans1 = 1e9;
for(int i = 1;i <= m;i ++) ans1 = min(ans1, f[n][i]);
if (ans1 < 1e9){
        printf(“1\n%d\n”, ans1);
        return 0;
}
```

NOIP2014 Day1 飞扬的小鸟

```
int last = 1;
for(int i = 1;i <= n;i ++){
        int flag = 0;
        for(int j = 1;j <= m;j ++) if (f[i][j] < 1e9) flag = 1;
        if (!flag){
                last = i;
                break;
        }
}
```

NOIP2014 Day1 飞扬的小鸟

```
int ans2 = 0;
for(int i = 1;i < last;i ++)
    if(l[i] > 0 || r[i] <= m + 1)
        ans2 ++;
printf("0\n%d\n", ans2);
```

下节课再见