全国青少年信息学奥林匹克联赛

NOIP 2024 模拟赛

时间： 2024 年 10 月 5 日 8:00 ~ 12:00

题目名称	花束插入	最小生成树	括号序列	奶牛吃草
题目类型	传统型	传统型	传统型	传统型
目录	`insert`	`tree`	`bracket`	`cow`
可执行文件名	`insert`	`tree`	`bracket`	`cow`
输入文件名	`insert.in`	`tree.in`	`bracket.in`	`cow.in`
输出文件名	`insert.out`	`tree.out`	`bracket.out`	`cow.out`
每个测试点时限	1.5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	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	`insert.cpp`	`tree.cpp`	`bracket.cpp`	`cow.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++ 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. 因违反以上两点而出现的错误或问题，申诉时一律不予受理。
4. 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）。
5. 选手提交的程序源文件必须不大于 100KB。
6. 程序可使用的栈空间内存限制与题目的内存限制一致。
7. 只提供 Windows 格式附加样例文件。
8. 禁止在源代码中改变编译器参数（如使用 pragma 命令），禁止使用系统结构相关指令（如内联汇编）和其他可能造成不公平的方法。
9. 评测时采用的机器配置为：Inter(R) Core(TM) Ultra 9 185H 2.30GHz，内存32GB。
10. 评测在 windows 下 lemon.v0.3.5 进行。

花束插入(insert)

【题目描述】

道路旁插满了鲜花。n 支花束在路边排成一排，每只花束有一个美丽值，美丽值越大的花束越能吸引人的目光。Alice 下班从路边走过时每次会看到相邻的 m 支花束，并被其中美丽值最大的花束吸引，产生这支花束美丽值的愉悦值。Alice 的家在道路另一端，他会走过整条道路，每次看到两个不同的花束区间时，愉悦值会累加。设第 i 支花束的美丽值为 a_i ，则 Alice 走过整条道路后产生的愉悦值为

$$\sum_{i=1}^{n-m+1} \max_{i \le j \le i+m-1} a_j$$

你手上有另一支美丽值为 A 的花束。你可以把他插入到任意两只相邻花束的中间，或道路的两端。在花束插入的若干种方案中，Alice 走过道路后能获得的愉悦值最大是多少？

【输入格式】

第一行输入三个正整数 n, m, A 分别表示道路原来的花束个数，Alice 一次看到的花束个数，自己手上花束的美丽值。

第二行输入 n 个正整数表示序列 a_i ，为原来道路旁第 i 支花束的美丽值。

【输出格式】

输出一行一个数，表示将手上的花束插入后，Alice 获得的愉悦值的最大值。

【样例 1 输入】

```
3 2 50
60 100 70
```

【样例 1 输出】

```
270
```

【样例 1 解释】

当花束插入在第一个位置时（$50, 60, 100, 70$），Alice 的愉悦值为 $\max\{60, 50\} + \max\{60, 100\} + \max\{100, 70\}$，即 $60 + 100 + 100 = 260$。

当花束插入在第二个位置时（$60, 50, 100, 70$），Alice 的愉悦值为 $\max\{60, 50\} + \max\{50, 100\} + \max\{100, 70\}$，即 $60 + 100 + 100 = 260$。

当花束插入在第三个位置时（$60, 100, 50, 70$），Alice 的愉悦值为 $\max\{60, 100\} + \max\{100, 50\} + \max\{50, 70\}$，即 $100 + 100 + 70 = 270$。

当花束插入在第四个位置时（$60, 100, 70, 50$），Alice 的愉悦值为 $\max\{60, 100\} + \max\{100, 70\} + \max\{70, 50\}$，即 $70 + 100 + 100 = 270$。

于是 Alice 的愉悦值的最大值为 $\max\{260, 260, 270, 270\} = 270$。

【样例2】

见选手目录下的 ***insert/insert2.in*** 与 ***insert/insert2.ans***。

该样例满足测试点 $1, 2$ 的限制。

【样例3】

见选手目录下的 ***insert/insert3.in*** 与 ***insert/insert3.ans***。

该样例满足测试点 $3, 4$ 的限制。

【样例4】

见选手目录下的 ***insert/insert4.in*** 与 ***insert/insert4.ans***。

该样例满足测试点 $5, 6$ 的限制。

【样例5】

见选手目录下的 ***insert/insert5.in*** 与 ***insert/insert5.ans***。

该样例满足测试点 $7 \sim 9$ 的限制。

【样例6】

见选手目录下的 ***insert/insert6.in*** 与 ***insert/insert6.ans***。

该样例满足测试点 $10, 11$ 的限制。

【样例7】

见选手目录下的 ***insert/insert7.in*** 与 ***insert/insert7.ans***。

该样例满足测试点 $12 \sim 14$ 的限制。

【数据范围】

对于所有数据，$1 \le m \le n \le 5 \times 10^6, 1 \le a_i, A \le 10^9$。

测试点编号	$n \le$	特殊性质
$1,2$	5000	无
$3 \sim 4$	5×10^6	A
$5 \sim 6$	5×10^6	B
$7 \sim 9$	5×10^6	C
$10 \sim 11$	5×10^5	D
$12 \sim 14$	5×10^5	无
$15 \sim 20$	5×10^6	无

