数字游戏

(number.cpp/c/pas)

【问题描述】

小 K 同学向小P 同学发送了一个长度为 8 的 **01 字符串**来玩数字游戏，小P 同学想要知道字符串中究竟有多少个 1。

注意：01 字符串为每一个字符是 0 或者 1 的字符串，如“101”（不含双引号）为一个长度为 3 的 01 字符串。

【输入格式】

输入文件名为number.in。

输入文件只有一行，一个长度为 8 的 01 字符串 s。

【输出格式】

输出文件名为number.out。

输出文件只有一行，包含一个整数，即 01 字符串中**字符 1** 的个数。

【输入输出样例 1】

number.in	**number.out**
00010100	2

见选手目录下的 number/number1.in和number/number1.ans。

【输入输出样例 1 说明】

该 01 字符串中有 2 个字符 1。

【输入输出样例 2】

title.in	**title.out**
11111111	8

见选手目录下的 number/number2.in和number/number2.ans。

【输入输出样例 2 说明】

该 01 字符串中有 8 个字符 1。

【输入输出样例 3】

见选手目录下的 number/number3.in和number/number3.ans。

【数据规模与约定】

对于 20% 的数据，保证输入的字符全部为 0。

对于 100% 的数据，输入只可能包含字符 0 和字符 1，字符串长度固定为 8。

公交换乘

(transfer.cpp/c/pas)

【问题描述】

著名旅游城市B 市为了鼓励大家采用公共交通方式出行，推出了一种地铁换乘公交车的优惠方案:

1. 在搭乘一次地铁后可以获得一张优惠票，有效期为 45 分钟，在有效期内可以消耗这张优惠票，免费搭乘一次票价不超过地铁票价的公交车。在有效期内指开始乘公交车的时间与开始乘地铁的时间之差小于等于 45 分钟，即:

$$t_{bus} - t_{subway} \leq 45$$

2. 搭乘地铁获得的优惠票可以累积，即可以连续搭乘若干次地铁后再连续使用优惠票搭乘公交车。
3. 搭乘公交车时，如果可以使用优惠票一定会使用优惠票；如果有多张优惠票满足条件，则优先消耗获得最早的优惠票。

现在你得到了小轩最近的公共交通出行记录，你能帮他算算他的花费吗?

【输入格式】

输入文件名为 transfer.in。

输入文件的第一行包含一个正整数 ，代表乘车记录的数量。

接下来的 行，每行包含 3 个整数，相邻两数之间以一个空格分隔。第 行的第 1 个整数代表第 条记录乘坐的交通工具，0 代表地铁，1 代表公交车；第 2 个整数代表第 条记录乘车的票价 ；第三个整数代表第 条记录开始乘车的时间 (距 0 时刻的分钟数)。

我们保证出行记录是按照开始乘车的时间顺序给出的，且不会有两次乘车记录出现在同一分钟。

【输出格式】

输出文件名为 transfer.out。

输出文件有一行，包含一个正整数，代表小轩出行的总花费

【输入输出样例 1】

transfer.in	transfer.out
6 0 10 3 1 5 46 0 12 50 1 3 96 0 5 110 1 6 135	36

见选手目录下的 transfer/transfer1.in和transfer/transfer1.ans。

【输入输出样例 1 说明】

第一条记录，在第 3 分钟花费 10 元乘坐地铁。

第二条记录，在第 46 分钟乘坐公交车，可以使用第一条记录中乘坐地铁获得的优惠票，因此没有花费。

第三条记录，在第 50 分种花费 12 元乘坐地铁。

第四条记录，在第 96 分钟乘坐公交车，由于距离第三条记录中乘坐地铁已超过 45 分钟，所以优惠票已失效，花费 3 元乘坐公交车。

第五条记录，在第 110 分钟花费 5 元乘坐地铁。

第六条记录，在第 135 分钟乘坐公交车，由于此时手中只有第五条记录中乘坐地铁获得的优惠票有效，而本次公交车的票价为 6 元，高于第五条记录中地铁的票价 5 元，所以不能使用优惠票，花费 6 元乘坐公交车。

