模拟赛

whzzt

Jan 16, 2019

题目名称	扫雷	矩阵	字符串
输入文件名	N/A	N/A	N/A
输出文件名	N/A	N/A	N/A
时间限制	1s	3s	10s
是否捆绑测试	否	否	否
内存限制	1GB	1GB	1GB
是否有部分分	是	是	是
题目类型	交互	交互	交互
编译开关	-O2 -std=c++11	-O2 -std=c++11	-O2 -std=c++11

注意：1. 评测时的栈空间大小不做单独限制，但使用的总空间大小不能超过内存限制。

2. 考试之中不要交流，AK 之后不要大声喧哗，没有 AK 的同学请安静做题。

3. 评测环境 Intel(R) Core(TM) i7-8650U CPU @1.90GHz 2.11GHz

4. 不要倦生，不要高喊 “这题太难我不会” 之类的话。

5. 代码长度限制为 64 KB。

Problem A. 扫雷 (mine.cpp)

Input file:	N/A
Output file:	N/A
Time limit:	1 seconds
Memory limit:	1 gigabytes

这是一道交互题。

老虎和蒜头是好朋友。

老虎最近沉迷上了一款名叫扫雷的游戏。为了能随时随地玩扫雷，老虎高价购买了一款扫雷机。而蒜头认为这是没有必要的，为了说服老虎，你需要帮助蒜头设计一个扫雷程序。

扫雷是在一个 $W \times H$ 的网格上进行的游戏，其中位置的下标从 0 开始，每个格子可能是雷或者空地。定义每个格子的状态有两种：开或关。初始所有格子都是关闭的。

你每次可以将一个关闭的格子打开，如果这个格子是雷，你就输掉了这场游戏，否则这个格子会显示一个在区间 $[0, 8]$ 内的整数，表示有多少个雷与这个格子相邻。两个格子相邻当且仅当它们有一条公共边或者一个公共点。如果所有空地都已经处于开的状态，那么你就赢得了这场游戏。

当然，有时候凭借已知信息无法判断哪个位置有雷哪个位置没有雷，那么你就只能猜了。为了你的游戏体验，蒜头允许你犯错，但你的得分与你踩雷的数量有关。

老虎的扫雷机生成地图的方式是随机的。具体地，扫雷机会获取四个参数：地图的大小 W, H，雷的数量 K 和随机数种子 S。

我们给出地图的生成方式，定义序列 T 满足：

$$T_{i+1} \equiv 48271 \times T_i \pmod{2147483647}$$

其中 $T_0 = S$，然后从下标 1 开始枚举，对于一个 T_i，其对应的雷位置为第 r_i 行，c_i 列，其中：

$$r_i = \left\lfloor \frac{T_i}{H} \right\rfloor \bmod W, c_i = T_i \bmod H$$

如果这里没有雷，则填上一个雷。当雷的数量为 K 时，退出循环，地图就造好了。

Interaction Protocol

你的任务是实现一个函数：

```
void sweep(int W, int H, int K)
```

其中 $W = 16$ 表示地图的行数，$H = 30$ 表示地图的列数，$100 \le K \le 200$ 表示地图上的地雷数目，我们并不会告知你 S，但已知有 $100 \le S \le 10000$。

你可以调用以下函数来表示操作：

```
int open(int r, int c)
```

表示打开 (r, c) 这个格子，当 (r, c) 是地雷时该函数返回 -1，否则返回相邻的格子中的地雷数目。你不能打开同一个格子两次。

当所有空地均被打开时，程序将直接结束，并根据你踩雷的次数计算你的得分。

你可以参考样例交互库对上述描述进行理解。

Notes

本题共有 10 组测试数据，记 T 表示这组数据中你踩雷的次数，你在这组数据中的得分 P 和 T 的关系为：

$T \in$	$P =$
$[0, 6]$	10
$(6, 10]$	$16 - T$
$(10, 20]$	$10 - 0.4S$
$(20, 50]$	1
$(50, \infty]$	0

本题交互库的运行时间不会超过 10ms, 占用空间大小不会超过 2MB，也就是说选手程序的可用时间至少为 990ms, 可用空间至少为 1022MB。

在下发文件中，我们提供了 mine.h, grader.cpp, mine.cpp 三个文件，你最终提交的代码应当包含在 mine.cpp 中，并包含头文件 mine.h。

