冲刺 NOI2023 模拟试题

Day2

（请选手务必仔细阅读本页内容）

一．题目概况

题目名称	合成小丹	路过中丹	膜拜大丹
题目类型	传统型	传统型	传统型
目录	merge	pass	worship
可执行文件名	merge	pass	worship
输入文件名	merge.in	pass.in	worship.in
输出文件名	merge.out	pass.out	worship.out
每个测试点时限	1.0 秒	2.0 秒	2.0 秒
内存限制	512MB	512MB	512MB
测试点数目	20	20	20
每个测试点分值	5	5	5

二．提交源程序文件名

对于 C++ 语言	merge.cpp	pass.cpp	worship.cpp

三．编译选项

对于 C++ 语言	-lm -O2	-lm -O2	-lm -O2

四．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/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、全国统一评测时采用的机器配置为：Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz，内存 32GB。上述时限以此配置为准。

4、只提供 Linux 格式附加样例文件。

5、特别提醒：评测在当前最新公布的 NOI Linux 下进行，各语言的编译器版本以其为准。

1. 合成小丹

（merge）

【题目描述】

丹最喜欢做的两件事情分别是踩人和位运算，所以现在他要用位运算来踩你。

丹给了你 n 个范围在 $[0, 2^w - 1]$ 内的非负整数，你需要执行下面两种操作共 $n-1$ 次：

1. 选择两个非负整数 x 和 y，将两个非负整数合成成一个非负整数 z，其中 $z = \lfloor \frac{(x|y)}{2} \rfloor$。（这里的运算符 | 表示的是按位或）
2. 选择一个非负整数 x 并将其删去。

不难发现每次操作后你拥有的整数数量都会减少恰好一个，所以在 $n-1$ 次操作后你会拥有恰好一个非负整数，你需要**最小化**剩下来的这个整数的值。你只需要输出最后这个非负整数的值。

【输入格式】

从文件 ***merge.in*** 中读入数据。

本题包含多组数据，输入第一行一个整数 T 表示数据组数，对于每组测试数据：

第一行两个正整数 n, w，含义见题面。

第二行 n 个非负整数 $a_1, a_2, \ldots, a_n$，描述你一开始拥有哪些数。

【输出格式】

输出到文件 ***merge.out*** 中。

对于每组测试数据：

输出一行一个非负整数表示最后你剩下来的值最小是什么。

【样例 1 输入】

```
3
3 4
9 10 12
4 3
7 7 7 7
7 3
5 2 0 1 3 1 4
```

【样例 1 输出】

```
5
1
0
```

【样例 1 解释】

在第一组数据中一种最优的操作方案：

一开始你拥有的整数有 $\{9,10,12\}$；第一次操作选择删除整数 12，你拥有的整数有 $\{9,10\}$；第二次操作选择合并整数 9 和 10，$9|10=11,\lfloor 11/2\rfloor=5$，你拥有的整数有 $\{5\}$；最后你剩下的整数为 5，不难发现没有更优的做法了。

在第二组数据中一种最优的操作方案：

一开始你拥有的整数有 $\{7,7,7,7\}$；第一次操作选择合并整数 7 和 7，$7|7=7,\lfloor 7/2\rfloor=3$，你拥有的整数有 $\{3,7,7\}$；第二次操作选择合并整数 7 和 7，你拥有的整数有 $\{3,3\}$；第三次操作选择合并整数 3 和 3，$3|3=3,\lfloor 3/2\rfloor=1$，你拥有的整数有 $\{1\}$；最后你剩下的整数为 1，不难发现没有更优的做法了。

在第三组数据中一种最优的操作方案：

删掉除了 0 以外的其他整数。

【样例 2 】

见选手目录下的 ***merge/merge2.in*** 与 ***merge/merge2.ans***。

该样例满足测试点 $1\sim 2$ 的性质。

【样例 3 】

见选手目录下的 ***merge/merge3.in*** 与 ***merge/merge3.ans***。

该样例满足测试点 $6\sim 8$ 的性质。

【数据范围】

对于所有测试点，满足 $1 \leq T \leq 10, 1 \leq n \leq 10^5, 0 \leq w \leq 60, a_i \in [0, 2^w - 1]$。每个测试点的具体限制见下表：

