

2026 年洛谷非专业级别收容能力认证

SCP-J/S 2026 第二轮

入门级

时间：2026 年 10 月 18 日 08:30 ~ 12:00

题目名称	打印	染色	进位	数组
题目类型	传统型	传统型	传统型	传统型
目录	print	color	carry	array
可执行文件名	print	color	carry	array
输入文件名	print.in	color.in	carry.in	array.in
输出文件名	print.out	color.out	carry.out	array.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 MiB	512 MiB	512 MiB	512 MiB
测试点数目	20	25	20	25
测试点是否等分	是	是	是	是

提交源程序文件名

对于 C++ 语言	print.cpp	color.cpp	carry.cpp	array.cpp
-----------	-----------	-----------	-----------	-----------

编译选项

对于 C++ 语言	-O2 -std=c++14 -static
-----------	------------------------

注意事项（请仔细阅读）

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. 提交的程序代码文件的放置位置请参考洛谷的具体要求。
4. 因违反以上三点而出现的错误或问题，申诉时一律不予受理。
5. 若无特殊说明，结果的比较方式为全文比较（过滤行末空格及文末回车）。
6. 选手提交的程序源文件必须不大于 100KB。
7. 程序可使用的栈空间内存限制与题目内存限制一致。
8. 只提供 Linux 格式附加样例文件。
9. 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此为准。
10. 在洛谷提交时，无需使用文件读写。

打印 (print)

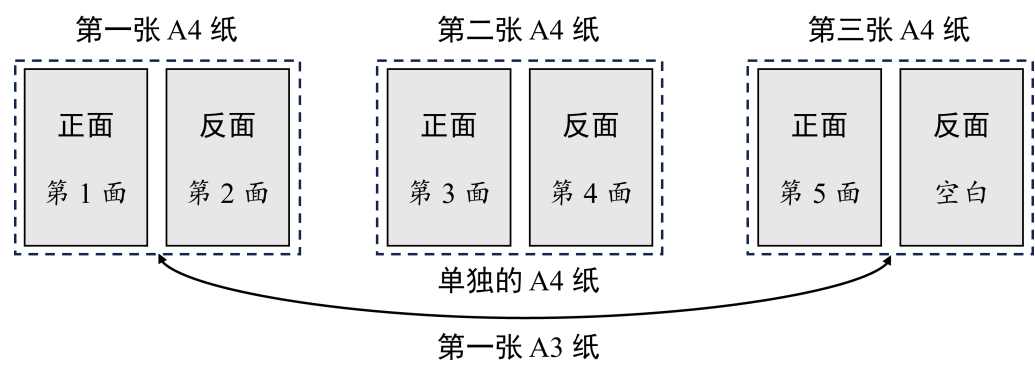
【题目描述】

小 W 要将一本有 n 面的书打印到 A4 纸上。小 W 想知道第 m 面最终会被打印到哪张纸上。

第 i 张 A4 纸的正面打印书的第 $2i - 1$ 面，反面打印书的第 $2i$ 面。若书没有第 $2i$ 面，则打印时第 $2i$ 面留空。

出版商在实际装订时使用的是 A3 纸。假设共使用了 k 张 A4 纸，第 1 张和第 k 张 A4 纸上所印的内容，会被分打印到第一张 A3 纸上；第 2 张和第 $k - 1$ 张 A4 纸上所印的内容，会被合并打印到第 2 张 A3 纸上……。如果最后剩余一张单独的 A4 纸未被合并打印，那么出版商会单独使用一张 A4 纸。

例如，当 $n = 5$ ，则将按如下顺序打印。



小 W 想要知道书第 m 面的内容实际会被打印到 A4 纸上，还是 A3 纸上。若打印到 A3 纸上，小 W 还想知道具体是哪一张。

【输入格式】

从文件 `print.in` 中读入数据。
输入一行两个正整数 n, m ，依次表示书的总面数和小 W 关心的面。

【输出格式】

输出到文件 `print.out` 中。
输出一行一个整数。
如果第 m 面的内容会被打印到单独的 A4 纸上，请输出 0。否则输出一个正整数 x ，表示第 m 面的内容会被打印到第 x 张 A3 纸上。

