NOIP 模拟赛

sunnuozhou

September 21, 2020

题目名称	生存	撤离	庇护
目录	survive	evacuate	refuge
可执行文件名	survive	evacuate	refuge
输入文件名	survive.in	evacuate.in	refuge.in
输出文件名	survive.out	evacuate.out	refuge.out
每个测试点时限	2.0s	1.5s	1.0s
内存限制	512MB	512MB	512MB
试题总分	100	100	100
测试点数目	5	10	7
每个测试点分值	N/A	10	N/A
是否有部分分	否	否	否
题目类型	传统型	传统型	传统型

提交的源程序文件名

对于 C++ 语言	survive.cpp	evacuate.cpp	refuge.cpp
对于 C 语言	survive.c	evacuate.c	refuge.c
对于 Pascal 语言	survive.pas	evacuate.pas	refuge.pas

编译开关

对于 C++ 语言	-O2 -std=c++11	-O2 -std=c++11	-O2 -std=c++11
对于 C 语言	-O2 -std=c11	-O2 -std=c11	-O2 -std=c11
对于 Pascal 语言	-O2	-O2	-O2

1 生存

1.1 题目背景

XXXX 年，严重的灾害爆发了。短短几个月，Z 国的人口就大幅下降。而你作为幸存者的领导，依靠着意外得来的灾害预测机，需要带领大家躲避灾害，维持幸存者的数量。

1.2 题目描述

Z 国有 n 个城市，通过 $n-1$ 条双向道路相连，任意两个城市都可以通过这些道路相互到达。

在第 1 天开始时，每个城市都有 a_i 个幸存者，由于交通不便，在一天的时间中，一个幸存者最多通过一条道路。为了保证安全，每个幸存者在一天结束时都需要呆在城市了，而不能呆在道路上。

如果一个城市某一天爆发了灾害，那么在这天结束时，所有在这个城市的幸存者都不能幸免于难。

借助灾害预测机的帮助，你可以在第一天时就知道接下来 m 天的灾害情况。你想要知道，在你的领导下，最多有多少人能活过这 m 天。

1.3 输入格式

第一行两个整数 n, m。

第二行读入 n 个非负整数 a_i。

接下来 $n-1$ 行，每行读入 x, y，表示 x, y 之间有一条双向道路。

接下来 m 行，每行先读入一个 k_i，表示第 i 天有 k_i 个城市爆发灾害。接着读入 k_i 个数 $p_{i,j}$ ，表示发生灾害的城市。

1.4 输出格式

一个整数表示最多能活下来多少人。

1.5 样例 1 输入

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

1.6 样例 1 输出

```
5
```

1.7 样例 1 解释

第一天城市 2 发生灾害，所以城市 2 的两个人都移动到城市 3，而城市 1 和 3 的人只能呆在原地。

第二天城市 1 和 2 都发生了灾害，城市 1 的人即不能呆在原地，又不能移动到城市 2，没有办法活下去，而城市 3 的人呆在原地就可以活下去。

1.8 样例 2

见下发文件

1.9 数据范围与约定

$1 \leq n \leq 10^6, 1 \leq m \leq 10^6, 0 \leq a_i \leq 10^4, \sum k_i \leq 2 \times 10^6$

子任务编号	分值	n,m	$\sum k_i$	特殊性质
1	20	≤ 10	≤ 50	$\sum a_i = 1$
2	11	≤ 200	≤ 10000	无
3	31	≤ 3000	$\leq 2 \times 10^5$	无
4	21	$\leq 2 \times 10^5$	$\leq 4 \times 10^5$	图是一个完全二叉树
5	17	$\leq 10^6$	$\leq 2 \times 10^6$	无

2 撤离

2.1 题目背景

即使有着灾害预测机的帮助，在没有物资补给的状态下，还是难以长期生存，于是你准备撤离 Z 国，带领大家前往最后的避难所。

2.2 题目描述

在通往避难所的路途中，有一段路受灾害影响严重。这段路可以被抽象为一个 $r \times c$ 的网格，左上角为 $(1,1)$，右下角为 (r,c)，其中有一些位置无法通行。

幸存者被分为了 n 支队伍，每支队伍拥有一个载具，初始时，第 i 支队伍的载具位于 $(1,w_i)$ 的正上方，且初始时方向都是向下。

在这段路的下方有 m 个出口，第 i 个出口位于 (r,p_i) 的正下方，出于某种原因，两个队伍不可以通过同一个出口出去。

由于灾害的影响，普通的载具在这段路上都会失灵导致无法拐弯，只有一辆小工程车可以在这段路上自由行走。为了让大家通过这段路，你想要先乘小工程车，到这段路的某些位置安装一些弹簧垫，来帮助大家通行。

