2022 年陕西省 NOI 省队选拔赛试题

时间：2022 年 5 月 8 日 08:30 ～ 14:00

题目概况

题目名称	垃圾回收	军队	倍增	数位
题目类型	传统型	传统型	传统型	传统型
目录	garbage	military	double	digit
可执行文件名	garbage	military	double	digit
输入文件名	garbage.in	military.in	double.in	digit.in
输出文件名	garbage.out	military.out	double.out	digit.out
每个测试点时限	2.0 秒	6.0 秒	2.0 秒	6.0 秒
内存限制	512 MB	1 GB	512 MB	1 GB
测试点数目	20	20	25	25
测试点是否等分	是	是	是	是
评测方式	全文比较	全文比较	Special Judge	全文比较

提交源程序文件名

对于 C++ 语言	garbage.cpp	military.cpp	double.cpp	digit.cpp

编译选项

对于 C++ 语言	-lm -O2 -std=c++14

注意事项

1. 文件名（程序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是 0。
3. 程序可以使用的栈空间大小与该题内存空间一致。
4. 如无特殊说明，全文比较默认忽略行末空格和文末回车。
5. 评测时使用的 CPU 型号为 Intel® Core™ i7-10750H CPU @ 2.60GHz，上述时间限制以此为准。

垃圾回收（garbage）

【题目描述】

通常的情况下，编程语言在管理内存时会进行如下的选择：

- 让用户进行手动内存管理（C，C++，Rust 等），这会收获很好的性能，但是给用户提供了很大的编程负担。
- 使用垃圾回收系统（Java，Go 等），这需要维护一个运行时系统，并且在内存使用和程序性能方面造成了许多不可预测的负担。

尽管存在许多的问题，目前最通用的自动化内存管理手段始终为 Tracing Garbage Collector。这种做法的最基础的思路是维护对象间的引用关系，形成一张图，每次回收时通过扫描引用关系推导出已经无法被访问到的对象，释放它们占用的内存。而这种传统的做法最大的问题在于维护引用链需要造成很大的开销，并且随着维护的对象越多，扫描的代价也会越大。

小 L 是一个喜欢思考的女孩子，她发现维护 Garbage Collector 是一件非常复杂的事情，于是她决定考虑一个更简单的模型（注意它与任何现实中的 GC 规则可能是完全不同的！）。

对于一个 n 个点 m 条边的无向图，没有重边自环，点和边均从 1 开始标号。其中每个节点代表一个占用了一定内存的对象，每条边对应一个引用关系（注意这里的引用关系是**无向**的），程序从第 0 秒开始运行，在第 $q+1$ 秒结束运行。对于 $i = 1, 2, 3, ..., q$ 的每个时刻 i ，程序会进行以下两种操作之一：

- DELETE i，删除边 (x_i, y_i) ，保证不会删除已经被删除的边。
- GC，进行一次内存回收，即杀死所有从 1 号点出发不能访问到的点，释放它们占用的内存。（注意这里对节点的删除不会删除与这些点相连的边）

你可以认为这些操作是被瞬间执行完成的，在所有操作执行后，也就是第 $q+1$ 秒，程序结束，删除所有剩余的节点（包括 1 号点）。

第 i 个节点占用的内存为 a_i。现在请你求出整个程序的开销，即 $\sum_{i=1}^{n} a_i \cdot t_i$，这里 t_i 表示第 i 个点存活的时间，在第 0 秒，所有节点都是存活的。

【输入格式】

从文件 ***garbage.in*** 中读入数据。

输入的第一行是三个正整数 n, m, q，分别表示对象的个数，引用关系的个数，操作的个数。

接下来 m 行，每行 2 个正整数 x_i, y_i 表示第 i 条引用关系的两个端点。

接下来 q 行，每行为以下两种形式之一，第 i 行表示程序在第 i 秒进行的操作：

- 一个字符串 DELETE 和一个正整数 x，含义见【题目描述】。
- 一个字符串 GC，含义见【题目描述】。

接下来的一行有 n 个正整数 $a_1, a_2, \ldots, a_n$，表示每个对象占用的内存大小。

【输出格式】

输出到文件 ***garbage.out*** 中。

输出一行一个整数表示程序的开销，含义见【题目描述】。

【样例输入 1】

见选手目录下的 ***garbage/garbage1.in***。

```
6 6 8
1 2
2 3
2 4
1 4
2 5
1 6
GC
DELETE 5
DELETE 3
GC
DELETE 1
GC
DELETE 2
GC
1 2 3 4 5 6
```

