

北京信息学集训队结业考试

BJOI2018

第一试

时间：2018 年 4 月 14 日 08:00 ~ 13:00

题目名称	二进制	染色	求和
题目类型	传统型	传统型	传统型
可执行文件名	binary.exe	color.exe	sum.exe
输入文件名	binary.in	color.in	sum.in
输出文件名	binary.out	color.out	sum.out
每个测试点时限	2.0 秒	1.0 秒	2.0 秒
内存限制	512 MB	512 MB	512 MB
测试点数目	10	10	10
每个测试点分值	10	10	10

提交源程序文件名

对于 C++ 语言	binary.cpp	color.cpp	sum.cpp
对于 C 语言	binary.c	color.c	sum.c
对于 Pascal 语言	binary.pas	color.pas	sum.pas

编译选项

对于 C++ 语言	- O2 - Wl,--stack=536870912	- O2 - Wl,--stack=536870912	- O2 - Wl,--stack=536870912
对于 C 语言	- O2 - Wl,--stack=536870912	- O2 - Wl,--stack=536870912	- O2 - Wl,--stack=536870912
对于 Pascal 语言	-O2	-O2	-O2

二进制 (binary)

【题目描述】

pupil 发现对于一个十进制数，无论怎么将其的数字重新排列，均不影响其是不是 3 的倍数。他想研究对于二进制，是否也有类似的性质。于是他生成了一个长为 n 的二进制串，希望你对于这个二进制串的一个子区间，能求出其有多少位置不同的连续子串，满足在重新排列后（可包含前导 0）是一个 3 的倍数。两个位置不同的子区间指开始位置不同或结束位置不同。由于他想尝试尽量多的情况，他有时会修改串中的一个位置，并且会进行多次询问。

【输入格式】

从文件 *binary.in* 中读入数据。

输入第一行包含一个正整数 n ，表示二进制数的长度。

之后一行 n 个空格隔开的整数，保证均是 0 或 1，表示该二进制串。

之后一行一个整数 m ，表示询问和修改的总次数。

之后 m 行每行为 1 i ，表示 pupil 修改了串的第 i 个位置（0 变成 1 或 1 变成 0），或 2 $l\ r$ ，表示 pupil 询问的子区间是 $[l,r]$ 。

串的下标从 1 开始。

【输出格式】

输出到文件 *binary.out* 中。

对于每次询问，输出一行一个整数表示对应该询问的结果。

【样例 1 输入】

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

【样例 1 输出】

```
2
3
```

【样例 1 解释】

对于第一个询问，区间 $[2,2]$ 只有数字 0，是 3 的倍数，区间 $[1,3]$ 可以重排成 $011_{(2)} = 3_{(10)}$ ，是 3 的倍数，其他区间均不能重排成 3 的倍数。

对于第二个询问，全部三个区间均能重排成 3 的倍数（注意 00 也是合法的）。

【子任务】

对于 20% 的数据， $1 \leq n, m \leq 100$ 。

对于 50% 的数据， $1 \leq n, m \leq 5000$ 。

对于 100% 的数据， $1 \leq n, m \leq 100000$ ， $l \leq r$ 。

染色 (color)

【题目描述】

pupil 喜欢给图的顶点染颜色。有一天, master 想刁难他, 于是给了他一个无重边和自环的无向图, 并且对每个点分别给了一个大小为 2 的颜色集合, pupil 只能从这个集合中选一种颜色给这个点染色。master 希望 pupil 的染色方案使得没有两个有边相连的点被染了相同的颜色。