如果你想要命令行编译你的代码，应当执行：g++ mine.cpp grader.cpp -o mine -O2 -std=c++11。

Problem B. 矩阵 (matrix.c/cpp/pas)

Input file:	N/A
Output file:	N/A
Time limit:	3 seconds
Memory limit:	1 gigabytes

这是一道交互题。

老虎和蒜头是好朋友。

老虎最近得到了一个 $N \times N$ 的 01 矩阵，他和蒜头计划用这个矩阵玩一个游戏：蒜头每次可以询问老虎一个矩阵是否是这个 $N \times N$ 的 01 矩阵的子矩阵，蒜头的目标是找到老虎得到的这个矩阵，你需要帮助蒜头在指定的操作次数内完成该任务。

Interaction Protocol

你的任务是实现一个函数：

```
void findMatrix(int N)
```

其中 N 是老虎的矩阵大小，你可以调用下面的函数进行询问：

```
bool isSubMatrix(std::vector<std::vector<char>> matrix)
```

你必须保证你传入的参数是一个矩阵。

你可以调用以下函数返回答案：

```
void foundMatrix(std::vector<std::vector<char>> matrix)
```

你必须保证你传入的参数是一个 N 行 N 列的矩阵，调用该函数后程序将会终止，该函数应当被调用恰好一次。

你可以参考样例交互库对上述描述进行理解。

Notes

本题共包括 20 个测试点，每个测试点的分值为 5 分，其中第 k 个测试点满足 $N = 3k$。

不妨假设你的程序调用函数 isSubMatrix 的次数为 T，若你得到的答案错误或是没有进行回答，那么你将得到零分。否则你的得分将取决于 $M = \frac{T}{N^2}$ 的大小，评分标准如下：

$M \in$	$P =$
$[0, 5]$	5
$(5, 10]$	$6 - 0.2M$
$(10, 30]$	$5 - 0.1M$
$(30, \infty]$	$3 - \frac{M}{30}$

我们保证交互库占用空间大小不会超过 128MB，也就是说选手程序的可用空间至少为 896MB。同时，我们保证你调用的函数的实现和下发的样例交互库中的实现相同。

在下发文件中，我们提供了 matrix.h, grader.cpp, matrix.cpp 三个文件，你最终提交的代码应当包含在 matrix.cpp 中，并包含头文件 matrix.h。

如果你想要命令行编译你的代码，应当执行：g++ matrix.cpp grader.cpp -o matrix -O2 -std=c++11。

Problem C. 字符串 (guess.cpp)

Input file:	N/A
Output file:	N/A
Time limit:	10 seconds
Memory limit:	1 gigabytes

这是一道交互题。

老虎和蒜头是好朋友。

老虎最近得到了一个长度为 1000 的 01 串，蒜头想要知道这个串，但老虎并不打算就这么简单地告诉蒜头真相，他计划和蒜头玩一个游戏。

Interaction Protocol

你的任务是实现一个函数：

```
std::string guess()
```

该函数应当返回老虎得到的 01 串。

你可以调用以下函数来向老虎询问该串的相关信息：

```
int query(int l, int r)
```

你必须保证 $0 \le l \le r < n$，该函数有 50% 的概率返回 $[l, r]$ 中数字 1 的个数 m，另 50% 的概率返回 $[0, m-1] \cup [m+1, r-l+1]$ 中的一个随机整数。**请注意：对一组 l, r 你只能询问至多一次。**

你可以参考样例交互库对上述描述进行理解。

Notes

本题共包括 1 个测试点，我们会对你的函数进行不超过 100 次调用，我们允许你出现不超过 2 次答案错误的情况，否则你将得到 0 分。否则，不妨设你最多一次调用了 T 次 query 函数，设你的得分为 P，评分标准如下：

$T \in$	$P =$
$[0, 6000]$	100
$(6000, 18000]$	$125 - 0.005T$
$(18000, 10^5]$	$35 - 0.0003T$
$(10^5, \infty]$	$5 - 0.00001T$

我们保证交互库占用的资源不会超过总资源的 10%。

在下发文件中，我们提供了 guess.h, grader.cpp, guess.cpp 三个文件，你最终提交的代码应当包含在 guess.cpp 中，并包含头文件 guess.h。

如果你想要命令行编译你的代码，应当执行：g++ guess.cpp grader.cpp -o guess -O2 -std=c++11。