【样例 1 输入】

1

5 3

【样例 1 输出】

1

0

【样例 1 解释】

共需要 1 张 A3 纸和 1 张 A4 纸。第 1,2,5 面的内容将打印在 A3 纸上，第 3,4 面的内容将打印在单独的 A4 纸上。

【样例 2 输入】

1

9721133095981013 3091273763060069

【样例 2 输出】

1

1545636881530035

【样例 3 输入】

1

9365613187328233 5617672623161991

【样例 3 输出】

1

1873970282083122

【数据范围】

对于所有测试点，保证：

- $1 \leq m \leq n \leq 10^{16}$

测试点编号	$n \leq$
1 ~ 3	3
4 ~ 6	10
7 ~ 14	10^6
15 ~ 20	10^{16}

染色 (color)

【题目描述】

小 W 有一个长度为 n 的正整数数组 A ，从左至右的第 i 个数为 A_i 。
 A_i 将被染成红色或蓝色中的一种，不同的染色方案可能会导致不同的最终得分。设 $\{C_1, C_2, \dots, C_n\}$ 为分数数组：

- 如果 A_i 左侧没有与其同色的数，则 $C_i = 0$ 。
- 否则，记其左侧与其最靠近的同色数为 A_j ，则 $C_i = A_i - A_j$ 。

小 W 的染色得分为 $C_1 + C_2 + \dots + C_n$ ，请你求出染色得分的最大值。

【输入格式】

从文件 *color.in* 中读入数据。
输入共两行。
第一行包含一个正整数 n ，表示数组长度。
第二行包含 n 个正整数 $A_1, A_2, \dots, A_n$ 。

【输出格式】

输出到文件 *color.out* 中。
输出一行包含一个整数，表示染色得分的最大值。

【样例 1 输入】

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

【样例 1 输出】

```
1 1
```

【样例 1 解释】

一个最优的染色方案为 1 2 1，此时最终得分为 1。可以证明没有更优的染色方案。

【样例 2 输入】

```
1 6
2 3 4 9 8 2 4
```

【样例 2 输出】

```
1 7
```

【样例 2 解释】

一个最优的染色方案为 349824，此时最终得分为 7。可以证明没有更优的染色方案。

【样例 3】

见选手目录下的 *color/color3.in* 与 *color/color3.ans*。
该样例满足测试点 16 ~ 20 的限制。

【样例 4】

见选手目录下的 *color/color4.in* 与 *color/color4.ans*。
该样例满足测试点 21 ~ 25 的限制。

【数据范围】

对于所有测试点，保证：

- $2 \leq n \leq 2 \times 10^5$
- $1 \leq A_i \leq 3 \times 10^8$

测试点编号	$n \leq$	特殊性质
1 ~ 5	3	无
6 ~ 10	15	
11 ~ 15	1000	
16 ~ 20	2×10^5	有
21 ~ 25		无

特殊性质：对于所有 $1 \leq i \leq n$ ，满足 $\frac{A_i}{i}$ 为定值。

进位 (carry)

【题目描述】

给定一个长度为 n 的非负整数数组 $A_1, A_2, \dots, A_n$ 。

对于区间 $l \sim r$ ，可以执行以下操作：

- 令 $S = 0$ 。
- 按照 $A_l, A_{l+1}, \dots, A_r$ 的顺序，依次将 S 的值增加 A_i 。

所有加法均按照普通的二进制加法进行，在一次二进制加法中，如果某一位向更高一位产生了一个进位，就称其发生了一次“进位”。不同二进制位产生的进位分别计算，同一次加法中发生的多次连续进位也分别计算。用 $C(l, r)$ 表示区间 $l \sim r$ 执行上述操作过程中发生的“进位”总数。

例如计算 $3 + 5$ ： $3 = 011_2$ ， $5 = 101_2$ ，相加得 1000_2 。从低位到高位分别在第 0, 1, 2 位产生进位，共 3 次进位。若发生连续进位，例如 $7 + 1 = 111_2 + 1_2 = 1000_2$ ，从第 0 位到第 2 位连续产生 3 次进位，也分别计入。

