有序数组查询（query）

【题目描述】

$A=[A_0,A_1,A_2,\cdots,A_n]$ 是一个由 $n+1$ 个 $[0,N)$ 范围内整数组成的有序数组，满足 $0=A_0<A_1<A_2<\cdots<A_n<N$。（这个定义中蕴含了 n 一定小于 N。）

基于数组 A，对于 $[0,N)$ 范围内任意的整数 x，查询 $f(x)$ 定义为：数组 A 中小于等于 x 的整数里最大的数的下标。具体来说有以下两种情况：

1. 存在 $i<n$ 满足 $A_i\le x<A_{i+1}$

此时数组 A 中从 A_0 到 A_i 均小于等于 x，其中最大的数为 A_i，其下标为 i，故 $f(x)=i$。

2. $A_n\le x$

此时数组 A 中所有的数都小于等于 x，其中最大的数为 A_n，故 $f(x)=n$。

令 $sum(A)$ 表示 $f(0)$ 到 $f(N-1)$ 的总和，即：

$$sum(A)=\sum_{i=0}^{N-1}f(i)=f(0)+f(1)+f(2)+\cdots+f(N-1)$$

对于给定的数组 A，试计算 $sum(A)$。

【输入格式】

从标准输入读入数据。

输入的第一行包含空格分隔的两个正整数 n 和 N。

输入的第二行包含 n 个用空格分隔的整数 $A_1,A_2,\cdots,A_n$。

注意 A_0 固定为 0，因此输入数据中不包括 A_0。

【输出格式】

输出到标准输出。

仅输出一个整数，表示 $sum(A)$ 的值。

【样例 1 输入】

```
3 10
2 5 8
```

【样例 1 输出】

```
15
```

【样例 1 解释】

$A = [0, 2, 5, 8]$
$f(0) = f(1) = 0$
$f(2) = f(3) = f(4) = 1$
$f(5) = f(6) = f(7) = 2$
$f(8) = f(9) = 3$
$sum(A) = f(0) * 2 + f(2) * 3 + f(5) * 3 + f(8) * 2 = 15$

【样例 2 输入】

```
9 10
1 2 3 4 5 6 7 8 9
```

【样例 2 输出】

```
45
```

【子任务】

50% 的测试数据满足 $1 \leq n < N \leq 1000$；
全部的测试数据满足 $1 \leq n \leq 200$ 且 $1 \leq N \leq 10^7$。

查询问题新解（query2）

【题目背景】

$A = [A_0, A_1, A_2, \cdots, A_n]$ 是一个由 $n+1$ 个 $[0, N)$ 范围内整数组成的有序数组，满足 $0 = A_0 < A_1 < A_2 < \cdots < A_n < N$。基于数组 A，对于 $[0, N)$ 范围内任意的整数 x，查询 $f(x)$ 定义为：数组 A 中**小于等于** x 的整数里**最大**的数的**下标**。

如上一题“有序数组查询”中所定义，对于给定的数组 A 和整数 x，查询 $f(x)$ 是一个很经典的问题，可以使用二分搜索在 $O(\log n)$ 的时间复杂度内轻松解决。但对于这一问题，小 P 同学有些不同的想法。

【题目描述】

小 P 同学认为，如果事先知道了数组 A 中整数的分布情况，就能直接估计出其中小于等于 x 的最大整数的大致位置。比如说，如果 $A_1, A_2, \cdots, A_n$ 均匀分布在 $(0, N)$ 的区间，那么就可以估算出：

$$f(x) \approx \frac{(n+1) \cdot x}{N}$$

为了方便计算，小 P 首先定义了比例系数 $r = \lfloor \frac{N}{n+1} \rfloor$，其中 $\lfloor\ \rfloor$ 表示下取整，即 r 等于 N 除以 $n+1$ 的商。进一步的，小 P 用 $g(x) = \lfloor \frac{x}{r} \rfloor$ 表示自己估算出的 $f(x)$ 的大小，这里同样使用了下取整来保证 $g(x)$ 是一个整数。

显然，对于任意的询问 $x \in [0, N)$，$g(x)$ 和 $f(x)$ 越接近则说明小 P 的估计越准确。因此，小 P 用两者差的绝对值 $|g(x) - f(x)|$ 来表示在询问 x 上的误差。

