全国青少年信息学奥林匹克竞赛
CCF NOI2022 浙江省选

第二试

时间：2022 年 5 月 4 日 08:30 ∼ 13:00

题目名称	面条	计算几何	深搜
题目类型	传统型	传统型	传统型
目录	noodle	geometry	dfs
可执行文件名	noodle	geometry	dfs
输入文件名	noodle.in	geometry.in	dfs.in
输出文件名	noodle.out	geometry.out	dfs.out
每个测试点时限	4.0 秒	1.0 秒	5.0 秒
内存限制	1024 MB	1024 MB	1024 MB
测试点数目	20	20	20
测试点是否等分	是	是	是

提交源程序文件名

对于 C++ 语言	noodle.cpp	geometry.cpp	dfs.cpp

编译选项

对于 C++ 语言	-lm -O2 -std=c++17

注意事项

1. 文件名（包括程序名，后缀名和输入输出文件名）必须使用英文小写。
2. 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为 0。
3. 提交的程序代码文件的放置位置请参照考场具体要求。
4. 因违反以上三点而出现的错误或问题，申诉时一律不予受理。
5. 若无特殊说明，输入文件中同一行内的多个整数、浮点数、字符串等均使用一个空格分隔。
6. 若无特殊说明，结果的比较方式为全文比较（过滤行末空格及文末回车）。
7. 程序可使用的栈内存空间限制与题目的内存限制一致。
8. 采用的机器配置为：Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz，内存 8GB。上述时限以此配置为准。
9. 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。
10. 最终评测时所用的编译命令中不含编译选项之外的任何优化开关。

面条（noodle）

【题目背景】

忍，爱丽丝，绫和阳子一众千辛万苦地总算出好了第一试，按原先计划，可怜会出第二试。

“不好了，可怜给我发信息说她降落后被拉去隔离 30 天了，没有电脑，出不了题”，绫突然收到了不幸的消息。

“那咋办？没idea了，编不出来了啊！”众人慌作一团。

看了看日期，离ZJOI还有一周。

欲知后事如何，请看下回分解。

【题目描述】

九条可怜是一个喜欢吃拉面的女孩子。

有一天她去吃拉面，她发现拉面师傅为她拉的是一个长度为 n 的面条，n 保证是偶数，一开始第 i 个位置调料的数量是 a_i。

如下过程称为一次“拉面”：

1. 将面条对折，面条的长度会变成 $\frac{n}{2}$，第 i 个位置的调料数量会变为原来第 i 个位置的调料与第 $n-i+1$ 个位置的调料数量之和，如果新面条第 i 个位置的调料数量为 b_i，那么满足 $b_i = a_i + a_{n-i+1}$。
2. 将面条拉回原来的长度 n，每个位置会变为两个位置，并且调料数量会均分，如果现在的第 i 个位置的调料数量是 a'_i，那么$a'_i = \frac{1}{2} \times b_{\lceil \frac{i}{2} \rceil}$。

现在对于一个固定的 x，你需要回答 q 个询问，每次面条经过 k 次“拉面”后，第 x 个位置的调料数量。你只需要求出答案对 998 244 353 取模的结果。具体地，即如果答案的最简分数表示为 $\frac{a}{b}$，输出 $a \times b^{-1} \bmod 998\,244\,353$。

【输入格式】

从文件 ***noodle.in*** 中读入数据。

第一行输入三个正整数 $test, T, seed$，代表测试点编号，数据组数和生成种子。

接下来输入 T 组数据，每组数据包含两行。

第一行输入四个正整数 $n, q, x, k_{\max}$，代表这组数据中面条的长度，询问的个数，询问的位置和生成询问中 k 的上限。

第二行输入 n 个非负整数，第 i 个整数 a_i 代表初始面条第 i 个位置的调料数量。

为了避免大量的输入与输出，q 个询问由我们提供的一个生成器生成。具体地，我们提供一个由C++书写的代码框架 `noodle_template.cpp` 供选手使用，见附录，同时在这里我们做一定量的说明：