总共花费 36 元。

【输入输出样例 2】

transfer.in	**transfer.out**
6 0 5 1 0 20 16 0 7 23 1 18 31 1 4 38 1 7 68	32

见选手目录下的 transfer/transfer2.in和transfer/transfer2.ans。

【输入输出样例 2 说明】

第一条记录，在第 1 分钟花费 5 元乘坐地铁。

第二条记录，在第 16 分钟花费 20 元乘坐地铁。

第三条记录，在第 23 分钟花费 7 元乘坐地铁。

第四条记录，在第 31 分钟乘坐公交车，此时只有第二条记录中乘坐的地铁票价高于本次公交车票价，所以使用第二条记录中乘坐地铁获得的优惠票。

第五条记录，在第 38 分钟乘坐公交车，此时第一条和第三条记录中乘坐地铁获得的优惠票都可以使用，使用获得最早的优惠票，即第一条记录中乘坐地铁获得的优惠票。

第六条记录，在第 68 分钟乘坐公交车，使用第三条记录中乘坐地铁获得的优惠票。

总共花费 32 元。

【输入输出样例 3】

见选手目录下的 transfer/transfer3.in和transfer/transfer3.ans。

【数据规模与约定】

对于 30% 的数据，$n \leq 1{,}000$，$t_i \leq 10^6$。

另有 15% 的数据，$t_i \leq 10^7$，$price_i$都相等。

另有 15% 的数据，$t_i \leq 10^9$，$price_i$都相等。

对于 100% 的数据，$n \leq 10^5$，$t_i \leq 10^9$，$1 \leq price_i \leq 1{,}000$。

标题统计

(title.cpp/c/pas)

【问题描述】

凯凯刚写了一篇美妙的作文，请问这篇作文的标题中有多少个字符?

注意：标题中可能包含大、小写英文字母、数字字符、空格和换行符。统计标题字符数时，空格和换行符不计算在内。

【输入格式】

输入文件名为 title.in。

输入文件只有一行，一个字符串 s。

【输出格式】

输出文件名为 title.out。

输出文件只有一行，包含一个整数，即作文标题的字符数（不含空格和换行符）。

【输入输出样例 1】

title.in	title.out
234	3

见选手目录下的 title/title1.in 和 title/title1.ans。

【输入输出样例 1 说明】

标题中共有 3 个字符，这 3 个字符都是数字字符。

【输入输出样例 2】

title.in	title.out
Ca 45	4

见选手目录下的 title/title2.in 和 title/title2.ans。

【输入输出样例 2 说明】

标题中共有 5 个字符，包括1 个大写英文字母，1 个小写英文字母和 2 个数字字符，还有 1 个空格。由于空格不计入结果中，故标题的有效字符数为 4 个。

【数据规模与约定】

规定 |s| 表示字符串 s 的长度（即字符串中的字符和空格数）。

对于 40% 的数据，$1 \le |s| \le 5$，保证输入为数字字符及行末换行符。

对于 80% 的数据，$1 \le |s| \le 5$，输入只可能包含大、小写英文字母、数字字符及行末换行符。

对于 100% 的数据，$1 \le |s| \le 5$，输入可能包含大、小写英文字母、数字字符、空格和行末换行符。

龙虎斗

【问题描述】

轩轩和凯凯正在玩一款叫《龙虎斗》的游戏，游戏的棋盘是一条线段，线段上有 n 个兵营（自左至右编号 $1\sim n$），相邻编号的兵营之间相隔 1 厘米，即棋盘为长度为 $n-1$ 厘米的线段。i 号兵营里有 c_i 位工兵。

下面图 1 为 $n=6$ 的示例：

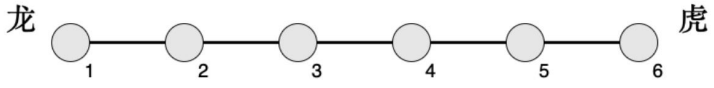

图 1. $n=6$的示例

轩轩在左侧，代表“龙”；凯凯在右侧，代表“虎”。 他们以 m 号兵营作为分界，靠左的工兵属于龙势力，靠右的工兵属于虎势力，而**第 m 号兵营中的工兵很纠结，他们不属于任何一方**。

