必可赛前公益众筹赛

第八试

时间: 2024 **年** 10 **月** 31 **日** 08:00 ~ 12:00

题目名称	特种训练	红色改造	势均力敌	车站爆破
题目类型	传统型	传统型	传统型	传统型
目录	`train`	`modification`	`equal`	`bomb`
可执行文件名	`train`	`modification`	`equal`	`bomb`
输入文件名	`train.in`	`modification.in`	`equal.in`	`bomb.in`
输出文件名	`train.out`	`modification.out`	`equal.out`	`bomb.in`
每个测试点时限	2.0 秒	2.0 秒	2.0 秒	2.0 秒
内存限制	512 MB	512 MB	1024 MB	512 MB
测试点数目	20	20	20	20
测试点是否等分	是	是	是	是

提交源程序文件名

对于 C++ 语言	`train.cpp`	`modification.cpp`	`equal.cpp`	`bomb.cpp`

编译选项

对于 C++ 语言	`-lm -O2 -std=c++14`

注意事项与提醒（请选手务必仔细阅读）

1. 选手请直接提交源程序至 becoder.com.cn 上的对应比赛。
2. 输入输出文件名必须使用英文小写。
3. 选手提交的源程序必须存放在**已建立**好的，且**带有样例文件和下发文件**的文件夹中，文件夹名称与对应试题英文名一致。
4. 文件名 (包括程序名和输入输出文件名) 必须使用英文小写。
5. C++ 中函数 main() 的返回值类型必须是 int，值必须为 0。
6. 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**。
7. 程序可使用的栈空间大小与该题内存空间限制一致。
8. 在终端中执行命令 ulimit -s unlimited 可将当前终端下的栈空间限制放大，但你使用的栈空间大小不应超过题目限制。
9. 每道题目所提交的**代码文件大小限制为 100KB**。
10. 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。

11. 输入文件中可能存在行末空格，请选手使用更完善的读入方式 (例如 scanf 函数) 避免出错。

12. 直接复制 PDF 题面中的多行样例，数据将带有行号，建议选手直接使用对应目录下的样例文件进行测试。

13. 使用 std::deque 等 STL 容器时，请注意其内存空间消耗。

14. 请务必使用题面中规定的的编译参数，保证你的程序在本机能够通过编译。此外**不允许在程序中手动开启其他编译选项**，一经发现，本题成绩以 0 分处理。

15. **对于因未遵守以上规则对成绩造成的影响，相关申诉不予受理**。

特种训练（train）

【题目描述】

小 L 是 Cyan 帝国中的高级 bot。别看它只是一只 bot，它还是 Cyan 帝国特种部队的一员。

小 L 为了准备接下来的一次任务，给自己安排了详细的训练计划，但是由于种种不可抗因素，它的计划全部都泡汤了，最后只能在家中狭小的区域进行训练。

为了合理的利用空间，小 L 将自己的家从左往右依次分为了 n 个区域，并在每一个区域中写上了 L 或 R 字符表示当站在这个区域时，下一步应该向左/向右移动。

但小 L 发现自己可能会一直运动，永远无法停止，所以进行了一些修改，当从一个区域离开时，该区域上的字符翻转（即 L 变为 R，R 变为 L），当从 1 区域向左走，或从 n 区域向右走时，小 L 会停止训练。

虽然小 L 是不会感到疲惫的，但它还是想知道从每一个区域出发，走多少步之后会停止。

注意从每一个区域出发的答案独立。

【输入格式】

从文件 ***train.in*** 中读入数据。

第一行一个正整数 n，表示区域个数。

第二行一个长度为 n 的字符串，表示初始序列。

【输出格式】

输出到文件 ***train.out*** 中。

共一行，第 i 个数字表示从区域 i 出发走多少步会停止。

【样例 1 输入】

```
3
RRL
```

【样例 1 输出】

```
5 6 3
```

见选手目录下的 *train/train1.in* 与 *train/train1.ans*。

【样例 1 解释】

从三个区域出发的路径分别为：

$1 \to 2 \to 3 \to 2 \to 1 \to out$

$2 \to 3 \to 2 \to 1 \to 2 \to 3 \to out$

$3 \to 2 \to 3 \to out$

【样例 2 输入】

```
10
RRLRLRRLLR
```

【样例 2 输出】

```
5 12 23 34 36 27 18 11 6 1
```

见选手目录下的 *train/train2.in* 与 *train/train2.ans*。

【样例 3】

见选手目录下的 *train/train3.in* 与 *train/train3.ans*。

【样例 4】

见选手目录下的 *train/train4.in* 与 *train/train4.ans*。

【数据范围】

对于 100% 的数据，$1 \le n \le 5 \times 10^5$。

测试点编号	$n \le$	特殊性质
$1 \sim 3$	100	无
$4 \sim 6$	10^5	保证不存在 R
$7 \sim 9$	10^5	保证 R 的总数量不超过 20
$10 \sim 12$	10^5	L，R 在初始序列中交替出现
$13 \sim 15$	10^5	无
$16 \sim 20$	5×10^5	