若一个长度为 m 的数组 $[p_1, p_2, \dots, p_m]$ 满足以下条件，则称其合法：

- $m > 0$
- 对任意 $1 \leq i \leq m$ ，均满足 $1 \leq p_i \leq n$
- 对任意 $1 \leq i < m$ ，均满足 $p_i < p_{i+1}$ 且 $C(p_i, p_{i+1}) = 1$

两个数组不同，当且仅当两个数组长度不同，或至少一个位置的数字的不同。

求有多少个不同的合法数组，答案对 998,244,353 取模。

【输入格式】

从文件 `carry.in` 中读入数据。

第一行输入一个整数 T ，表示测试数据组数。

对于每组测试数据，输入两行：

- 第一行输入一个整数 n 。
- 第二行输入 n 个非负整数 $A_1, A_2, \dots, A_n$ 。

【输出格式】

输出到文件 `carry.out` 中。

对于每组测试数据，输出一行一个非负整数，表示合法数组数量对 998,244,353 取模后的结果。

【样例 1 输入】

```
1 3
2 5
3 1 4 4 3 8
4 6
5 6 3 7 4 6 6
6 5
7 52843137796 640771484120 983154867810 489455928042 397256148081
```

【样例 1 输出】

```
1 8
2 9
3 5
```

【样例 1 解释】

第一组数据合法下标序列为 [1], [2], [3], [4], [5], [1, 3], [2, 3], [2, 4], 共 8 个。

第二组数据合法下标序列为 [1], [2], [3], [4], [5], [6], [3, 4], [4, 5], [3, 4, 5], 共 9 个。

第三组数据合法下标序列为 [1], [2], [3], [4], [5], 共 5 个。

【样例 2】

见选手目录下的 *carry/carry2.in* 与 *carry/carry2.ans*。

该样例满足测试点 3 ~ 5 的限制。

【样例 3】

见选手目录下的 *carry/carry3.in* 与 *carry/carry3.ans*。

该样例满足测试点 6 ~ 8 的限制。

【样例 4】

见选手目录下的 *carry/carry4.in* 与 *carry/carry4.ans*。

该样例满足测试点 9 ~ 14 的限制。

【样例 5】

见选手目录下的 *carry/carry5.in* 与 *carry/carry5.ans*。

该样例满足测试点 15 ~ 16 的限制。

【样例 6】

见选手目录下的 `carry/carry6.in` 与 `carry/carry6.ans`。

该样例满足测试点 17 ~ 20 的限制。

【数据范围】

对于所有测试点，保证：

- $1 \leq T \leq 30$
- $1 \leq n \leq 5 \times 10^5$
- $0 \leq A_i \leq 10^{12}$
- $\sum n \leq 6 \times 10^6$

其中， $\sum n$ 表示一个输入文件内所有测试数据中 n 的总和。

测试点编号	$n \leq$	$\sum n \leq$
1, 2	10	300
3 ~ 5	2×10^3	3×10^3
6 ~ 8	5×10^3	10^4
9 ~ 14	10^5	5×10^5
15 ~ 16	2×10^5	2×10^6
17 ~ 20	5×10^5	6×10^6

【提示】

在本题中，你可能需要快速求出一个整数在二进制表示下有多少位为 1。你可以使用如下内建函数：

- `__builtin_popcount(x)`：返回 x 的二进制表示中 1 的个数，参数 x 通常为 `int` 或 `unsigned int` 类型。
- `__builtin_popcountll(x)`：返回 x 的二进制表示中 1 的个数，参数 x 通常为 `long long` 或 `unsigned long long` 类型。

二者的时间复杂度均为 $O(1)$ 。

注意：本题输入量较大，建议选手使用较快的读入方式，避免因输入耗时影响程序效率。

数组 (array)

【题目描述】

若数组 a, S 长度均为 n ，且对于所有 $1 \leq i \leq n$ ，均满足 $S_i = a_1 + a_2 + \cdots + a_i$ ，则数组 S 为数组 a 的前缀和数组。