首先我们从数据中依次读入两个 32 位整型变量 $test, T$，一个无符号 64 位长整型变量 $seed$。接下来循环 T 次，代表 T 组数据。

在每次循环中，我们对一组数据进行计算。首先依次读入三个 32 位整型变量 n, q, x，一个 64 位整型变量 $k_{\max}$。接下来读入 n 个 32 位整型变量放入数组 $a_1, \cdots, a_n$ 中。

接下来是生成 q 个询问的部分，每次调用 $rd()$ 函数，将 $seed$ 作为引用参数传入，将返回值（返回值为无符号 64 位长整型）对 $k_{\max}$ 取模的结果作为该次询问的参数 k，注意到 $seed$ 也会被修改。

【输出格式】

输出到文件 ***noodle.out*** 中。

输出 T 行，每行一个整数代表该组数据的答案。具体地，假设该组数据有 q 个询问，令第 i 个询问的答案为 Ans_i，那么需要你输出 $\oplus_{i=1}^{q}(\mathrm{Ans}_i \cdot i)$，**注意这里不需要取模**。$\oplus$ 指按位异或运算符。

【样例输入】

见下发文件中的noodle_ex1.in和noodle_ex2.in。

【样例输出】

见下发文件中的noodle_ex1.ans和noodle_ex2.ans。

【数据范围与提示】

对于所有测试点：保证 $T \le 10, \sum n \le 2 \times 10^6, \sum q \le 5 \times 10^7, k_{\max} \le 10^{18}, 1 \le x \le n, 0 \le a_i < 998\,244\,353, 0 \le seed \le 2^{60} - 1$，保证 n 是偶数。

注意，对于样例，测试点编号 $test$ 为 0。

每个测试点的具体限制见下表：

测试点编号	$\sum n \le$	$\sum q \le$	$k_{\max} \le$	特殊限制
1	500	500	500	无
2	2×10^6	2×10^6	10	无
3	2×10^6	2×10^6	10^{18}	$n = 2^k$
4	50	50	10^{18}	无
5, 6	150	150	10^{18}	无
7	2×10^6	2×10^6	10^{18}	$n = 98\,304$
8, 9	500	500	10^{18}	无
10, 11	5×10^3	2×10^6	10^{18}	无
12, 13	2×10^6	50	10^{18}	无
14, 15, 16	10^6	10^5	10^{18}	无
17, 18	2×10^6	2×10^7	10^{18}	无
19, 20	2×10^6	5×10^7	10^{18}	无

【样例解释】

对于样例 1：

第一组测试数据中，$\{a_i\}$ 初始为 $\{1, 4, 2, 3\}$。

操作一次后为 $\{2, 2, 3, 3\}$。

操作两次后为 $\{\frac{5}{2}, \frac{5}{2}, \frac{5}{2}, \frac{5}{2}\}$。

其中生成询问为：

询问位置：$x = 1$；

第一个询问：$k = 0$，$a_x = 1$；

第二个询问：$k = 1$，$a_x = 2$；

答案为 $(1 \times 1) \oplus (2 \times 2) = 4 \oplus 1 = 5$。

第二组测试数据中，$\{a_i\}$ 初始为 $\{6, 2, 5, 3, 1, 4\}$。

操作一次后为 $\{5, 5, \frac{3}{2}, \frac{3}{2}, 4, 4\}$。

操作两次后为 $\{\frac{9}{2}, \frac{9}{2}, \frac{9}{2}, \frac{9}{2}, \frac{3}{2}, \frac{3}{2}\}$。

其中生成询问为：

询问位置：$x = 3$；

第一个询问 $k = 2$，$a_x = \frac{9}{2}$，$\frac{9}{2} \equiv 499122181 \pmod{998244353}$；

第二个询问 $k = 0$，$a_x = 5$；

答案为 $(499122181 \times 1) \oplus (5 \times 2) = 499122181 \oplus 10 = 499122191$。