测试点编号	n	w	特殊性质
$1 \sim 2$	≤ 6	≤ 60	无
$3 \sim 5$	≤ 8		
$6 \sim 8$	≤ 300	≤ 12	
$9 \sim 12$	≤ 5000	≤ 18	
$13 \sim 15$		≤ 40	
$16 \sim 17$	$\leq 10^5$	≤ 60	A
$18 \sim 20$			无

特殊限制 A：保证所有 a_i 都可以表示成 2 的整数次幂减一的形式。

2. 路过中丹

（pass）

【题目描述】

丹不仅是 OI 之神、whk 之神，还是游戏之神。

对于一个字符串 T，定义**一次行走**为你选择一个任意长度（不妨设长度为 m）的正整数序列 $t_1, t_2, \ldots, t_m$，其中 $\forall i \in [1, m], t_i \in [1, |T|]$、$\forall i \in [2, m], |t_i - t_{i-1}| = 1$、$t_1 \neq t_m$ 并且 $T_{t_1} T_{t_2} \ldots T_{t_m}$ 是一个回文串，在这次行走后我们认为了你经过了 $t_1, t_2, \ldots, t_m$ 这些位置各一次。

称一个字符串是“配得上丹”的，当且仅当你可以通过**若干次**行走使得这个字符串的每个位置都被经过**至少一次**。

现在给定一个长度为 n 的字符串 S，有 q 次询问，第 i 次询问给出两个数 l_i, r_i，你需要判断 S 中 l_i 到 r_i 的这个子串是否是“配得上丹”的。

【输入格式】

从文件 ***pass.in*** 中读入数据。

第一行一个正整数 n 表示字符串 S 的长度。

第二行一个长度为 n 的**仅包含英文小写字母**的字符串描述 S。

第三行一个正整数 q 表示询问数量。

接下来 q 行，每行两个正整数 l_i, r_i 表示一次询问。

【输出格式】

输出到文件 ***pass.out*** 中。

方便起见，你只需要输出长度为 q 的 01 串，其中第 i 个位置等于 1 当且仅当第 i 次询问的字符串是“配得上丹”的。

【样例 1 输入】

```
7
danaand
3
2 6
4 5
1 3
```

【样例 1 输出】

```
110
```

【样例 1 解释】

第一次询问的字符串为 anaan，方便起见令 T=anaan，那么你可以行走一次 $(1,2,3,4,5,4)$，满足起点终点不重合并且得到的字符串 anaana 是一个回文串。

第二次询问的字符串为 aa，方便起见令 T=aa，那么你可以行走一次 $(2,1)$，满足起点终点不重合并且得到的字符串 aa 是一个回文串。

第三次询问的字符串为 dan，不难发现这个字符串“配不上丹”。

【样例 2 输入】

```
8
bcdbacab
10
5 6
2 7
1 5
1 5
2 7
4 8
4 6
2 4
1 5
3 3
```

【样例 2 输出】

```
0100110000
```

【样例 3 】

见选手目录下的 ***pass/pass3.in*** 与 ***pass/pass3.ans***。

该样例满足测试点 $6 \sim 9$ 的性质。

【数据范围】

对于所有测试点，满足 $1 \leq n, q \leq 10^6, 1 \leq l_i \leq r_i \leq n$。

每个测试点的具体限制见下表：

测试点编号	n	q	特殊限制
$1 \sim 2$	≤ 5	≤ 10	无
$3 \sim 5$	≤ 50	≤ 500	
$6 \sim 9$	≤ 2000	≤ 2000	
$10 \sim 13$	$\leq 10^5$	$\leq 10^5$	
$14 \sim 15$	$\leq 10^6$	$\leq 10^6$	A
$16 \sim 20$			无

特殊限制 A：保证给定字符串 S 中只包含两种字母“a”和“b”。

3. 膜拜大丹

（worship）

【题目描述】

丹是万物之神，所以你想去膜拜丹。

现在有两个国家信仰丹，不妨记作国家 A 和国家 B。国家 A 有 n 座城市，编号为 $1 \sim n$；国家 B 有 m 座城市，编号为 $1 \sim m$。

