CCF 全国信息学奥林匹克联赛（NOIP2015）复赛

提高组 day2

（请选手务必仔细阅读本页内容）

一．题目概况

中文题目名称	跳石头	子串	运输计划
英文题目与子目录名	stone	substring	transport
可执行文件名	stone	substring	transport
输入文件名	stone.in	substring.in	transport.in
输出文件名	stone.out	substring.out	transport.out
每个测试点时限	1 秒	1 秒	1 秒
测试点数目	10	10	20
每个测试点分值	10	10	5
附加样例文件	有	有	有
结果比较方式	全文比较（过滤行末空格及文末回车）		
题目类型	传统	传统	传统
运行内存上限	128M	128M	256M

二．提交源程序文件名

对于 C++语言	stone.cpp	substring.cpp	transport.cpp
对于 C 语言	stone.c	substring.c	transport.c
对于 pascal 语言	stone.pas	substring.pas	transport.pas

三．编译命令（不包含任何优化开关）

对于 C++语言	g++ -o stone stone.cpp -lm	g++ -o substring substring.cpp -lm	g++ -o transport transport.cpp -lm
对于 C 语言	gcc -o stone stone.c -lm	gcc -o substring substring.c -lm	gcc -o transport transport.c -lm
对于 pascal 语言	fpc stone.pas	fpc substring.pas	fpc transport.pas

注意事项：

1、文件名（程序名和输入输出文件名）必须使用英文小写。

2、C/C++中函数 main()的返回值类型必须是 int，程序正常结束时的返回值必须是 0。

3、全国统一评测时采用的机器配置为：CPU AMD Athlon(tm) II x2 240 processor，2.8GHz，内存 4G，上述时限以此配置为准。

4、只提供 Linux 格式附加样例文件。

5、**特别提醒：评测在当前最新公布的 NOI Linux 下进行，各语言的编译器版本以其为准。**

1．跳石头

(stone.cpp/c/pas)

【问题描述】

一年一度的“跳石头”比赛又要开始了！

这项比赛将在一条笔直的河道中进行，河道中分布着一些巨大岩石。组委会已经选择好了两块岩石作为比赛起点和终点。在起点和终点之间，有 N 块岩石（不含起点和终点的岩石）。在比赛过程中，选手们将从起点出发，每一步跳向相邻的岩石，直至到达终点。

为了提高比赛难度，组委会计划移走一些岩石，使得选手们在比赛过程中的最短跳跃距离尽可能长。由于预算限制，组委会至多从起点和终点之间移走 M 块岩石（不能移走起点和终点的岩石）。

【输入格式】

输入文件名为 stone.in。

输入文件第一行包含三个整数 L，N，M，分别表示起点到终点的距离，起点和终点之间的岩石数，以及组委会至多移走的岩石数。

接下来 N 行，每行一个整数，第 i 行的整数 Di（0 < Di < L）表示第 i 块岩石与 起点的距离。这些岩石按与起点距离从小到大的顺序给出，且不会有两个岩石出现在同一个位置。

【输出格式】

输出文件名为 stone.out。

输出文件只包含一个整数，即最短跳跃距离的最大值。

【输入输出样例 1】

stone.in	stone.out
25 5 2 2 11 14 17 21	4

见选手目录下的 stone/stone1.in 和 stone/stone1.ans。

【输入输出样例 1 说明】

将与起点距离为 2 和 14 的两个岩石移走后，最短的跳跃距离为 4（从与起点距离 17 的岩石跳到距离 21 的岩石，或者从距离 21 的岩石跳到终点）。

【输入输出样例 2】

见选手目录下的 stone/stone2.in 和 stone/stone2.ans。

【数据规模与约定】

对于 20%的数据，0 ≤ M ≤ N ≤ 10。

对于 50%的数据，0 ≤ M ≤ N ≤ 100。

对于 100%的数据，0 ≤ M ≤ N ≤ 50,000，1 ≤ L ≤ 1,000,000,000。

2. 米仓(ricehub)

时间限制: 1.000 Sec 内存限制: 128 MB

题目描述

乡间有一条笔直而长的路称为“米道”。沿着这条米道上R块稻田，每块稻田的坐标均为一个1到L(含1和L)的整数。

这些稻田按照坐标以不减的顺序给出，即对于$0 \leq i < R$，稻田i 的坐标X_i满足

$1 \leq X_0 \leq \cdots \leq X_{R-1} \leq L$。

注意：可能有多块稻田位于同一个坐标上。

我们计划建造一个米仓用于储存尽可能多的稻米。和稻田一样，米仓将建在米道上，其坐标也是1到L之间的整数（含1和L）。这个米仓可以建在满足上述条件的任一个位置上，包括那些原来一个或多个稻田存在的位置。