小 D 有 n 个数组 $A_0, A_1, \cdots, A_{n-1}$ ，长度均为 n 。用 $A_{i,j}$ 表示数组 A_i 的第 j 项。已知数列 A_0 的初始值，对于所有 $1 \leq i < n$ ，数组 A_i 为数组 A_{i-1} 的前缀和数组。

现在小 D 有 p 个修改操作和 q 个查询操作：

- 对于修改操作，给出 x, l, r, k 。将数组 A_x 的第 $l \sim r$ 项均增加 k 。增加后，重新计算除了数组 A_x 外的所有数组，使得对所有 $1 \leq i < n$ ，满足数组 A_i 为数组 A_{i-1} 的前缀和数组。
- 对于查询操作，给出 x, l, r 。请求出数组 A_x 中第 $l \sim r$ 项之和。结果对 $10^9 + 7$ 取模。

请你执行这些操作，并输出所有查询操作的结果。

【输入格式】

从文件 `array.in` 中读入数据。

输入的第一行包含三个整数 n, p, q ，分别表示数组数量，修改操作数量和询问操作数量。

输入的第二行包含 n 个整数 $A_{0,1}, A_{0,2}, \dots, A_{0,n}$ ，表示初始的数列 A_0 。

接下来 $p + q$ 行，每行输入第一个数字 o 表示操作种类：

- 若 $o = 1$ ，则为修改操作。接下来输入四个数字 x, l, r, k ，表示将数组 A_x 的第 $l \sim r$ 项增加 k 。
- 若 $o = 2$ ，则为查询操作。接下来输入三个数字 x, l, r ，表示查询数组 A_x 中第 $l \sim r$ 项之和。

【输出格式】

输出到文件 `array.out` 中。

输出 q 行，一行一个非负整数，表示查询操作的结果。结果对 $10^9 + 7$ 取模。

【样例 1 输入】

```
1 10 2 3
2 1 2 3 4 5 6 7 8 9 10
3 1 0 2 4 1
4 2 0 2 4
```

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

【样例 1 输出】

```
1 12
2 25
3 378
```

【样例 1 解释】

第一次修改后：

- 原数组 A_0 为 $[1, 3, 4, 5, 5, 6, 7, 8, 9, 10]$;
- A_1 为 $[1, 4, 8, 13, 18, 24, 31, 39, 48, 58]$;
- 查询的答案分别为 $3 + 4 + 5 = 12$ 和 $4 + 8 + 13 = 25$ 。

第二次修改后：

- A_1 为 $[1, 4, 7, 12, 17, 24, 31, 39, 48, 58]$;
- A_2 为 $[1, 5, 12, 24, 41, 65, 96, 135, 183, 241]$;
- 第 2 项到第 8 项的和为 $5 + 12 + 24 + 41 + 65 + 96 + 135 = 378$ 。

【样例 2】

见选手目录下的 *array/array2.in* 与 *array/array2.ans*。
该样例满足测试点 4 ~ 7 的限制。

【样例 3】

见选手目录下的 *array/array3.in* 与 *array/array3.ans*。
该样例满足测试点 8 ~ 10 的限制。

【样例 4】

见选手目录下的 *array/array4.in* 与 *array/array4.ans*。
该样例满足测试点 22 ~ 25 的限制。

【数据范围】

对于所有测试点，保证：

- $1 \leq n \leq 300$
- $0 \leq p, q \leq 10^5$
- 对于所有 $1 \leq i \leq n$ ，均有 $|A_{0,i}| \leq 10^9$
- $o \in \{1, 2\}$
- 对于所有操作，均有 $0 \leq x < n$ ， $1 \leq l \leq r \leq n$
- $|k| \leq 10^9$

测试点编号	$n \leq$	特殊性质
1 ~ 3	10	$p, q \leq 10$
4 ~ 7	50	无
8 ~ 10	300	$x = 0$
11, 12		$p = 0$
13, 14		$q = 1$
15, 16		询问中的 $x = 0$
17, 18		修改中的 $x = 0$
19 ~ 21		$p, q \leq 3000$
22 ~ 25		无