为了整体评估小 P 同学提出的估算方法在数组 A 上的准确性，试计算：

$$error(A) = \sum_{i=0}^{N-1} |g(i) - f(i)| = |g(0) - f(0)| + \cdots + |g(N-1) - f(N-1)|$$

【输入格式】

从标准输入读入数据。

输入的第一行包含空格分隔的两个正整数 n 和 N。

输入的第二行包含 n 个用空格分隔的整数 $A_1, A_2, \cdots, A_n$。

注意 A_0 固定为 0，因此输入数据中不包括 A_0。

【输出格式】

输出到标准输出。

仅输出一个整数，表示 $error(A)$ 的值。

【样例 1 输入】

```
3 10
2 5 8
```

【样例 1 输出】

```
5
```

【样例 1 解释】

$A = [0, 2, 5, 8]$

$r = \lfloor \frac{N}{n+1} \rfloor = \lfloor \frac{10}{3+1} \rfloor = 2$

i	0	1	2	3	4	5	6	7	8	9
f(i)	0	0	1	1	1	2	2	2	3	3
g(i)	0	0	1	1	2	2	3	3	4	4
|g(i) - f(i)|	0	0	0	0	1	0	1	1	1	1

【样例 2 输入】

```
9 10
1 2 3 4 5 6 7 8 9
```

【样例 2 输出】

```
0
```

【样例 3 输入】

```
2 10
1 3
```

【样例 3 输出】

```
6
```

【样例 3 解释】

$A = [0, 1, 3]$
$r = \lfloor \frac{N}{n+1} \rfloor = \lfloor \frac{10}{2+1} \rfloor = 3$

i	0	1	2	3	4	5	6	7	8	9
f(i)	0	1	1	2	2	2	2	2	2	2
g(i)	0	0	0	1	1	1	2	2	2	3
|g(i) - f(i)|	0	1	1	1	1	1	0	0	0	1

【子任务】

70% 的测试数据满足 $1 \leq n \leq 1000$ 且 $1 \leq N \leq 10^5$；
全部的测试数据满足 $1 \leq n \leq 10^5$ 且 $1 \leq N \leq 10^9$。

【提示】

需要注意，输入数据（$A_1 \cdots A_n$）并不一定均匀分布在 $(0, N)$ 区间，因此总误差 $error(A)$ 可能很大。

登机牌条码（barcode）

【题目背景】

西西艾弗岛景色优美，游人如织。但是，由于和外界的交通只能靠渡船，交通的不便严重制约了岛上旅游业的发展。西西艾弗岛管委会经过努力，争取到了一笔投资，建设了一个通用航空机场。在三年紧锣密鼓的主体建设后，西西艾弗岛通用航空机场终于开始进行航站楼内部软硬件系统的安装和调试工程了。小 C 是机场运营公司信息部的研发工程师，最近，信息部门的一项重要任务是，研发登机牌自主打印系统。如图所示的是设计部门根据国际民航组织的行业标准设计的登机牌样张。

图 1: 登机牌

登机牌上最重要的部分就是最下方的机读条形码了。小 C 承担了生成机读条形码算法的开发工作。从被编码的数据到条形码，中间有好多步骤要走。小 C 请你来帮忙，让你帮忙处理一下数据编码的问题。

【题目描述】

登机牌上的条形码，是 PDF417 码。PDF417 码的结构如下图所示。

PDF417 码组成的基本元素是码元，所有的码元都是等大的矩形，填充有黑色或白色。码元先组成行，若干行堆叠组成整个 PDF417 码。每一行中，每 17 个码元表示一个**码字**。码字是 PDF417 编码中的最小数据单位。每个码字图案中，有交替排列的四个黑色矩形和四个白色矩形，这便是“417”的由来。每行开始和结尾有固定的起始和中止图案。与他们相邻的是行左侧和右侧标志，表示行号、行内码字个数等信息。中间的是有效数据区。编码的步骤是：先按照编码规则，将被编码的数据转换为码字；接着根据选定 PDF417 码的宽度，即每行有多少码字，以及冗余程度计算校验码字；最后将码

