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.
                        */
                }
        }
}
```