在收获季节，每一块稻田刚好出产一满货车的稻米。为了将这些稻米运到米仓，需要雇用一位货机来运米。司机的收费是每一满货车运送一个单位的距离收取1元。

换言之，将稻米从特定的稻田运到米仓的费用在数值上等于稻田坐标与米仓坐标之差的绝对值。

不幸的是，今年预算有限，我们至多只能花费B元运费。你的任务是要帮我们找出一个建造米仓置，可以收集到尽可能多的稻米。

输入

第一行三个整数R, L, B。

接下来R行，从小到大排列，表示每个稻田的坐标。

注意：运送稻米的总预算可能会很高，预算的总数是以 64 位整数给出的，建议在运算 中使用 6数。在 C/C++中，请使用 long long 类型数据；在 Pascal 中，请用 Int64 类 型数据。

输出

输出一个整数表示最多能收集到的稻米。

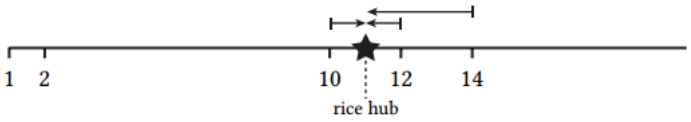
样例输入

```
5 20 6
1
2
10
12
14
```

样例输出

```
3
```

提示



对于这个测试数据，米仓的最优放置位置有多种可能：你可以将它放置在 10 和 14 之间（含 10 14）的任何一个位置上(上图展示了多种最优放置中的一种)。这样，就可以将位 于坐标 10,12 和 稻田出产的米运到米仓。而对于其中任一最优的米仓位置而言，其总 运输成本都不超过 6 元。很 显，在这个样例中，无论米仓选在何处，你最多只能收集到来 自 3 个稻田的稻米。

20%的数据，1 ≤ R ≤ 100,1 ≤ L ≤ 100, 0 ≤ B ≤ 10,000没有两个稻田在同一个坐标上。
40%的数据，1 ≤ R ≤ 500，1 ≤ L ≤ 10,000，0 ≤ B ≤ 1,000,000
70%的数据，1 ≤ R ≤ 5,000，1 ≤ L ≤ 1,000,000，0 ≤ B ≤ 2,000,000,000
100%的数据，1 ≤ R ≤ 100,000，1 ≤ L ≤ 1,000,000,000，0 ≤ B ≤ 2,000,000,000,000,000

3. 子串
(substring.cpp/c/pas)

【问题描述】

有两个仅包含小写英文字母的字符串A和B。现在要从字符串A中取出k个***互不重叠***的非空子串，然后把这 k 个子串按照其在字符串 A 中出现的顺序依次连接起来得到一个新的字符串，请问有多少种方案可以使得这个新串与字符串 B 相等？注意：子串取出的位置不同也认为是不同的方案。

【输入格式】

输入文件名为 substring.in。

第一行是三个正整数 n，m，k，分别表示字符串 A 的长度，字符串 B 的长度，以及问题描述中所提到的 k，每两个整数之间用一个空格隔开。

第二行包含一个长度为 n 的字符串，表示字符串A。

第三行包含一个长度为 m 的字符串，表示字符串B。

【输出格式】

输出文件名为 substring.out。

输出共一行，包含一个整数，表示所求方案数。**由于答案可能很大，所以这里要求输出答案对 1,000,000,007 取模的结果。**

【输入输出样例 1】

substring.in	substring.out
6 3 1 aabaab aab	2

见选手目录下 substring/substring1.in 与 substring/substring1.ans。

【输入输出样例 2】

substring.in	substring.out
6 3 2 aabaab aab	7

见选手目录下 substring/substring2.in 与 substring/substring2.ans。

【输入输出样例 3】

substring.in	substring.out
6 3 3 aabaab aab	7

见选手目录下 substring/substring3.in 与 substring/substring3.ans。

【输入输出样例说明】

所有合法方案如下：（加下划线的部分表示取出的子串）

样例 1：aab aab / aab aab

样例 2：a ab aab / a aba ab / a a ba ab / aab a ab
aa b aab / aa baa b / aab aa b

样例 3：a a b aab / a a baa b / a ab a a b / a aba a b
a a b a a b / a a ba a b / aab a a b

【输入输出样例 4】

见选手目录下 substring/substring4.in 与 substring/substring4.ans。

【数据规模与约定】

对于第 1 组数据：1≤n≤500，1≤m≤50，k=1；

对于第 2 组至第 3 组数据：1≤n≤500，1≤m≤50，k=2；

对于第 4 组至第 5 组数据：1≤n≤500，1≤m≤50，k=m；

对于第 1 组至第 7 组数据：1≤n≤500，1≤m≤50，1≤k≤m；

对于第 1 组至第 9 组数据：1≤n≤1000，1≤m≤100，1≤k≤m；

对于所有 10 组数据：1≤n≤1000，1≤m≤200，1≤k≤m。

4. 运输计划

（transport.cpp/c/pas）

【问题描述】

公元 2044 年，人类进入了宇宙纪元。

L 国有 n 个星球，还有 n-1 条*双向*航道，每条航道建立在两个星球之间，这 n-1 条航道*连通*了 L 国的所有星球。

小 P 掌管一家物流公司，该公司有很多个运输计划，每个运输计划形如：有一艘物流飞船需要从 ui 号星球沿*最快*的宇航路径飞行到 vi 号星球去。显然，飞船驶过一条航道是需要时间的，对于航道j，任意飞船驶过它所花费的时间为 tj，并且任意两艘飞船之间*不会*产生任何干扰。