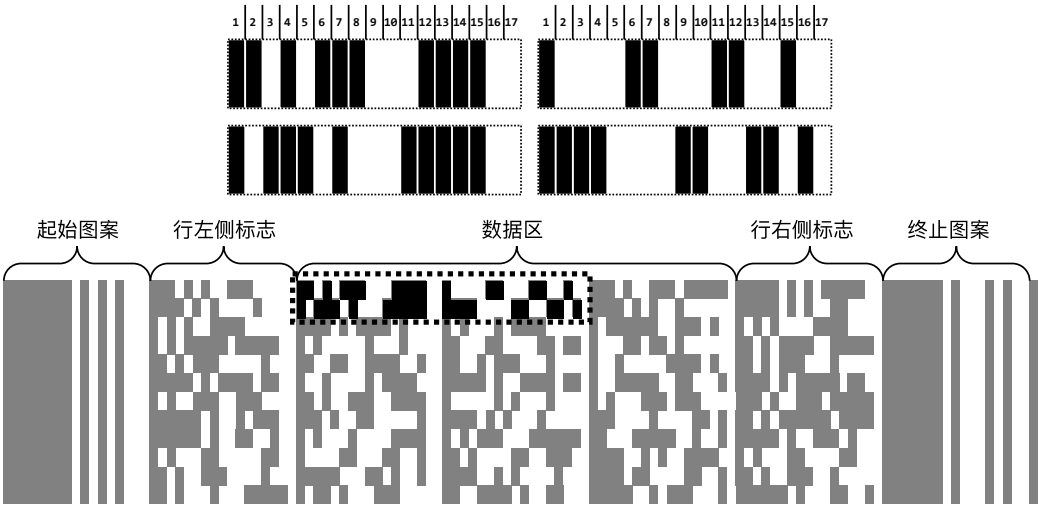

图 2: PDF417 码

字按规则转换为对应的图案，并按照从左至右，从上至下的的顺序填入有效数据区，并与起始终止图案和行左右标志拼合，形成完整的 PDF417 码。

每个码字是一个 0 至 928 之间的数字，每个码字可以编码两个输入字符。对于输入的被编码的数据，按照下表进行编码。编码器共有三种模式：大写字母模式、小写字母模式和数字模式。在编码开始时，编码器处于大写字母模式。编码器处于某种模式时，仅能编码对应类型的字符，如果需要编码其它类型的字符，需要通过特殊值切换到对应模式下。要进行模式切换，可以有多种切换方法。例如，要从大写模式切入小写模式，可以直接用 27 切入，也可以先用 28 切入数字模式后立刻再用 27 切入小写模式。你需要选择最短的方式进行切换，因此只有前一种方法是正确的。需要注意的是，从小写模式不能直接切入大写模式，必须要经过数字模式过渡。

值	大写模式	小写模式	数字模式
0	A	a	0
1	B	b	1
2	C	c	2
3	D	d	3
4	E	e	4
5	F	f	5
6	G	g	6
7	H	h	7
8	I	i	8
9	J	j	9
10	K	k	
11	L	l	
12	M	m	
13	N	n	
14	O	o	
15	P	p	
16	Q	q	
17	R	r	
18	S	s	
19	T	t	
20	U	u	
21	V	v	
22	W	w	
23	X	x	
24	Y	y	
25	Z	z	
27	小写		小写
28	数字	数字	大写
29	填充	填充	填充

按照这个方法可以得到一系列的不超过 30 的数字。如果有奇数个这样的数字，则在最后补充一个 29，使之成为偶数个。将它们两两成组，假设 H 和 L 是一组中连续出现的两个数字，那么可以得到一个码字是：

$$30 \times H + L$$

。

例如，要编码”HE1lo”，首先先根据字母表，产生数字序列：

```
H E    1    l  o
7 4 28 1 27 11 14
```

然后在末尾补充 29，并让它们两两成组：

```
(7, 4), (28, 1), (27, 11), (14, 29)
```

最后计算码字为：

```
214, 841, 821, 449
```

接下来要计算校验码。校验码字的数目，由校验级别确定。假设校验级别为 $s(0 \leq s \leq 8)$，则校验码字的数目为 $k = 2^{s+1}$。要计算校验码字，首先要确定数据码字。数据码字由以下数据按顺序拼接而成：