红色改造（modification）

【题目描述】

Red 共和国是 Cyan 帝国的眼中刺，为了自身的长远发展，Cyan 帝国派出了小 L 作为总指挥去进行情报收集与破坏工作。

由于 Cyan 帝国的 bot 与 Red 共和国的 bot 在基因上有着很大的不同，在进行潜入工作前，必须对小 L 进行基因改造，代号「红色改造」。

虽然小 L 是高级 bot，但是 bot 就是 bot，和所有的 bot 相同，小 L 的基因可以被抽象为一个 01 序列。为了让小 L 的基因与 Red 共和国的 bot 的基因更为相像，研究员抽象出了「价值」这一概念，定义一个 01 序列的「价值」为它的 **最长不下降子序列**，现在研究员想知道，如果将小 L 的基因划分为 k 个子段并进行重排，可以获得的最大「价值」是多少。

【输入格式】

从文件 ***modification.in*** 中读入数据。

第一行一个正整数 n，表示小 L 的基因长度。

第二行一个长度为 n 的字符串，表示小 L 基因的 01 序列。

【输出格式】

输出到文件 ***modification.out*** 中。

共一行，第 i 个数字表示 $k = i$ 时的最大「价值」。

【样例 1 输入】

```
8
01100100
```

【样例 1 输出】

```
5 7 7 8 8 8 8 8
```

见选手目录下的 *modification/modification1.in* 与 *modification/modification1.ans*。

【样例 1 解释】

$k = 1$ 时划分为子段 01100100，最长不下降子序列为 01100100。

$k = 2$ 时划分为子段 011, 00100，拼接为 00100011，最长不下降子序列为 00100011。

$k = 3$ 时划分为子段 011, 001, 00，拼接为 00001011，最长不下降子序列为 00001011。

$k = 4$ 时划分为子段 011, 00, 1, 00，拼接为 00000111，最长不下降子序列为 00000111。

$k > 4$ 时子段拼接结果与 $k = 4$ 相同。注意上面展示的划分，拼接方案与最长不下降子序列都可能不是唯一的。

【样例 2 输入】

```
20
10110001010110001101
```

【样例 2 输出】

```
12 14 14 16 16 17 17 18 18 19 19 20 20 20 20 20 20 20 20 20
```

见选手目录下的 *modification/modification2.in* 与 *modification/modification2.ans*。

【样例 3】

见选手目录下的 *modification/modification3.in* 与 *modification/modification3.ans*。

【数据范围】

对于 100% 的数据，$1 \le n \le 3 \times 10^5$。

测试点编号	$n \le$
$1 \sim 3$	8
$4 \sim 6$	20
$7 \sim 9$	200
$10 \sim 13$	2×10^3
$14 \sim 17$	8×10^4
$18 \sim 20$	3×10^5

势均力敌（equal）

【题目描述】

成功潜入 Red 共和国后，小 L 建立了自己的情报组，开始着手调查情报。

经过深入的调查，小 L 发现 Red 共和国虽然名面上是一个整体，但暗地中分为了两个大的敌对集团 A, B。

小 L 首先整理出了还未加入任何一个集团的 n 个重要人物，每一个重要人物都有一个能力值 a_i，小 L 非常好心，已经将 a 序列转换成了一个 **排列**。

小 L 事先将这些重要人物按照职权 **从小到大** 排序，为了更好的估量出两个集团的新生实力，小 L 定义了「核心人物」作为基准。一个重要人物是「核心人物」当且仅当他的能力值大于同一个集团中职权比他小的所有人的能力值。

小 L 希望使两边的「核心人物」人数相同，以此加剧他们的内耗。虽然小 L 已经用自己的最强 CPU 进行 dfs 搜索快速计算出了一种方案，但它希望你能帮助它验证答案。即你需要输出一个 **字典序最小** 的 01 字符串表示分配方案，若不存在分配方案使得两边的「核心人物」人数相同，输出 -1。