【附录】

```cpp
#include <bits/stdc++.h>
using namespace std;

unsigned long long rd(unsigned long long &x) {
	x ^= (x << 13);
	x ^= (x >> 7);
	x ^= (x << 17);
	return x;
}

int main() {
	int test, T;
	unsigned long long seed;
	scanf("%d%d%llu", &test, &T, &seed);
	for(int Case = 1; Case <= T; Case++) {
		int n, q, x;
		long long k_max;
		scanf("%d%d%d%lld", &n, &q, &x, &k_max);
		vector<int> a(n + 1);
		for(int i = 1; i <= n; i++) {
			scanf("%d", &a[i]);
		}
		for(int i = 1; i <= q; i++) {
			long long k = rd(seed) % k_max;
			/*
			Code your solution here.
			*/
		}
	}
}
```

计算几何（geometry）

【题目描述】

九条可怜是一个喜欢计算几何的女孩子，她画了一个特别的平面坐标系，其中 x 轴正半轴与 y 轴正半轴夹角为 60 度。

从中，她取出所有横纵坐标不全为偶数，且满足 $-2a+1 \le x \le 2a-1$，$-2b+1 \le y \le 2b-1$，$-2c+1 \le x+y \le 2c-1$ 的整点。

可怜想将其中一些点染色，但相邻的点不能同时染色。具体地，对于点 (x,y)，它和 $(x,y+1),(x,y-1),(x+1,y),(x-1,y),(x+1,y-1),(x-1,y+1)$ 六个点相邻，可结合样例解释理解。

可怜想知道在这个规则下最多能将多少点染色，以及染最多点的染色方案数。由于后者值可能很大，对于染色方案数，你只需要输出对 998 244 353 取模后的结果。**注意不需要将最多染色点数取模。**

【输入格式】

从文件 ***geometry.in*** 中读入数据。

第一行一个整数 T 代表数据组数。

接下来 T 行，每行三个整数 a,b,c 代表一组数据。

【输出格式】

输出到文件 ***geometry.out*** 中。

输出共 T 行，每行两个整数，代表最多能染的点数**（不取模）**和方案数对 998 244 353 取模的结果。

【样例输入】

见下发文件中的geometry_ex1.in。

【样例输出】

见下发文件中的geometry_ex1.ans。

【数据范围与提示】

对于所有测试点：$1 \le T \le 10, 1 \le a, b, c \le 10^6$ 。

每个测试点的具体限制见下表：

测试点编号	$a \le$	$b, c \le$	特殊限制
1	3	3	$a = b = c$
2	4	4	$a = b = c$
3	4	4	无
4	3	100	无
5, 6	3	1 000	无
7, 8	3	5 000	无
9, 10	100	100	$a = b = c$
$11 \sim 14$	100	100	无
15	10^5	10^5	$a = b = c$
16	10^5	10^5	无
17, 18	10^6	10^6	$a \cdot b \cdot c \le 10^6$
19	10^6	10^6	$a = b = c$
20	10^6	10^6	无

【样例解释】

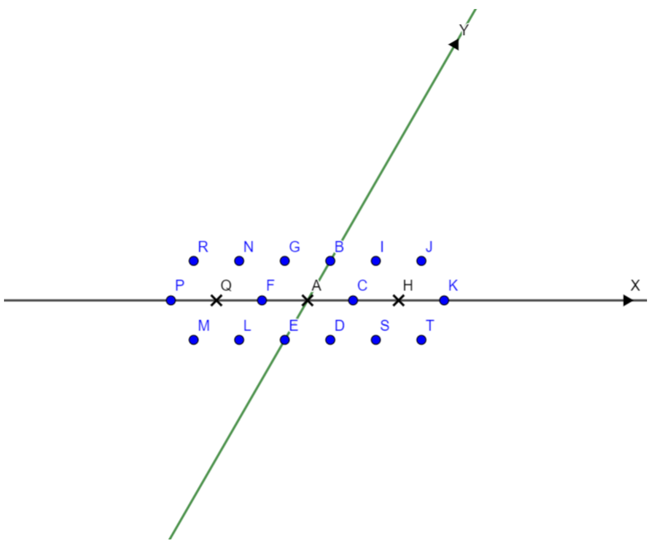
如下图所示，点 J 的坐标为 $(2, 1)$ ，点 F 的坐标为 $(-1, 0)$ ，点 H 的坐标为 $(2, 0)$ 。在这三个点中，只有点 H 是横纵坐标全为偶数的点。图中与点 A 距离为 1 的点有 $BCDEFG$ 六个点。