- 一个长度码字，表示全部数据码字的个数 n，包括该长度码字、有效数据码字、填充码字；
- 若干有效数据码字，是此前计算的码字序列；
- 零个或多个由重复的 900 组成的填充码字，使得码字总数恰能被有效数据区的行宽度整除。

设全部数据码字依次为 $d_{n-1}, d_{n-2}, \ldots, 0$；校验码字依次为 $c_{k-1}, c_{k-2}, \ldots, c_0$。那么校验码字按照如下方式计算：

取 k 次多项式 $g(x) = (x-3)(x-3^2)\ldots(x-3^k)$, n 次多项式 $d(x) = d_{n-1}x^{n-1} + \ldots d_{n-2}x^{n-2} + \ldots d_1 x + d_0$，找到不超过 $(k-1)$ 次的多项式 $r(x)$ 和多项式 $q(x)$ 使得

$$x^k d(x) \equiv q(x)g(x) - r(x)$$

。那么多项式 $r(x)$ 中 x 的 i 次项系数对 929 取模后的数字即为校验码字 c_i。

例如，如果要将 `HE1lo` 编码为 PDF417 条码，且有效数据区的行宽是 4 码字（即 68 码元），校验级别为 0。此时校验码字有两个。加上长度码字，一共是 7 个码字。因此需要补充一个填充码字，使总码字数能够被 4 整除。这样，用于计算校验码字的数据码字有：

```
6, 214, 841, 821, 449, 900
```

因此有 $g(x) = x^2 - 12x + 27$，$d(x) = 6x^5 + 214x^4 + 841x^3 + 821x^2 + 449x + 900$，不难得到 $r(x) = -32902164x + 98246277$，因此相应可以计算出 $c_1 = 229 \equiv -32902164 \mod 929$，$c_0 = 811 \equiv 98246277 \mod 929$。这样，全部码字序列即为：

```
6, 214, 841, 821, 449, 900, 229, 811
```

在本题中，你需要帮助小 C 完成的任务是，给定被编码的数据，计算出需要填入有效数据区的码字序列。被处理的数据中只含有大写字母、小写字母和数字。

【输入格式】

从标准输入读入数据。

输入的第一行包含两个用空格分隔的正整数 w、s，分别表示有效数据区每行能容纳的码字数和校验级别。保证 $w < 929$，$0 \leq s \leq 8$。

输入的第二行是一个非空字符串，仅包含大小写字母和数字，长度保证编码后全部数据码字的个数少于 929。

【输出格式】

输出到标准输出。

输出若干行，每行一个数字，表示编码后的全部码字序列。

【样例 1 输入】

```
4 0
HE1lo
```

【样例 1 输出】

```
6
214
841
821
449
900
229
811
```

【样例 1 解释】

本组数据即为此前用于说明编码过程的示例。

【子任务】

不要使用 markdown 原生的表格，使用下列方式渲染一个表格，其中表格存放在试题目录的 `tables` 子目录下。

原理上用一个二维的 json 表格或 python 表格存储，`null` 表示和上一行合并，不支持横向合并。建议用 python 的格式写，如例子中的 `data.pyinc`，这样可以根据数据生成；跟数据无关的表格则可以像 `table.json` 那样存储。

【提示】

这里是一个非常温馨的提示。

磁盘（disk）

【题目描述】

计算机中有 n 段程序（编号为 $1 \sim n$ ），它们共享一块大小为 m 的磁盘空间（编号为 $1 \sim m$ ），磁盘上的每个位置可以写入一个整数。

最初，磁盘上每个位置上的数都是 0 ，并不被任何程序占用。

现在，这 n 段程序同时执行，在某一时刻，某段程序可能对磁盘数据进行读写操作。

操作共 k 个，按时间先后顺序给出，具体操作如下：

0 id l r x ：编号为 id 的程序尝试向磁盘空间中 $[l,r]$ 位置上每个位置都写入一个整数 x。对于 $[l,r]$ 范围的每个位置，可能的结果如下：

- 若该位置目前不被任何程序占用，则成功写入整数 x ，并将其视为被程序 id 占用；
- 若该位置目前正被程序 id 自己占用，则这次写入的 x 可以覆盖之前写入的结果，此后该位置仍被程序 id 占用；
- 若该位置目前正被其他程序占用，则忽略该位置。