【样例输出 1】

见选手目录下的 ***garbage/garbage1.ans***。

```
149
```

【样例解释 1】

在第 4 秒时，节点 5 被删除。

在第 6 秒时，节点 2, 3 被删除。

在第 9 秒时，节点 1, 4, 6 被删除。

因此。程序的开销为 $5 \times 4 + (2 + 3) \times 6 + (1 + 4 + 6) \times 9 = 20 + 30 + 99 = 149$。

【样例 2】

见选手目录下的 ***garbage/garbage2.in*** 和 ***garbage/garbage2.ans***。

本组数据满足测试点 6 的限制。

【样例 3】

见选手目录下的 ***garbage/garbage3.in*** 和 ***garbage/garbage3.ans***。
本组数据满足测试点 11 的限制。

【数据规模与约定】

对于全部数据，$1 \leq n, m, q \leq 4 \times 10^5$，$1 \leq a_i \leq 10^8$。
具体的数据规模与约定见下表。

测试点编号	n	m	q	特殊约定
$1 \sim 2$	≤ 500	≤ 500	≤ 500	无
$3 \sim 5$	≤ 3000	≤ 3000	≤ 3000	
$6 \sim 10$	≤ 5000	≤ 5000	≤ 5000	
$11 \sim 14$	$\leq 2 \times 10^5$	$n-1$	$\leq 2 \times 10^5$	保证一开始图是一棵树
$15 \sim 16$	$\leq 2 \times 10^5$	$\leq 2 \times 10^5$	$\leq 2 \times 10^5$	无
$17 \sim 20$	$\leq 4 \times 10^5$	$\leq 4 \times 10^5$	$\leq 4 \times 10^5$	

军队（military）

【题目描述】

R 国的历史非常悠久。

R 国有 n 个城市，国内有 C 个党派，分别记为 $1, 2, \ldots, C$。由于 R 国的版图非常长，这 n 个城市的位置可以近似为坐标轴上的 n 个点。在历史的最初，记载了第 i 个城市归属党派 c_i，城中有数量为 a_i 的军队。

R 国的历史上，经常发生以下三种事件：

1. 党派 y 进行了一次游说，使城市 l 到城市 r 的所有归属党派 x 的城市全部归属了 y。

2. 党派 x 进行了一次征兵，使城市 l 到城市 r 的所有归属党派 x 的城市中的军队数量增加了 v。

3. 城市 l 到城市 r 之间的所有城市爆发了战争。这场战争的规模可以描述为两地之间的所有城市中的军队数量之和。注意战争不一定发生在不同党派之间，归属同一个党派的一些城市内部也可能发生内战。由于 R 国的医护系统足够先进，战争不会造成军队数量的减少。

小 N 是一个喜欢历史的女孩子，最近她想整理一下 R 国的战争史，特别是每场战争的规模。但是由于 R 国的历史实在太长了，她用纸和笔进行运算实在力不从心。于是她找到了你，希望你写一个程序，统计出 R 国历史上所有战争的规模。