现在 pupil 想知道, 是否无论 master 的颜色集合是什么, 他均有办法按照要求染色。

【输入格式】

从文件 *color.in* 中读入数据。

输入包含多组数据。

第一行一个正整数 T , 表示数据组数。

之后每组数据第一行两个空格隔开的整数 n, m , 表示这个无向图的点数和边数。

之后 m 行, 每行两个空格隔开的正整数 i, j , 表示图中的一条连接点 i 和点 j 的边。

图的节点从 1 开始标号。

【输出格式】

输出到文件 *color.out* 中。

对于每组数据, 如果 pupil 无论如何均能染色, 输出一行一个字符串 YES, 否则输出一行一个字符串 NO。

【样例 1 输入】

```
3
6 9
1 2
1 4
1 6
3 2
3 4
3 6
5 2
5 4
5 6
```

2 1
1 2
3 3
1 2
1 3
2 3

【样例 1 输出】

NO
YES
NO

【样例 1 解释】

对于第一组数据，如果第一个点和第二个点的集合为 $\{A, B\}$ ，第三个点和第四个点的集合为 $\{A, C\}$ ，第五个点和第六个点的集合为 $\{B, C\}$ ，则奇数点至少使用了两种颜色，偶数点至少使用了两种颜色，因此至少有一个奇数点和一个偶数点颜色相同。但每两个奇数点和每两个偶数点之间均有边，因此无法满足“没有两个有边相连的点被染了相同的颜色”。

对于第二组数据，无论两个集合是什么，第一个点随便染它的集合中的其中一种颜色，第二个点染它的集合中某个与第一个点不同的颜色即可。

对于第三组数据，如果三个点的集合均是 $\{A, B\}$ ，那么无法满足“没有两个有边相连的点被染了相同的颜色”。

【子任务】

对于 10% 的数据， $1 \leq n \leq 3$ 。

对于 20% 的数据， $1 \leq n \leq 6$ 。

对于 50% 的数据， $1 \leq n \leq 1000$ ， $0 \leq m \leq 2000$ 。

对于 100% 的数据， $1 \leq n \leq 10000$ ， $0 \leq m \leq 20000$ ， $1 \leq T \leq 10$ 。

求和 (sum)

【题目描述】

master 对树上的求和非常感兴趣。他生成了一棵有根树，并且希望多次询问这棵树上一段路径上所有节点深度的 k 次方和，而且每次的 k 可能是不同的。此处节点深度的定义是这个节点到根的路径上的边数。他把这个问题交给了 pupil，但 pupil 并不会这么复杂的操作，你能帮他解决吗？

【输入格式】

从文件 *sum.in* 中读入数据。

第一行包含一个正整数 n ，表示树的节点数。

之后 $n-1$ 行每行两个空格隔开的正整数 i, j ，表示树上的一条连接点 i 和点 j 的边。

之后一行一个正整数 m ，表示询问的数量。

之后每行三个空格隔开的正整数 i, j, k ，表示询问从点 i 到点 j 的路径上所有节点深度的 k 次方和。由于这个结果可能非常大，输出其对 998244353 取模的结果。

树的节点从 1 开始标号，其中 1 号节点为树的根。

【输出格式】

输出到文件 *sum.out* 中。

对于每组数据输出一行一个正整数表示取模后的结果。

【样例 1 输入】

```
5
1 2
1 3
2 4
2 5
2
1 4 5
5 4 45
```

【样例 1 输出】

```
33
503245989
```

【样例 1 解释】

以下用 $d(i)$ 表示第 i 个节点的深度。

对于样例中的树，有 $d(1) = 0, d(2) = 1, d(3) = 1, d(4) = 2, d(5) = 2$ 。

因此第一个询问答案为 $(2^5 + 1^5 + 0^5) \bmod 998244353 = 33$ ，第二个询问答案为 $(2^{45} + 1^{45} + 2^{45}) \bmod 998244353 = 503245989$ 。

【子任务】

对于 30% 的数据， $1 \leq n, m \leq 100$ 。

对于 60% 的数据， $1 \leq n, m \leq 1000$ 。

对于 100% 的数据， $1 \leq n, m \leq 300000, 1 \leq k \leq 50$ 。

【提示】

数据规模较大，请注意使用较快速的输入输出方式。