NOIp 模拟赛

题目名称	字符串	位运算	迷宫	树
题目类型	Special Judge	传统型	Special Judge	传统型
可执行文件名	string	bit	maze	tree
输入文件名	string.in	bit.in	maze.in	tree.in
输出文件名	string.out	bit.out	maze.out	tree.out
每个测试点时限	1.0 秒	1.0 秒	1.0 秒	1.0 秒
内存限制	512 MB	1024 MB	512 MB	512 MB
提交源程序文件名	string.cpp	bit.cpp	maze.cpp	tree.cpp
子任务/测试点数目	4	5	7	7
是否等分	否	否	否	否

编译选项： `-lm -O2 -std=c++14`

注意事项（请务必仔细阅读）

1. 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。
2. 若无特殊说明，结果比较方式为忽略行末空格、文末回车后的全文比较。
3. 选手提交的程序代码文件请**直接放在个人目录下，不需要建立子文件夹**。
4. 若无特殊说明，输入文件与输出文件中同一行的相邻整数均使用一个空格分隔。
5. 选手提交的程序源文件必须不大于 100 KB。
6. 程序可使用的栈空间内存限制与题目的内存限制一致。
7. 使用 `std::deque` 等 STL 容器时，请注意其内存空间消耗。
8. 评测时采用的机器配置为 AMD Ryzen 7 7500F，内存 32 GiB。上述时限以此配置为准。
9. 评测在 Windows 11 下进行，使用 LemonLime 进行评测。
10. 题目**大致按难度排序。**
11. 每道题会设置合理的**子任务依赖。**

字符串（string）

题目描述

给你一个长度为 $3n$ 的字符串 S，由字符 `A`、`B` 和 `C` 组成。

你可以执行下列操作任意次：

- 选择该字符串的某个子段和一个字符 ch（`A`、`B` 和 `C` 中的一个）。然后，用 ch 替换该子段中的所有字符。

你需要让 S 中恰好有 n 个 `A`、`B` 和 `C` 。请你求出最小的操作次数，并且输出方案。

输入格式

第一行输入一个整数 n，第二行输入一个由 `A`、`B` 或 `C` 构成的长度为 $3n$ 的字符串。

输出格式

第一行输出一个整数 k，表示最小的操作次数。

接下来 k 行，每行两个整数 l, r 和一个字符 ch，表示将从 l 到 r 的字符替换为 ch。

如果有多种方案，输出任意一种即可。

输入输出样例

样例输入 #1

```
1
AAA
```

样例输出 #1

```
2
1 1 B
2 2 C
```

样例输入 #2

```
1
ABC
```

样例输出 #2

```
0
```

样例输入 #3

```
3
ABABCABAB
```

样例输出 #3

```
1
1 2 c
```

数据范围及约定

对于所有的数据，$1 \leq n \leq 3 \times 10^5$。

Subtask 编号	特殊性质	分值
Subtask #1	$1 \leq n \leq 5$	10
Subtask #2	$1 \leq n \leq 15$	20
Subtask #3	保证字符串中只有 `A`，`B` 两种字符	20
Subtask #4	无	50

由于某些原因，本题不下发效验器和大样例。

位运算（bit）

题目描述

给定一个长度为 n 的序列 a，定义一个区间 $[l,r]$ 的划分为满足 $S_1 \cup S_2 = [l,r] \cap \mathbb{N}, S_1 \cap S_2 = \varnothing$ 的两个**非空**下标集合 S_1, S_2。

定义一个划分是好的，当且仅当 $\mathrm{or}_{i \in S_1} a_i = \mathrm{and}_{i \in S_2} a_i$，其中 $\mathrm{or}, \mathrm{and}$ 分别表示按位或、按位与。