【输入格式】

从文件 ***military.in*** 中读入数据。

输入的第一行是三个正整数 n, q, C，分别表示城市的个数，事件的个数，和 R 国国内党派的个数。

接下来一行有 n 个正整数 $a_1, a_2, \ldots, a_n$，表示每个城市内初始的军队数。

接下来一行有 n 个正整数 $c_1, c_2, \ldots, c_n$，表示每个城市初始归属的党派。

接下来 q 行，每行 3 到 5 个正整数，表示一次事件：

第一个正整数 op 表示事件的类型。$op = 1, 2, 3$ 分别表示【题目描述】中所述的游说，征兵和战争事件。

对于每个游说事件，接下来有 4 个正整数 l, r, x, y，意义见【题目描述】。

对于每个征兵事件，接下来有 4 个正整数 l, r, x, v，意义见【题目描述】。

对于每个战争事件，接下来有 2 个正整数 l, r，意义见【题目描述】。

【输出格式】

输出到文件 ***military.out*** 中。

对于每个战争事件，输出一行一个整数，表示此次战争的规模。

【样例输入 1】

见选手目录下的 ***military/military1.in***。

```
5 7 3
1 2 4 8 16
1 2 3 2 3
2 2 4 2 32
3 1 4
1 1 5 3 1
2 2 5 1 64
3 2 4
2 1 3 3 128
3 3 5
```

【样例输出 1】

见选手目录下的 ***military/military1.ans***。

```
79
142
188
```

【样例解释 1】

最初，五个城市的军队数量分别为 $1, 2, 4, 8, 16$，归属的党派分别为 $1, 2, 3, 2, 3$。
发生的事件依次为：

- 党派 2 尝试在城市 $2, 3, 4$ 征兵，归属党派 2 的城市 $2, 4$ 各增加了 32 军队。
- 城市 1 和 4 之间的所有城市爆发了战争，规模为 $1 + 34 + 4 + 40 = 79$。
- 党派 1 在城市 $1, 2, 3, 4, 5$ 进行了一次游说，使得原本归属党派 3 的城市 $3, 5$ 归属了党派 1。
- 党派 1 尝试在城市 $2, 3, 4, 5$ 征兵，归属党派 1 的城市 $3, 5$ 各增加了 64 军队。
- 城市 2 和 4 之间的所有城市爆发了战争，规模为 $34 + 68 + 40 = 142$。
- 党派 3 尝试在城市 $1, 2, 3$ 征兵，但是党派 3 现在不拥有任何城市，因此并没有成功征兵。
- 城市 3 和 5 之间的所有城市爆发了战争，规模为 $68 + 40 + 80 = 188$。

因此你的程序应该依次输出 $79, 142, 188$。

【样例 2】

见选手目录下的 ***military/military2.in*** 和 ***military/military2.ans***。
本组数据满足测试点 4 的限制。

【样例 3】

见选手目录下的 ***military/military3.in*** 和 ***military/military3.ans***。
本组数据满足测试点 14 的限制。

【样例 4】

见选手目录下的 ***military/military4.in*** 和 ***military/military4.ans***。
本组数据满足测试点 18 的限制。

【数据规模与约定】

对于全部数据，$1 \le n, q, C \le 2.5 \times 10^5$，$1 \le a_i, v \le 10^8$，$1 \le c_i, x, y \le C$。
具体的数据规模与约定见下表。

测试点编号	n, q	C	特殊约定
1	≤ 20	≤ 20	无
2	≤ 50	≤ 50	
3	≤ 300	≤ 300	
4	≤ 5000	≤ 5000	
5	$\le 10^5$	≤ 10	
6	$\le 1.5 \times 10^5$		
7	$\le 2 \times 10^5$		
8	$\le 2.5 \times 10^5$		
9	$\le 1.5 \times 10^5$	$\le 1.5 \times 10^5$	对于所有操作，保证 $l = 1, r = n$
10	$\le 2.5 \times 10^5$	$\le 2.5 \times 10^5$	
11	$\le 1.5 \times 10^5$	$\le 1.5 \times 10^5$	对于所有操作 $1, 2$，保证 $l = 1, r = n$
12	$\le 2 \times 10^5$	$\le 2 \times 10^5$	
13	$\le 2.5 \times 10^5$	$\le 2.5 \times 10^5$	
14	$\le 1.5 \times 10^5$	$\le 1.5 \times 10^5$	保证不存在操作 1
15	$\le 2.5 \times 10^5$	$\le 2.5 \times 10^5$	
16	$\le 10^5$	$\le 10^5$	无
17	$\le 1.5 \times 10^5$	$\le 1.5 \times 10^5$	
18	$\le 2 \times 10^5$	$\le 2 \times 10^5$	
19	$\le 2.5 \times 10^5$	$\le 2.5 \times 10^5$	
20			