一个兵营的气势为：该兵营中的工兵数 × 该兵营到 m 号兵营的距离；参与游戏一方的势力定义为：属于这一方所有兵营的气势之和。

下面图 2 为 $n=6, m=4$ 的示例，其中红色为龙方，黄色为虎方：

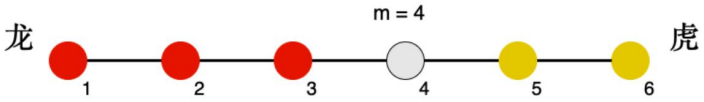

图 2. $n=6, m=4$的示例

游戏过程中，某一刻天降神兵，共有 s_1 位工兵突然出现在了 p_1 号兵营。作为轩轩和凯凯的朋友，你知道如果龙虎双方气势差距太悬殊，轩轩和凯凯就不愿意继续玩下去了。为了让游戏继续，你需要选择一个兵营 p_2，并将你手里的 s_2 位工兵**全部**派往兵营 p_2，使得双方气势差距尽可能小。

注意：你手中的工兵落在哪个兵营，就和该兵营中其他工兵有相同的势力归属（如果落在 m 号兵营，则不属于任何势力）。

【输入格式】

输入文件名为 `fight.in`。

输入文件的第一行包含一个正整数 n，代表兵营的数量。

接下来的一行包含 n 个正整数，相邻两数之间以一个空格分隔，第 i 个正整数代表编号为 i 的兵营中起始时的工兵数量 c_i。

接下来的一行包含四个正整数，相邻两数间以一个空格分隔，分别代表 m, p_1, s_1, s_2。

【输出格式】

输出文件名为 `fight.out`。

输出文件有一行，包含一个正整数，即 p_2，表示你选择的兵营编号。如果存在多个编号同时满足最优，取最小的编号。

【输入输出样例 1】

fight.in	**fight.out**
6 2 3 2 3 2 3 4 6 5 2	2

见选手目录下的 fight/fight1.in 和 fight/fight1.ans。

【输入输出样例 1 说明】

见问题描述中的图 2。

双方以 $m=4$ 号兵营分界，有 $s_1=5$ 位工兵突然出现在 $p_1=6$ 号兵营。

龙方的气势为：

$$2\times(4-1)+3\times(4-2)+2\times(4-3)=14$$

虎方的气势为：

$$2\times(5-4)+(3+5)\times(6-4)=18$$

当你将手中的 $s_2=2$ 位工兵派往 $p_2=2$ 号兵营时，龙方的气势变为：

$$14+2\times(4-2)=18$$

此时双方气势相等。

【输入输出样例 2】

fight.in	**fight.out**
6 1 1 1 1 1 16 5 4 1 1	1

见选手目录下的 fight/fight2.in 和 fight/fight2.ans。

【输入输出样例 2 说明】

双方以 $m=5$ 号兵营分界，有 $s_1=1$ 位工兵突然出现在 $p_1=4$ 号兵营。

龙方的气势为：

$$1\times(5-1)+1\times(5-2)+1\times(5-3)+(1+1)\times(5-4)=11$$

虎方的气势为：

$$16\times(6-5)=16$$

当你将手中的 $s_2=1$ 位工兵派往 $p_2=1$ 号兵营时，龙方的气势变为：

$$11+1\times(5-1)=15$$

此时可以使双方气势的差距最小。

【数据规模与约定】

$1<m<n, 1\le p_1\le n$。

对于 20% 的数据，$n=3,\ m=2,\ c_i=1,\ s_1,s_2\le 100$。

另有 20% 的数据，$n\le 10,\ p_1=m,\ c_i=1,\ s_1,s_2\le 100$。

对于 60% 的数据，$n\le 100,\ c_i=1,\ s_1,s_2\le 100$。

对于 80% 的数据，$n\le 100,\ c_i,s_1,s_2\le 100$。

对于 100% 的数据，$n\le 10^5,\ c_i,s_1,s_2\le 10^9$。

成绩

(score.cpp/c/pas)