国家 A 和国家 B 之间有单向航线连接，具体地，有长度为 n 数组 a 和长度为 m 的数组 b，国家 A 的第 i 座城市有单向航线可以到达国家 B 的编号为 $1 \sim a_i$ 的这些城市，国家 B 的第 j 座城市有单向航线可以到达国家 A 的编号为 $1 \sim b_j$ 的这些城市。

所有这 $n+m$ 座城市都无比崇拜丹，定义一次“膜拜”为你选择从某个国家的某座城市出发，沿着单向航线走，不重复地经过至少一座城市，最后回到起点的过程（简单来说就是走一个简单有向环）。为了展现你的虔诚，你希望在你的所有“膜拜”中经过国家 A 的第 i 座城市不超过 c_i 次，经过国家 B 的第 j 座城市不超过 d_j 次（**注意：在一次“膜拜”中起点和终点相同，但是认为起点只经过了一次**）。

现在你想知道你可以最多进行多少次“膜拜”。

【输入格式】

从文件 ***worship.in*** 中读入数据。

输入第一行两个整数 n, m 表示两个国家的城市数量。

接下来一行 n 个整数描述数组 a。

接下来一行 m 个整数描述数组 b。

接下来一行 n 个整数描述数组 c。

接下来一行 m 个整数描述数组 d。

【输出格式】

输出到文件 ***worship.out*** 中。

输出仅一个整数表示你可以进行的最多的膜拜次数。

【样例 1 输入】

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

【样例 1 输出】

```
1
```

【样例 1 解释】

用符号 A_i 表示国家 A 的第 i 座城市，B_j 表示国家 B 的第 j 座城市，这个样例中所有城市只能经过最多一次，不难发现最好情况下最多只能“膜拜”一次，一种“膜拜”方案为 $A_3 \to B_2 \to A_2 \to B_1 \to A_1 \to B_3 \to A_3$，经过所有点各一次。

【样例 2 输入】

```
2 1
1 1
2
1 1
2
```

【样例 2 输出】

```
2
```

【样例 2 解释】

用符号 A_i 表示国家 A 的第 i 座城市，B_j 表示国家 B 的第 j 座城市，这个样例中 A_1, A_2 只能经过最多一次，B_1 只能经过最多两次，不难发现最好情况下最多只能“膜拜”两次，一种“膜拜”方案为 $A_1 \to B_1 \to A_1, A_2 \to B_1 \to A_2$，经过 A_1, A_2 各一次，经过 B_1 两次。

【样例 3 】

见选手目录下的 ***worship/worship3.in*** 与 ***worship/worship3.ans***。

该样例满足测试点 $4 \sim 6$ 的性质。

【数据范围】

对于所有测试数据，满足 $1 \leq n, m \leq 5 \times 10^5, 0 \leq a_i \leq m, 0 \leq b_j \leq n, 1 \leq c_i, d_j \leq 10^9$。

每个测试点的具体限制见下表：

<table>
<tr><th>测试点编号</th><th>n</th><th>m</th><th>特殊性质</th></tr>
<tr><td>$1 \sim 3$</td><td>≤ 10</td><td>≤ 10</td><td rowspan="2">A</td></tr>
<tr><td>$4 \sim 6$</td><td>≤ 300</td><td>≤ 300</td></tr>
<tr><td>$7 \sim 10$</td><td rowspan="2">≤ 5000</td><td>≤ 5000</td><td rowspan="2">无</td></tr>
<tr><td>$11 \sim 13$</td><td rowspan="3">$\leq 5 \times 10^5$</td></tr>
<tr><td>$14 \sim 15$</td><td rowspan="2">$\leq 5 \times 10^5$</td><td>B</td></tr>
<tr><td>$16 \sim 20$</td><td>无</td></tr>
</table>

特殊性质 A：保证 $\forall i \in [1, n], c_i = 1$ 并且 $\forall j \in [1, m], d_j = 1$。

特殊性质 B：保证 $a_1 \leq a_2 \leq \cdots \leq a_n$。