有 q 次询问，每次询问区间 $[l,r]$ 是否存在一个划分是好的。

输入格式

第一行两个整数 n, q，第二行 n 个整数表示序列 a。

接下来 q 行每行两个整数 l, r，表示一次询问。

输出格式

对于每个询问，若这个区间是好的，输出 `YES`，否则输出 `NO`。

输入输出样例

样例输入 #1

```
4 4
1 7 3 11
1 1
1 2
1 3
1 4
```

样例输出 #1

```
NO
NO
NO
YES
```

对于第四个询问，可以将区间划分为 $\{1,3\}, \{2,4\}$（下标集合）。

样例 #2

见附件中的 `bit_2.in/out`。

该组样例满足 Subtask #2 的限制条件。

样例 #3

见附件中的 `bit_3.in/out`。

该组样例满足 Subtask #3 的限制条件。

样例 #4

见附件中的 `bit_4.in/out`。

该组样例满足 Subtask #4 的限制条件。

样例 #5

见附件中的 `bit_5.in/out`。

该组样例满足 Subtask #5 的限制条件。

数据范围及约定

对于所有的数据，$1 \leq n \leq 10^5, 1 \leq q \leq 10^5, 0 \leq a_i \leq 2^{30} - 1$。

Subtask 编号	特殊性质	分值
Subtask #1	$1 \leq n \leq 20, 1 \leq q \leq 5$	20
Subtask #2	保证 a_i 为 2 的整数次方	15
Subtask #3	保证 a_i 为 2 的整数次方 -1	20
Subtask #4	$0 \leq a_i \leq 2^{10} - 1$	20
Subtask #5	无	25

迷宫（maze）

题目描述

给你一个迷宫，这个迷宫可以看作一个 $n \times m$ 的网格图。定义 (i, j) 表示网格图中第 i 行第 j 列的点，则迷宫的起点是 $(1, 1)$，终点是 (n, m)。

玩家在这个迷宫里只能向下或者向右走，而有些网格的一些边框是不能通过的（比如从 $(1, 1)$ 到 $(1, 2)$ 的边是不能通过的）。定义这个迷宫的简单程度为从起点到终点的方案数。

给出这个迷宫的简单程度，然后让你构造出一个满足要求的迷宫。要求 $1 \leq n, m \leq 50$ 并且不能通过的边框个数小于等于 300。

输入格式

输入共一行一个正整数 T，表示迷宫简单程度。

输出格式

第一行输出两个正整数 n, m，分别表示迷宫的行数和列数。

第二行输出一个正整数 k，表示迷宫内不能通过的边框个数。

接下来 k 行，每行 4 个正整数 $x1, y1, x2, y2$，表示从 $(x1, y1)$ 到 $(x2, y2)$ 的边框不能通过。

如果有多种方案，输出任意一种即可。

特别的，在输出中，你需要保证 $1 \leq n, m \leq 50, 0 \leq k \leq 300$，每一对 $(x1, y1), (x2, y2)$ 是相邻的并且在迷宫内。否则可能会出现不可预料的错误。

输入输出样例

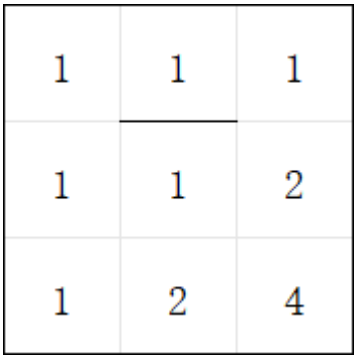
样例输入 #1

```
4
```

样例输出 #1

```
3 3
1
1 2 2 2
```

构造方案如图所示（网格内的数为从起点走到这个点的方案数）：



样例 #2

见附件中的 `maze_2.in/out`。

该组样例满足 Subtask #3 的限制条件。

样例 #3

见附件中的 `maze_3.in/out`。

该组样例满足 Subtask #6 的限制条件。

数据范围及约定

对于所有的数据，$0 \leq T \leq 10^{18}$。

Subtask 编号	特殊性质	分值
Subtask #1	保证存在一种方案使得 $n \leq 3, m \leq 4$	5
Subtask #2	$0 \leq T \leq 50$	10
Subtask #3	$0 \leq T \leq 1275$	10
Subtask #4	$0 \leq T \leq 2 \times 10^{14}$ 且存在一个 $k \in \mathbb{N}$ 使得 $T = 2^k$	10
Subtask #5	$0 \leq T \leq 2 \times 10^{14}$ 且存在一个 $k \in \mathbb{N}$ 使得 $T = 2^k - 1$	10
Subtask #6	$0 \leq T \leq 2 \times 10^{14}$	20
Subtask #7	无	35