【问题描述】

牛牛最近学习了 C++入门课程，这门课程的总成绩计算方法是：

总成绩 = 作业成绩× 20% + 小测成绩× 30% + 期末考试成绩× 50%

牛牛想知道，这门课程自己最终能得到多少分。

【输入格式】

输入文件名为 score.in。

输入文件只有 1 行，包含三个非负整数A、B、C，分别表示牛牛的作业成绩、小测成绩和期末考试成绩。相邻两个数之间用一个空格隔开，三项成绩满分都是 100 分。

【输出格式】

输出文件名为 score.out。

输出文件只有 1 行，包含一个整数，即牛牛这门课程的总成绩，满分也是 100 分。

【输入输出样例 1】

score.in	score.out
100 100 80	90

见选手目录下的 score/score1.in 和 score/score1.ans。

【输入输出样例 1 说明】

牛牛的作业成绩是 100 分，小测成绩是 100 分，期末考试成绩是 80 分，总成绩是 100 × 20% + 100 × 30% + 80 × 50% = 20 + 30 + 40 = 90。

【输入输出样例 2】

score.in	score.out
60 90 80	79

见选手目录下的 score/score2.in 和 score/score2.ans。

【输入输出样例 2 说明】

牛牛的作业成绩是 60 分，小测成绩是 90 分，期末考试成绩是 80 分，总成绩是 60 × 20% + 90 × 30% + 80 × 50% = 12 + 27 + 40 = 79 。

【数据说明】

对于 30% 的数据，A = B = 0。

对于另外 30% 的数据，A = B = 100。

对于 100% 的数据， 0 ≤ A、B、C ≤ 100 且 A、B、C 都是 10 的整数倍。

图书管理员

(librarian.cpp/c/pas)

【问题描述】

图书馆中每本书都有一个图书编码，可以用于快速检索图书，这个图书编码是一个正整数。

每位借书的读者手中有一个需求码，这个需求码也是一个正整数。如果一本书的图书编码恰好以读者的需求码结尾，那么这本书就是这位读者所需要的。

小D 刚刚当上图书馆的管理员，她知道图书馆里所有书的图书编码，她请你帮她写一个程序，对于每一位读者，求出他所需要的书中图书编码最小的那本书，如果没有他需要的书，请输出-1。

【输入格式】

输入文件名为 librarian.in。

输入文件的第一行，包含两个正整数 n 和 q，以一个空格分开，分别代表图书馆里书的数量和读者的数量。

接下来的n 行，每行包含一个正整数，代表图书馆里某本书的图书编码。

接下来的q 行，每行包含两个正整数，以一个空格分开，第一个正整数代表图书馆里读者的需求码的长度，第二个正整数代表读者的需求码。

【输出格式】

输出文件名为 librarian.out。

输出文件有 q 行，每行包含一个整数，如果存在第 i 个读者所需要的书，则在第 i 行输出第i 个读者所需要的书中图书编码最小的那本书的图书编码，否则输出-1。

【输入输出样例 1】

librarian.in	librarian.out
5 5 2123 1123 23 24 24 2 23 3 123 3 124 2 12 2 12	23 1123 -1 -1 -1

见选手目录下的 librarian /librarian1.in和librarian/librarian1.ans。

【输入输出样例 1 说明】

第一位读者需要的书有 2123、1123、23，其中 23 是最小的图书编码。第二位读者需要的书有 2123、1123，其中 1123 是最小的图书编码。对于第三位，第四位和第五位读者，没有书的图书编码以他们的需求码结尾，即没有他们需要的书，输出-1。

【输入输出样例 2】

见选手目录下的 librarian/librarian2.in和 librarian/librarian2.ans。

【数据规模与约定】

对于 20%的数据，1 ≤ n ≤ 2。另有 20% 的数据，q = 1。

另有 20%的数据，所有读者的需求码的长度均为 1。

另有 20%的数据，所有的图书编码按从小到大的顺序给出。

对于 100%的数据，1 ≤ n ≤ 1,000，1 ≤ q ≤ 1,000，所有的图书编码和需求码均不超过 10,000,000。