为了鼓励科技创新，L 国国王同意小 P 的物流公司参与 L 国的航道建设，即允许小 P 把某一条航道改造成虫洞，飞船驶过虫洞*不消耗*时间。

在虫洞的建设完成前小 P 的物流公司就预接了 m 个运输计划。在虫洞建设完成后，这 m 个运输计划会*同时*开始，所有飞船*一起*出发。当这 m 个运输计划*都完成*时，小 P 的 物流公司的阶段性工作就完成了。

如果小 P 可以*自由选择*将哪一条航道改造成虫洞，试求出小 P 的物流公司完成阶段性工作所需要的最短时间是多少？

【输入格式】

输入文件名为 transport.in。

第一行包括两个正整数 n、m，表示 L 国中星球的数量及小 P 公司预接的运输计划的数量，星球从 1 到 n 编号。

接下来 n-1 行描述航道的建设情况，其中第 i 行包含三个整数 ai, bi 和 ti，表示第 i 条双向航道修建在 ai 与 bi 两个星球之间，任意飞船驶过它所花费的时间为 ti。

接下来 m 行描述运输计划的情况，其中第 j 行包含两个正整数 uj 和 vj，表示第 j 个 运输计划是从 uj 号星球飞往 vj 号星球。

【输出格式】

输出文件名为 transport.out。

共 1 行，包含 1 个整数，表示小 P 的物流公司完成阶段性工作所需要的最短时间。

【输入输出样例 1】

transport.in	transport.out
6 3 1 2 3 1 6 4 3 1 7 4 3 6 3 5 5 3 6 2 5 4 5	11

见选手目录下的 transport/transport1.in 与 transport/transport1.ans。

【输入输出样例 1 说明】

将第 1 条航道改造成虫洞：则三个计划耗时分别为：11、12、11，故需要花费的时间为 12。

将第 2 条航道改造成虫洞：则三个计划耗时分别为：7、15、11，故需要花费的时间为 15。

将第 3 条航道改造成虫洞：则三个计划耗时分别为：4、8、11，故需要花费的时间为 11。

将第 4 条航道改造成虫洞：则三个计划耗时分别为：11、15、5，故需要花费的时间为 15。

将第 5 条航道改造成虫洞：则三个计划耗时分别为：11、10、6，故需要花费的时间为 11。

故将第 3 条或第 5 条航道改造成虫洞均可使得完成阶段性工作的耗时最短，需要花费的时间为 11。

【样例输入输出 2】

见选手目录下的 transport/transport2.in 与 transport/transport2.ans。

【数据规模与约定】

所有测试数据的范围和特点如下表所示

<table>
<tr><th>测试点编号</th><th>n=</th><th>m=</th><th>约定</th></tr>
<tr><td>1</td><td rowspan="3">100</td><td>1</td><td></td></tr>
<tr><td>2</td><td rowspan="2">100</td><td rowspan="2">第 i 条航道连接 i 号星球与 i+1 号星球</td></tr>
<tr><td>3</td></tr>
<tr><td>4</td><td>2000</td><td>1</td></tr>
<tr><td>5</td><td>1000</td><td>1000</td><td rowspan="3">第 i 条航道连接 i 号星球与 i+1 号星球</td></tr>
<tr><td>6</td><td>2000</td><td>2000</td></tr>
<tr><td>7</td><td>3000</td><td>3000</td></tr>
<tr><td>8</td><td>1000</td><td>1000</td><td rowspan="5"></td></tr>
<tr><td>9</td><td>2000</td><td>2000</td></tr>
<tr><td>10</td><td>3000</td><td>3000</td></tr>
<tr><td>11</td><td>80000</td><td rowspan="2">1</td></tr>
<tr><td>12</td><td>100000</td></tr>
<tr><td>13</td><td>70000</td><td>70000</td><td rowspan="4">第 i 条航道连接 i 号星球与 i+1 号星球</td></tr>
<tr><td>14</td><td>80000</td><td>80000</td></tr>
<tr><td>15</td><td>90000</td><td>90000</td></tr>
<tr><td>16</td><td>100000</td><td>100000</td></tr>
<tr><td>17</td><td>80000</td><td>80000</td><td rowspan="4"></td></tr>
<tr><td>18</td><td>90000</td><td>90000</td></tr>
<tr><td>19</td><td>100000</td><td>100000</td></tr>
<tr><td>20</td><td>300000</td><td>300000</td></tr>
<tr><td>所有数据</td><td></td><td></td><td>$1 \leqslant a_i, b_i, u_j, v_j \leqslant n, 0 \leqslant t_i \leqslant 1000$</td></tr>
</table>

请注意常数因子带来的程序效率上的影响。