评测方式

本题采用 Special Judge 进行评测，选手可以用下发文件中的效验器（命名为 `checker.cpp` 的文件）进行本地自测。

具体的使用方式如下：

首先编译文件，在终端内执行 `g++ checker.cpp -o checker -std=c++14 -O2`，如果系统不自带编译器，你可以直接使用带编译功能的编辑器进行编译。同时，你需要保证 `checker.cpp` 和下发的 `testlib.h` 在同一目录下。然后执行命令：

- 对于 Linux 系统，在终端内执行 `./checker <Input File> <Output File> <Answer File>`；
- 对于 Windows 系统，在终端内执行 `checker.exe <Input File> <Output File> <Answer File>`。

其中，`<Input File>` 指输入文件路径，`<Output File>` 指输出文件路径，`<Answer File>` 表示答案文件路径。使用时不需要加 `<` 和 `>`。**注意最终评测使用的效验器可能会与下发的不同。**

树（tree）

题目描述

给你一个 n 个点的，以一为根的树，每个点有两个值 a_i 和 b_i 。定义 $R(v)$ 为 v 祖先的集合（包括自己）。定义一个点 v 的权值为：

$$\left|\sum_{x\in R(v)} a_x\right| \times \left|\sum_{x\in R(v)} b_x\right|$$

请你支持两种操作：

- `1 x y`：将 a_x 加上 y，保证 y **非负**。
- `2 x`：输出以 x 为根的子树中最大的权值

输入格式

第一行，两个整数 n, q 表示树的大小和操作数。

第二行，$n-1$ 个整数，第 i 个数为 p_i ，表示点 i 的父亲是 p_i 。

第三行 n 个整数表示 a_i 。

第四行 n 个整数表示 b_i 。

接下来 q 行，每行表示一个操作。

输出格式

对每个询问操作输出一行表示其答案。

输入输出样例

样例输入 #1

```
5 6
1 1 2 2
10 -3 -7 -3 -10
10 3 9 3 6
2 1
2 2
1 2 6
2 1
1 2 5
2 1
```

样例输出 #1

```
100
91
169
240
```

样例 #2

见附件中的 `tree_2.in/out`。

该组样例满足 Subtask #2 的限制条件。

样例 #3

见附件中的 `tree_3.in/out`。

该组样例满足 Subtask #3 的限制条件。

样例 #4

见附件中的 `tree_4.in/out`。

该组样例满足 Subtask #4 的限制条件。

样例 #5

见附件中的 `tree_5.in/out`。

该组样例满足 Subtask #5 的限制条件。

样例 #6

见附件中的 `tree_6.in/out`。

该组样例满足 Subtask #6 的限制条件。

样例 #7

见附件中的 `tree_7.in/out`。

该组样例满足 Subtask #7 的限制条件。

数据范围及约定

对于所有数据，满足 $1 \leq n, q \leq 2 \times 10^5, 0 \leq |a_i|, |b_i|, y \leq 5000, 1 \leq p_i < i$。

请选手相信自己代码的常数并注意常数因子对代码效率的影响。

Subtask 编号	特殊性质	分值
Subtask #1	$n, q \leq 5 \times 10^3$	10
Subtask #2	保证只对叶子进行修改	10
Subtask #3	保证只对根节点进行修改	15
Subtask #4	保证只对叶子进行询问	10
Subtask #5	保证只对根节点进行询问	15
Subtask #6	$n, q \leq 5 \times 10^4$	20
Subtask #7	无	20

注：叶子节点指子树大小为 1 的点。