# 马术射击锦标赛

1491 年，Milan Lodovico Sforza 公爵就他和 Beatrice d'Este 之间的婚礼要求 Leonardo 来筹备婚礼相关庆祝活动，包括一个伟大的马术射击锦标赛，该比赛将进行三天。但当中最热门的骑士却来晚了……

## 锦标赛

在这场锦标赛中，N 位骑士首先排成一行，他们的位置将顺序被编号为 0 到 N-1。比赛的主持通过决定两个位置 S 和 E（$0 \le S < E \le N - 1$）来决定一轮比赛。在该轮比赛中，所有位置在 S 和 E 之间（含）的所有骑士都需要参加锦标赛。比赛中胜出的骑士可以返回他原来排队中的位置，而失败的骑士则要离场。留在赛场的骑士将会向开始线靠拢并保持原来的相互之间旳位置顺序，因此，这些骑士的位置被重新编号为 0 至 N -（E - S）- 1。然后比赛主持将再决定另一轮比赛的骑士。这样一直进行下去直至场上只剩下一位骑士。

Leonardo 知道所有的骑士实力不同，并以从 0（最弱）到 N - 1（最強）的不同等级表示，。他知道对于 C 轮比赛，主持人将会給出的所有命令，毕竟他是 Leonardo ... 他也知道在这些比赛中，每轮等级最高的骑士将会胜出。

# 迟到的骑士

N 位骑士中的 N - 1 位已经排好在开始线上，只是最热门的那位骑士还未到。这位第级为 R 的骑士来得有些迟。为了增加娱乐性，Leonardo 希望利用这位迟到的骑士的知名度，并为他选择的一个位置，使得这位迟到的骑士参赛时获胜的轮数为最大。请注意，我们对这位骑士未参加的轮数不感兴趣，只是计算他参与并获胜的轮数。

# 样例

当 N = 5 时，N - 1 位骑士已在开始线上排好队，他们的等级分别顺序为 [1, 0, 2, 4]。所以迟到的骑士的等级为 R = 3。当轮数 C = 3 时，比赛主持将会决定的 (S, E) 位置编号分别顺序为：(1, 3), (0, 1), (0, 1)。

若 Leonardo 将迟到的骑士编在第一位上，则在开始线上的骑士的等级将为 [3, 1, 0, 2, 4]。这样，第一轮比赛将包括骑士编号 1, 2, 3 ，其等级顺序为 1, 0, 2 ，所以等级为 2 的骑士将会胜出。比赛后新組成的开始线上的骑士(等级)顺序为 [3, 2, 4]。下一轮比赛等级为 3 的骑士(位置 0)面对等级为 2 的骑士(位置 1)，且等级为 R=3 的骑士将会胜出，余下的骑士(等级)顺序为[3, 4]。而最后一轮比赛中，位置 0 的骑士将对位置 1 的骑士，当然等级为 4 的那位骑士将会胜出。这样一来，迟到的骑士就只能在一轮比赛(第二轮比赛)中获胜。

但若 Leonardo 将迟到的骑士加在等级为 1 及 0 的骑士之间，开始线上的骑士等级顺序将变为[1, 3, 0, 2, 4]。这时，第一轮比赛将包括等级为 3, 0, 2 的三位骑士，而等级为 R=3 的骑士将胜出。开始线上的骑士将变为 [1, 3, 4]，而下一轮比赛中，等级为 1 的骑士将面

对等级为 3 的骑士，因而等级为 R=3 的骑士将再次胜出。而最后一场比赛中，开始线上的骑士将余下 [3, 4]，当然等级为 4 的骑士将胜出。这样一来，迟到的那位骑士将胜出两场。事实上，这是最好的安排，因为沒有任何其他可能的安排会使这位迟到的骑士胜出多过两场。

## 说明

你的任务是编写一个程序以选择一个最好的位置給这位迟到的骑士，使他如 Leonardo 所愿，胜出的轮数为最多。你需要编写一个名称为 GetBestPosition(N, C, R, K, S, E)的函数，其中:

- N 是骑士的数目;
- C 是比赛主持要负责的轮数 (1 ≤ C ≤ N - 1);
- R 是迟到的骑士的等级; 所有骑士的等级(包括那些已在起点线上及那位迟到的骑士在内)均为 0, ..., N-1 之间不相重的整数。迟到骑士的等级 R 将明确地给出，虽然它可以由其他骑士的等级推算出来。
- K 是一个含有 N - 1 个整数的数组，它们顺序表示这 N - 1 位已在开始线上排好队的骑士的等级;
- S 和 E 是两个大小为 C 的数组: 对于每个在 0 和 C-1 之间的整数 i (包含 0 及 C-1 在内)，比赛主持在第 i + 1 轮比赛中将会叫出位置在 S[i] 至 E[i] 的所有骑士参加锦标。你可以假设对所有 i 而言，S[i] < E[i]。

所有对次函数的调用都将是合法的: E[i]少于第 i+1 轮比赛中剩下的骑士数量，而 C 轮比赛之后,场上将只剩下一位骑士。

GetBestPosition(N, C, R, K, S, E)必须返回最优的位置值 P，表示 Leonardo 为那位迟

到的骑士安排的最好位置。如果有若干个置位都一样是最好的话，则输出最小的一个。(位置 P 是从 0 开始的，它代表迟到的骑士在开始线上的位置。换言之，P 是在最好的解中，排在迟到的骑士前面的骑士数目。如果 P=0，则代表迟到的骑士是排在第一位，而若 P=N-1 则代表迟到的骑士是排在最后一位)。

## 子任务 1 [17 分]

你可以假设 $N \leq 500$.

## 子任务 2 [32 分]

你可以假设 $N \leq 5\ 000$.

## 子任务 3 [51 分]

你可以假设 $N \leq 100\ 000$.

## 实现细节

你必须只提交一个名为 tournament.c, tournament.cpp 或 tournament.pas 的程序文件。这个文件必须实现上述的相关函数并符合下面的说明。

### C/C++ 程序

```
intGetBestPosition(int N, int C, int R, int *K, int *S, int *E);
```

### Pascal 程序

```
functionGetBestPosition(N, C, R : LongInt; var K, S, E : array of LongInt) : LongInt;
```

这些子程序一定要根据上述的特点来编写。当然你也可自由地编写其他在计算过程中需要的函数。你所编写的程序不能与标准输入输出有任何的直接交互。也不能使用任何其他的文件。

### 样例评分器

样例评分器将提供需要以下输入格式的评测环境:

第 1 行: N, C, R;

第 2, ..., N 行: K[i];

第 N + 1, ..., N + C + 1 行: S[i], E[i].

## 时间及内存限制

时间限制: 1 秒

内存限制: 256 MB.