1 id l r ：程序 id 尝试删除磁盘中 $[l,r]$ 位置上的所有数据。

- 这一操作当且仅当 $[l,r]$ 区间内所有位置都正在被程序 id 占用时才能成功执行。
- 执行效果为将其中所有位置都解除占用，即恢复到可以被任意程序写入的状态。**但为了便于恢复数据，不会立即将全部位置重新覆盖成** 0 。
- 否则，认为此操作执行失败，不进行任何修改。

2 l r ：系统尝试直接删除磁盘中 $[l,r]$ 位置上的所有数据。

- 这一操作将区间内所有被任意程序占用的位置全部解除占用，未被占用的位置保持不变，**同样不会直接改变每个位置上的值**。

3 id l r ：程序 id 尝试恢复磁盘中 $[l,r]$ 位置上的所有数据。

- 这一操作当且仅当 $[l,r]$ 区间内所有位置都未被占用，且**上一次被占用是被程序** id **占用**时才能成功执行
- 执行效果为将其中所有位置恢复为**被程序** id **占用**的状态，同时由于之前删除操作并未改变其存储的值，因此本次操作也不需要改变每个位置上的值。
- 否则，认为此操作执行失败，不进行任何修改。

4 l r ：系统尝试恢复磁盘中 $[l,r]$ 位置上的所有数据。

- 这一操作当且仅当 $[l,r]$ 区间内所有位置都未被占用，且**曾经被占用过**时才能成功执行
- 执行效果为将其中所有位置恢复为**被上一个占用该位置的程序占用**的状态，同时由于之前删除操作并未改变其存储的值，因此本次操作也不需要改变每个位置上的值。
- 否则，认为此操作执行失败，不进行任何修改。

5 p ：尝试读取磁盘中 p 位置的数据，返回结果为两个整数。

- 如果该位置当前正被程序 id 占用且存储的值为 p ，返回结果为 $id\ p$ 。
- 否则，返回 $0\ 0$ 。

你需要实现一个程序模拟上述过程，并对于每个读取操作输出查询结果。

【输入格式】

从标准输入读入数据。

第一行：3 个正整数 n, m, k 。

接下来 k 行，每行若干个整数描述一个操作，格式如上所述。

【输出格式】

输出到标准输出。

对于每个读取操作输出一行，两个整数表示此次查询的结果。

【样例 1 输入】

```
This is the input of the first sample case.
Edit this file at `down/1.in`.
```

【样例 1 输出】

```
This is the output of the first sample case.
Edit this file at `down/1.ans`.
```

【样例 1 解释】

【样例 2】

见题目目录下的 ***2.in*** 与 ***2.ans***。

【子任务】

对于 100% 的数据，$1 \le n, k \le 2 \times 10^5,\ 1 \le m \le 10^9,\ 1 \le id \le n,\ 1 \le l \le r \le m,\ 1 \le p \le m,\ |x| \le 10^9$ 。

极差路径（min_max_path）

【题目背景】

众所周知，西西艾弗岛是一个非常缺钱的岛。

【题目描述】

为了从游客手中获取更多的经济利润，岛上仅有的三个小学生小 C、小 S 和小 P 建立了 n 个景点，编号依次从 1 到 n。编号为 i 的景点是第 i 个被修建的。由于越到后期经费越是不足，所以编号更大的景点通常更令人不满意——方便起见，假定编号为 i 的景点的令人不满意程度是 i。

有些景点之间修有双向可通行的道路，但是出于减少经费的考虑，他们只修了能使得所有景点连通的最少数量的道路，从而这些景点和其间的道路形成一棵树的结构。

对于每个游客而言，由于只修了 $n-1$ 条道路，所以他只能沿着树上的边参观，并且由于他不可能重复参观一个景点，所以他的游览路径一定是树上的一条简单路径。

现在西西艾弗岛希望制定一些推荐游览路径，但并非所有树上的路径都是合意的，因为这条路径上的景点令人不满意程度的极差可能过大，使游客产生这些景点质量不稳定的错觉。由于最开始的景点和最后的景点令人印象比较深刻，所以游客通常会把游览路径上的景点和这两个景点作比较。因此，最令人不满意的景点不能比这两个景点差太多，最优秀的景点也不能比这两个景点优秀太多。