倍增（double）

【题目描述】

小 Z 是一个喜欢编程的女孩子。

这天，她在做一道编程题的时候偶然发现了一个神奇的整数 142857。

$142857 \times 2 = 285714$，而 285714 的所有数位恰好是 142857 的一个排列。

她很好奇，有没有更大的满足这种性质的整数。

她写了一个搜索的程序，发现了一些更大的有趣的数：

$26835741 \times 2 = 53671482$

$0987312654 \times 2 = 1974625308$

...

她不满足于解决十进制下这样的问题，于是她想知道，是否在 B 进制下存在一个 n 位正整数 x，满足 $2x$ 的所有数位在 B 进制下是 n 的所有数位的一个排列。

由于她讨厌数字 0，因此她还要求对于任意 $1 \le i \le n$，x 和 $2x$ 在 B 进制下的第 i 位不能同时为 0。

【输入格式】

从文件 ***double.in*** 读入数据。

输入包含多组数据。

输入的第一行是一个正整数 T，表示数据组数。

接下来 T 行，第 i 行包含两个正整数 n 和 B，表示第 i 组数据。

【输出格式】

输出到文件 ***double.out*** 中。

对于每组数据，输出一行。

若本组数据有解，按照从高位到低位的顺序输出 n 个非负整数，表示你找到的答案在 B 进制下的值。

否则只需要输出一个数 -1。

【样例输入 1】

见选手目录下的 ***double/double1.in***。

```
3
6 10
3 3
6 7
```

【样例输出 1】

见选手目录下的 ***double/double1.ans***。

```
1 4 2 8 5 7
-1
0 3 5 3 1 6
```

【样例解释 1】

第一组数据的解释参见【题目描述】。

对于第二组数据，可以通过枚举所有的 n 位 B 进制数说明一定不能找到这样的正整数。

对于第三组数据，$2x$ 的 7 进制表示为 $103635_{(7)}$，因此这是一个满足题意的答案。

注意此样例的答案文件仅表明了一种可能的合法答案，不表明答案文件恰好对应标准程序的输出。

【样例 2】

见选手目录下的 ***double/double2.in*** 和 ***double/double2.ans***。

本组数据满足测试点 3 的限制。

注意此样例的答案文件仅表明了一种可能的合法答案，不表明答案文件恰好对应标准程序的输出。

【样例 3】

见选手目录下的 ***double/double3.in*** 和 ***double/double3.ans***。

本组数据满足测试点 17 的限制。

注意此样例的答案文件仅表明了一种可能的合法答案，不表明答案文件恰好对应标准程序的输出。

【提示】

由于答案可能不唯一，我们下发了校验器 `checker.cpp` 和库文件 `testlib.h`。

可以使用以下命令编译 `checker.cpp`：

```
g++ -o checker checker.cpp -O2 -std=c++14
```

编译得到可执行文件 `checker` 后，你可以使用以下方式测试你的答案：

`checker <input> <output> <answer>`：利用选手目录下的 `double*.ans` 可以用来检验你的答案在样例测试点 `double*.in` 的正确性。

`checker <input> <output> <output>`：会检查你对这组数据的所有**有解输出**是否符合题目要求。注意以此种方式测试的时候，输出无解总会被报告为合法。

请选手注意多组数据之间的清空问题。

【数据规模与约定】

对于全部数据，$1 \le T \le 10^4$，$2 \le \sum B \le 2 \times 10^5$，$1 \le \sum n \le 2 \times 10^5$，$n \ge 1$，$B \ge 2$。具体的数据规模与约定见下表。