注意每一个重要人物必须加入一个集团，否则将不再会是重要人物。

【输入格式】

从文件 ***equal.in*** 中读入数据。

第一行一个正整数 n，表示重要人物的个数。

第二行给出一个长度为 n 的排列，第 i 个数字表示第 i 个人的能力值。注意这里的第 i 个人是职权第 i 小的。

【输出格式】

输出到文件 ***equal.out*** 中。

共一行，若无解，输出 -1，否则输出一个长为 n 的 01 串表示字典序最小的合法分配方案。

【样例 1 输入】

```
6
3 1 4 6 2 5
```

【样例 1 输出】

```
001001
```

见选手目录下的 *equal/equal1.in* 与 *equal/equal1.ans*。

【样例 1 解释】

两个集团中的人的能力值分别为 $\{3,1,6,2\},\{4,5\}$。「核心人物」人数均为 2。

【样例 2 输入】

```
5
1 2 3 4 5
```

【样例 2 输出】

```
-1
```

见选手目录下的 *equal/equal2.in* 与 *equal/equal2.ans*。

【样例 3】

见选手目录下的 *equal/equal3.in* 与 *equal/equal3.ans*。

【样例 4】

见选手目录下的 *equal/equal4.in* 与 *equal/equal4.ans*。

【数据范围】

对于 100% 的数据，$1 \le n \le 2 \times 10^5$，保证 a 为一个排列。

测试点编号	$n \le$	特殊限制
$1 \sim 3$	20	无
$4 \sim 5$	2×10^5	保证 a 数组先增后减
$6 \sim 7$	50	无
$8 \sim 9$	100	
$10 \sim 11$	200	
$12 \sim 13$	500	
$14 \sim 15$	1000	
$16 \sim 17$	5×10^3	
$18 \sim 20$	2×10^5	

车站爆破（bomb）

【题目描述】

由于小 L 居高自傲，鼠目寸光，恃强凌弱，滥用职权，Cyan 帝国决定将它调回国内重新改造，小 L 当然不乐意了，它决定在此之前干一番大事来挽回自己的名声。

小 L 盯上了 Red 共和国内一个有着奇怪设计的运输站——众清北站。众清北站的过往车辆络绎不绝，虽然如此，但为了方便车站管控，只存在一个出入口，每一辆车只能从该出入口处进出。小 L 给每一辆车根据它运输的货物设定了一个价值，并窃取了众清北站的车辆进出数据，希望找到一个合适的时间进行爆破。小 L **按时间顺序** 记录了 n 组数据，每一组数据包含两个参数 k_i, v_i。

1. $k_i = 0$，代表一辆价值为 v_i 的车进入车站。
2. $k_i = 1$，代表最后进入车站的 v_i 辆车离开。

但是众清北站的数据记录员玩忽职守，数据经常出现差错，所以小 L 不得不经常重新调整数据。

具体来说，小 L 会进行 m 次修改，每一次对其中的一组数据进行修改，在修改中甚至 k_i 都会发生改变!!! 更为离奇的是，可能当 $k_i = 1$ 时，离开的车辆数量甚至大于原有车辆数量，小 L 只好假装此时车站被清空。

小 L 到底是受不了了，最终放弃了爆破计划，但它还是想知道在进行完每一次修改操作之后，车站中剩余车辆价值总和。

【输入格式】

从文件 ***bomb.in*** 中读入数据。

第一行两个整数 n, m，分别表示数据个数与修改次数。

接下来 n 行，每行两个整数 k_i, v_i，含义与题目描述中相同。

接下来 m 行，每行三个整数 x_j, k_j, v_j，表示一次修改操作，其中 x_j 表示数据编号。

【输出格式】

输出到文件 ***bomb.out*** 中。

共 m 行，每行一个整数，表示第 i 次修改之后的答案。

【样例 1 输入】

```
4 3
0 1
0 2
1 2
0 3
```

```
2 0 3
3 1 1
4 1 1
```

【样例 1 输出】

```
3
4
0
```

见选手目录下的 *bomb/bomb1.in* 与 *bomb/bomb1.ans*。

【样例 2 输入】

```
10 10
1 2
0 10
1 1
1 3
0 9
0 1
0 2
0 7
0 10
0 6
8 1 3
1 0 7
2 0 6
1 1 2
7 0 4
2 1 1
10 1 3
9 0 2
4 0 8
3 0 2
```

【样例 2 输出】

```
16
```

```
16
16
16
16
16
0
0
0
0
```

见选手目录下的 *bomb/bomb2.in* 与 *bomb/bomb2.ans*。

【样例 3】

见选手目录下的 *bomb/bomb3.in* 与 *bomb/bomb3.ans*。

【样例 4】

见选手目录下的 *bomb/bomb4.in* 与 *bomb/bomb4.ans*。

【数据范围】

对于 100% 的数据，$1 \le n, m \le 2 \times 10^5$，任意时刻车辆的价值 $\le 10^4$。

测试点编号	$n, m \le$	特殊性质
$1 \sim 3$	2000	无
$4 \sim 5$	2×10^5	保证在所有的修改操作中，只会修改车的价值
$6 \sim 9$	5×10^4	保证在任意时刻，$\forall k_i = 1$，$v_i = 1$
$10 \sim 13$	5×10^4	无
$14 \sim 17$	2×10^5	保证在任意时刻，$\forall k_i = 1$，$v_i = 1$
$18 \sim 20$	2×10^5	无