在样例的第一组数据中，满足条件的整点有 $RNGBIJPFCKMLEDST$。

最多能染 7 个点，方案共 4 种，具体为：$PNLBDJT$，$RMFBDJT$，$RMGECJT$，$RMGEISK$。

在样例的第二组数据中，满足条件的整点有 $GBIFCLED$ 。

最多能染 4 个点，方案共 1 种，具体为：$LGID$ 。



深搜（dfs）

【题目描述】

九条可怜是一个喜欢算法的女孩子，在众多算法中她尤其喜欢深度优先搜索（DFS）。

有一天，可怜得到了一棵有根树，树根为 $root$ ，树上每个节点 x 有一个权值 a_x。

在一棵树上从 x 出发，寻找 y 节点，如果使用深度优先搜索，则可描述为以下演算过程：

1. 将递归栈设置为空。
2. 首先将节点 x 放入递归栈中。
3. 从递归栈中取出栈顶节点，如果该节点为 y，则结束演算过程；否则，如果存在未访问的直接子节点，则以均等概率随机选择一个子节点加入递归栈中。
4. 重复步骤 3，直到不存在未访问的直接子节点。
5. 将上一级节点加入递归栈中，重复步骤 3。
6. 重复步骤 5，直至当前一级节点为 x，演算过程结束。

我们定义 $f(x,y)$ 合法当且仅当 y 在 x 的子树中。它的值为从 x 出发，对 x 的子树进行深度优先搜索寻找 y 期间访问过的所有节点（包括 x 和 y）权值最小值的期望。

九条可怜想知道对于所有合法的点对 (x,y) ，$\sum f(x,y)$ 的值。你只需要输出答案对 $998\,244\,353$ 取模的结果。具体地，如果答案的最简分数表示为 $\frac{a}{b}$ ，输出 $a \times b^{-1} \bmod 998\,244\,353$。

【输入格式】

从文件 ***dfs.in*** 中读入数据。

输入包含多组数据，第一行输入数据组数 T 。

对于接下来的每组数据，第一行两个整数 $n, root$ ，分别表示树的大小，树根的编号。

接下来一行 n 个整数 $a_1, a_2, ..., a_n$ ，表示树上每个节点的权值。

接下来 $n-1$ 行，每行包含两个整数 u, v ，表示 u 和 v 之间有一条树边。

【输出格式】

输出到文件 ***dfs.out*** 中。

对于每组数据，输出一行，包含一个整数，代表对于所有合法点对 (x,y) ，$\sum f(x,y)$ 对 $998\,244\,353$ 取模的结果。

【样例输入】

见下发文件中的dfs_ex1.in和dfs_ex2.in。

【样例输出】

见下发文件中的dfs_ex1.ans和dfs_ex2.ans。

【数据范围与提示】

对于所有测试点，满足 $1 \le T \le 100$，$\sum n \le 800\,000$，$1 \le n \le 400\,000$，$1 \le root, u, v \le n$，$1 \le a_i \le 10^9$。

每个测试点的具体限制见下表：

测试点编号	$\sum n \le$	$n \le$	特殊限制
1	50	10	无
$2 \sim 4$	40 000	5 000	无
$5 \sim 10$	400 000	100 000	无
11	800 000	400 000	树的生成方式随机
12	800 000	400 000	树是一条链
13	800 000	400 000	根的度数为$n-1$
$14 \sim 20$	800 000	400 000	无

对于测试点 11，树的生成方式为：以 1 为根，对于节点 $i \in [2, n]$，从 $[1, i-1]$ 中等概率随机选择一个点作为父亲。之后将编号随机重排。