测试点编号	n	B	T	特殊约定
1	≤ 8	≤ 8	≤ 10	无
2			$\le 10^4$	
3	$\le 2 \times 10^5$		≤ 10	
4			$\le 10^4$	
5				
6	≤ 15	≤ 15	≤ 100	
7	≤ 40	≤ 40		
8	≤ 100	≤ 100		
9	≤ 300	≤ 300		
10	≤ 1000	≤ 1000		
11	≤ 3000	≤ 3000		
12	≤ 15000	≤ 15000		
13	≤ 50000	≤ 50000		
14	$\le 2 \times 10^5$	$\le 2 \times 10^5$		
15	≤ 200	≤ 200	$\le 10^4$	$n \ge 100$
16	≤ 5000	≤ 5000		
17	$\le 2 \times 10^5$	$\le 2 \times 10^5$		
18	≤ 300	≤ 300		$B = 3k - 1$, k 为正整数
19	$\le 2 \times 10^5$	$\le 2 \times 10^5$		
20	≤ 300	≤ 300		$B = 3k$, k 为正整数
21	$\le 2 \times 10^5$	$\le 2 \times 10^5$		
22	≤ 100	≤ 100		无
23	≤ 5000	≤ 5000		
24	$\le 2 \times 10^5$	$\le 2 \times 10^5$		
25				

数位（digit）

【题目描述】

小 S 是一个喜欢数数的女孩子。

有一天，她在睡前躺在床上数数，当她数到 977431 的时候，她终于困了，并且决定睡觉。但此时她突然发现这个数字的各位数码是单调不增的！她觉得这相当有趣，于是她又睡不着了。

她想知道有多少个数在 L, R 之间，并且它的各位数码是单调不增的。但这个问题太无聊了。

她又想知道有多少数对 (a, b) 在 L, R 之间，并且 $(a + b)$ 的各位数码是单调不增的。但这个问题也太无聊了。

终于，她想到了一个有趣一些的问题：

给定整数 L, R, k，求有多少个 k 维向量 $(a_1, a_2, ..., a_k)$ 满足 $(a_1 + a_2 + ... + a_k)$ 的数码是单调不增的，并且 $\forall i \in [1, k], L \leq a_i \leq R$。

她不会了。

由于答案可能很大，你只需要帮她求出答案对 998244353 取模的结果。

【输入格式】

从文件 ***digit.in*** 读入数据。

输入包含三行，第一行包含一个正整数 L，第二行包含一个正整数 R，第三行包含一个正整数 k，具体意义见【题目描述】。

【输出格式】

输出到文件 ***digit.out*** 中。

输出一行一个非负整数，表示满足上述要求的 k 维向量 $(a_1, a_2, ..., a_k)$ 的个数对 998244353 取模的值。

【样例输入 1】

见选手目录下的 ***digit/digit1.in***。

```
1
100
2
```

【样例输出 1】

见选手目录下的 ***digit/digit1.ans***。

```
3728
```

【样例输入 2】

见选手目录下的 ***digit/digit2.in***。

```
19260817
1000000000
3
```

【样例输出 2】

见选手目录下的 ***digit/digit2.ans***。

```
28745082
```

【样例输入 3】

见选手目录下的 ***digit/digit3.in***。

```
114514233
1919810233
10
```

【样例输出 3】

见选手目录下的 ***digit/digit3.ans***。

```
135934411
```

【样例 4】

见选手目录下的 ***digit/digit4.in*** 和 ***digit/digit4.ans***。

【样例 5】

见选手目录下的 ***digit/digit5.in*** 和 ***digit/digit5.ans***。

【数据规模与约定】

对于全部数据，$1 \le L \le R < 10^{1000}$，$1 \le k \le 50$。

具体的数据规模与约定见下表。

测试点编号	R	k
1	$< 10^6$	1
2		10
3		20
4		30
5		50
6	$< 10^{17}$	10
7		
8		20
9		30
10		50
11	$< 10^{50}$	2
12		10
13	$< 10^{100}$	2
14		3
15		10
16	$< 10^{200}$	3
17		10
18	$< 10^{300}$	
19		
20		20
21	$< 10^{500}$	10
22		20
23	$< 10^{1000}$	30
24		50
25		