特殊性质 A：$A \ge \max a_i$。

特殊性质 B：$A \le \min a_i$。

特殊性质 C：a 序列单调不降或单调不升。

特殊性质 D : $a_i \le 10$。

最小生成树(tree)

【题目描述】

有一个 n 个点 m 条边的无向图，第 i 条边的边权为 i。

以下是一个错误的最小生成树算法：

Algorithm 1 minimum spanning tree

```
vis := an array of length n
s := a set of edges

function dfs(u):
    vis[u] := true
    iterate through each edge (u, v) in the order from smallest to largest edge weight
        if vis[v] = false
            add edge (u, v) into the set (s)
            dfs(v)

function findMST(u):
    reset all elements of (vis) to false
    reset the edge set (s) to empty
    dfs(u)
    return the edge set (s)
```

每次运行 $findMST(1), findMST(2), \cdots, findMST(n)$ 都给出了一棵生成树。判断哪些生成树是最小生成树。

【输入格式】

每个测试点内有多组数据。

第一行一个整数 T 表示数据组数。

对于每组数据，第一行两个整数 n, m 表示图的点数和边数。

接下来 m 行，第 i 行输入两个整数 u_i, v_i，表示图中有一条 (u_i, v_i) 的无向边，边权为 i 。

【输出格式】

每组数据输出一行长为 n 的 0/1 串 s ，当 $findMST(i)$ 为最小生成树时 $s_i = 1$，否则 $s_i = 0$。

【样例 1 输入】

```
2
5 5
1 2
3 5
1 3
3 2
4 2
10 11
1 2
2 5
3 4
4 2
8 1
4 5
10 5
9 5
8 2
5 7
4 6
```

【样例 1 输出】

```
01111
0011111011
```

【样例2】

见选手目录下的 $tree/tree2.in$ 与 $tree/tree2.ans$。

【数据范围】

对于所有数据，$n \le 2 \times 10^5, \sum n \le 6 \times 10^5, \sum m \le 10^6$。保证图中不存在重边及自环。保证图联通。

数据点编号	$n \le$	$\sum n \le$	$\sum m \le$	特殊性质
$1, 2$	5000	20000	10^5	无
$3 \sim 5$	2×10^5	6×10^5	6×10^5	A
$6 \sim 8$	2×10^5	6×10^5	10^6	B
$9 \sim 11$	2×10^5	6×10^5	10^6	C
12	2×10^5	6×10^5	10^6	D
$13 \sim 18$	5×10^4	2×10^5	5×10^5	无
$19, 20$	2×10^5	6×10^5	10^6	无

特殊性质 A：$n = m$。

特殊性质 B：最小生成树中每个点度数不超过 2。

特殊性质 C：最小生成树中存在一个点度数为 $n - 1$。

特殊性质 D：在保证图联通的基础上每条边随机生成。

括号序列(bracket)

【题目描述】

Alice 有一个长度为 n 的括号序列 s。

对于一个括号序列 S，Alice 可以执行两种操作：

1. 变换：选择一个位置 i 满足 $1 \le i \le |S|$，将 S 变为 $S_iS_{i+1}\cdots S_{|S|}S_1S_2\cdots S_{i-2}S_{i-1}$。

2. 插入：在这个序列的任意位置插入一个括号（左右括号均可）。

Alice 定义括号序列 S 的权值 $f(S)$ 为能将这个括号序列变成一个合法括号序列所需的最小操作数。

其中，合法括号序列的定义为：

1. 空串为合法括号序列。
2. 若 $\underline{A}$ 为合法括号序列，则 $\underline{(A)}$ 为合法括号序列。
3. 若 $\underline{A},\underline{B}$ 均为合法括号序列，则 $\underline{AB}$ 也为合法括号序列。

现在 Alice 想要求出：

$$\sum_{l=1}^{n}\sum_{r=l}^{n} f(s[l,r])$$

其中 $s[l,r]$ 表示由 $s_l, s_{l+1}, \cdots, s_r$ 形成的连续子序列。

但是 Alice 还在工厂上班检查零件。请你帮他算出答案。

【输入格式】

每个测试点内有多组数据。

第一行一个正整数 T 表示测试数据组数。

对于每组数据，第一行一个正整数 n。

第二行一个长度为 n 的括号序列 s。

【输出格式】

输出共 T 行，第 i 行一个整数表示第 i 组测试数据的答案。

【样例 1 输入】

```
5
2
((
4
```

```
())(
5
()(()
5
()()(
15
()())(()))()()
```

【样例 1 输出】

```
4
11
16
12
241
```

【样例 1 解释】

对于 $s = \underline{())(}$ 来说：

考虑 $s[1,4] = \underline{())(}$。执行变换操作 $i = 4$，有 $\underline{())(} \Rightarrow \underline{(())}$，其中 $\underline{(())}$ 是合法括号序列，故 $f(s[1,4]) = 1$。可以证明不存在更优的策略。