具体来说，一条从 x 到 y 的游览路径（记作 (x,y)）是推荐的，当且仅当下式成立：

$$\min\{x,y\}-k_1 \le \min P(x,y) \le \max P(x,y) \le \max\{x,y\}+k_2$$

其中 $P(x,y)$ 表示一条从 x 出发到达 y 的简单路径上的点的令人不满意程度的数值的集合（包括 x 和 y，也就是 x 到 y 的简单路径上的点的编号的集合），$\min S$ 和 $\max S$ 分别表示集合 S 中的最小和最大值，k_1,k_2 是西西艾弗岛经过数次尝试后选取的两个给定的参数，保证 $k_1,k_2 \ge 0$。

特别的，容易验证 $x=y$ 时 (x,y) 总是推荐的。

现在西西艾弗岛想知道，一共有多少树上的简单路径作为游览路径是被推荐的？这里我们认为 (x,y) 和 (y,x) 是同一条路径。

【输入格式】

从标准输入读入数据。

第一行输入三个非负整数 n,k_1,k_2。

接下来 $n-1$ 行，每行两个正整数 x,y 表示树上的一条边。

【输出格式】

输出到标准输出。

输出一行一个非负整数表示答案。

【样例 1 输入】

```
2 0 0
1 2
```

【样例 1 输出】

```
3
```

【样例 1 解释】

容易验证 $(1,1),(1,2),(2,2)$ 都是推荐的游览路径，因此答案是 3。

【样例 2】

见题目目录下的 ***2.in*** 与 ***2.ans***。

【子任务】

<table>
<tr><th>测试点</th><th>$n \leq$</th><th>k_1</th><th>k_2</th><th>树的形态</th><th>堆性质</th></tr>
<tr><td>1</td><td rowspan="3">$2,000$</td><td rowspan="3">$\leq n$</td><td rowspan="3">$\leq n$</td><td rowspan="3">A_3</td><td rowspan="3">无</td></tr>
<tr><td>2</td></tr>
<tr><td>3</td></tr>
<tr><td>4</td><td rowspan="15">2×10^5</td><td rowspan="2">$=0$</td><td rowspan="4">$=0$</td><td rowspan="6">A_1</td><td>有</td></tr>
<tr><td>5</td><td>无</td></tr>
<tr><td>6</td><td rowspan="4">$\leq n$</td><td>有</td></tr>
<tr><td>7</td><td>无</td></tr>
<tr><td>8</td><td rowspan="2">$\leq n$</td><td>有</td></tr>
<tr><td>9</td><td rowspan="8">无</td></tr>
<tr><td>10</td><td rowspan="2">$=0$</td><td rowspan="4">$=0$</td><td rowspan="6">A_2</td></tr>
<tr><td>11</td></tr>
<tr><td>12</td><td rowspan="4">$\leq n$</td></tr>
<tr><td>13</td></tr>
<tr><td>14</td><td rowspan="2">$\leq n$</td></tr>
<tr><td>15</td></tr>
<tr><td>16</td><td rowspan="3">$=0$</td><td rowspan="6">$=0$</td><td rowspan="9">A_3</td><td>有</td></tr>
<tr><td>17</td><td rowspan="2">无</td></tr>
<tr><td>18</td></tr>
<tr><td>19</td><td rowspan="5">$\leq n$</td><td>有</td></tr>
<tr><td>20</td><td rowspan="2">无</td></tr>
<tr><td>21</td></tr>
<tr><td>22</td><td rowspan="3">$\leq n$</td><td>有</td></tr>
<tr><td>23</td><td rowspan="3">无</td></tr>
<tr><td>24</td></tr>
<tr><td>25</td></tr>
</table>

对于 100% 的数据，$1 \leq n \leq 2 \times 10^5, 0 \leq k_1, k_2, \leq n$，保证输入的 $n-1$ 条边一定构成一棵树。

A_1：树是一条链；

A_2：存在一个度数为 $n-1$ 的点；

A_3：树的形态无特殊性质。

堆性质：若取编号为 1 的点为根，则除 1 号点外，每个点的编号都比其父节点的编号大。

【提示】

请注意答案可能的取值范围。