考虑 $s[2,4] = \underline{))(}$。执行变换操作 $i = 2$，再在序列开头插入一个左括号，有 $\underline{))(} \Rightarrow \underline{)()} \Rightarrow \underline{()()}$，其中 $\underline{()()}$ 是合法括号序列，故 $f(s[2,4]) = 2$。可以证明不存在更优的策略。

【样例2】

见选手目录下的 ***bracket/bracket2.in*** 与 ***bracket/bracket2.ans***。

该样例满足测试点 $1, 2$ 的限制。

【样例3】

见选手目录下的 ***bracket/bracket3.in*** 与 ***bracket/bracket3.ans***。

该样例满足测试点 6 的限制。

【样例4】

见选手目录下的 ***bracket/bracket4.in*** 与 ***bracket/bracket4.ans***。

该样例满足测试点 $7, 8$ 的限制。

【样例5】

见选手目录下的 ***bracket/bracket5.in*** 与 ***bracket/bracket5.ans***。
该样例满足测试点 9 的限制。

【样例6】

见选手目录下的 ***bracket/bracket6.in*** 与 ***bracket/bracket6.ans***。
该样例满足测试点 10, 11 的限制。

【样例7】

见选手目录下的 ***bracket/bracket7.in*** 与 ***bracket/bracket7.ans***。
该样例满足测试点 $12 \sim 16$ 的限制。

【数据范围】

对于全部数据，$n \leq 2 \times 10^6, \sum n \leq 2 \times 10^7$。

数据点编号	$n \leq$	$\sum n \leq$	特殊性质
$1, 2$	500	2000	无
$3, 4, 5$	5000	20000	无
6	2×10^6	2×10^7	A
$7, 8$	2×10^6	2×10^7	B
9	2×10^6	2×10^7	C
$10, 11$	2×10^5	10^6	D
$12 \sim 16$	2×10^5	10^6	无
$17 \sim 20$	2×10^6	2×10^7	无

特殊性质 A：s 内不含有右括号。
特殊性质 B：对于所有整数 $1 \leq i < n$， $s_i \neq s_{i+1}$。
特殊性质 C：存在 $1 \leq k < n$，满足 $s_1 = s_2 = \cdots = s_k = (, s_{k+1} = \cdots = s_n =)$。
特殊性质 D：s 内左括号数不超过 10。

奶牛吃草(cow)

【题目描述】

Bessie 是一头很饿的奶牛。每天晚上，如果还有干草他就会吃一份，没有就不吃。初始时没有干草。

每天早上 Farmer John 会给他送干草。具体的，他会在第 k 天送 a_k 份干草。初始时 $a_k = 0$ ，表示该天不送干草。q 次操作，每次操作给出 (x, y) ，表示将第 x 天送的干草数量改成 y ，即修改 $a_x = y$。每次操作后，请将 Bessie 有干草吃的日子的编号求和并输出。对 $10^9 + 7$ 取模。操作间互不独立。

【输入格式】

第一行一个整数 q，表示操作数量。

接下来 q 行每行两个整数 x, y ，表示将第 x 天送的干草数量改为 y。

【输出格式】

输出 q 行，每行一个数表示答案。

【样例 1 输入】

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

【样例 1 输出】

```
15
36
18
```

【样例 1 解释】

每次更新后对应的答案：

$4 + 5 + 6 = 15$

$1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 = 36$

$1 + 2 + 4 + 5 + 6 = 18$

【样例 2 输入】

```
9
1 89
30 7
101 26
1 24
5 1
60 4
5 10
101 0
1 200
```

【样例 2 输出】

```
4005
4656
7607
3482
3507
3753
4058
1107
24531
```

【样例3】

见选手目录下的 $cow/cow3.in$ 与 $cow/cow3.ans$。
该样例满足测试点 $1, 2$ 的限制。

【样例4】

见选手目录下的 $cow/cow4.in$ 与 $cow/cow4.ans$。
该样例满足测试点 $3 \sim 6$ 的限制。

【样例5】

见选手目录下的 ***cow/cow5.in*** 与 ***cow/cow5.ans***。
该样例满足测试点 $7 \sim 9$ 的限制。

【样例6】

见选手目录下的 ***cow/cow6.in*** 与 ***cow/cow6.ans***。
该样例满足测试点 $10, 11$ 的限制。

【数据范围】

对于全部数据，$1 \le q \le 3 \times 10^5, 1 \le x \le 10^{14}, 1 \le y \le 10^9$。

数据点编号	$q \le$	特殊性质
$1, 2$	5000	无
$3 \sim 6$	10^5	A
$7 \sim 9$	10^5	B
$10, 11$	10^5	C
$12 \sim 17$	10^5	无
$18 \sim 20$	3×10^5	无

特殊性质 A：所有修改只会增加某一天送的干草数。
特殊性质 B：$1 \le x \le 10^5$。
特殊性质 C：$1 \le y \le 10$。