弹簧垫都是 $45°$ 角放置的，一共有 4 种放置方法，分别可以产生以下效果：

1. 使从上方来到这个位置的载具方向变为向左，使从左方来到这个位置的载具方向变为向上，不能从右方和下方进入。

2. 使从上方来到这个位置的载具方向变为向右，使从右方来到这个位置的载具方向变为向上，不能从左方和下方进入。

3. 使从下方来到这个位置的载具方向变为向左，使从左方来到这个位置的载具方向变为向下，不能从右方和上方进入。

4. 使从下方来到这个位置的载具方向变为向右，使从右方来到这个位置的载具方向变为向下，不能从左方和上方进入。

每个位置至多放置一个弹簧垫，有些特定的位置和无法通行的位置不能放置弹簧垫。

放好弹簧垫后，n 支队伍的载具会依次出发，持续向着当前的方向前进（初始为下），直到从出口离开集体存活或者撞上障碍（不能通行区域或没有出口的墙壁或不能进入的弹簧垫区域）集体去世。

在每个位置安装弹簧垫需要的时间是不同的，你希望在保证所有人都存活的情况下，尽快摆好弹簧垫。

2.3 输入格式

第一行输入 T 表示有 T 组数据。

对于一组数据，先读入 r,c,n,m。

接着读入一个 $r \times c$ 的矩阵 A，如果 $A_{i,j}$ 是非负整数，表示在这个位置安装弹簧垫的时间。

如果 $A_{i,j} = -1$，表示这个位置无法通行。

如果 $A_{i,j} = -2$，表示这个位置不能安装弹簧垫。

下一行读入 n 个数，表示载具位置 $w_1 \cdots w_n$。

再下一行读入 m 个数，表示载具位置 $p_1 \cdots p_n$。

2.4 输出格式

对于一组数据，如果可以保证全员存活，先输出"Yes"，然后下一行输出最少需要的时间。

对于一组数据，如果不可以保证全员存活，先输出"No"，然后下一行输出最多可以存活的队伍数量。

2.5 样例 1 输入

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

2.6 样例 1 输出

```
Yes
4
No
1
Yes
```

9

2.7 样例 1 解释

对于第一组数据，在 $(1,2)$ 处放 4 号弹簧垫，在 $(1,4)$ 处放 1 号弹簧垫，在 $(3,1)$ 处放 2 号弹簧垫，在 $(3,4)$ 放 3 号弹簧垫，就可以让全员通过，总耗时为 4。

对于第二组数据，无法使全员通过，在不放弹簧垫的情况下，在 $(1,2)$ 上方的载具可以通过 $(3,2)$ 下方的出口，最多通过一支队伍。

对于第三组数据，在 $(1,1),(1,3),(3,3),(3,2)$ 放弹簧垫是用时最少的可行方案。

2.8 样例 2

见下发文件

2.9 数据范围与约定

对于 100% 的测试数据，$T \leq 10, r, c \leq 100, 0 \leq n \leq \min(c, 50), 0 \leq m \leq c, -2 \leq A_{i,j} \leq 100$ $1 \leq w_i, p_i \leq c$。

同一类型的测试点数据范围可能有梯度。

对于测试点 1~2，满足 $r, c \leq 10$，对于每组数据，$A_{i,j} \geq 0$ 的位置数 ≤ 8；

对于测试点 3~4，满足 $n = m = 1$；

对于测试点 5~6，满足 $c \leq 5, A_{i,j} \leq 0$；

对于测试点 7，满足 $A_{i,j} \leq 0$；

对于测试点 8~10，无特殊限制。

3 庇护

3.1 题目背景

你带领大家来到了庇护所，但庇护所入口的密码是一种特定代码的输出结果，而这种代码的解释器已经遗失了，于是你准备重写一个。

3.2 题目描述

你现在要实现一种语言的解释器，该语言的介绍如下：

该语言会顺序地执行每一条语句，语句包含赋值语句，if 语句，for 语句，输入输出。语句之间用换行隔开。

在程序中，英文字段和英文字段间用一个空格隔开，英文字段和符号直接相连（英文字段包括数字）

在每行程序的开头，可能会有任意数量的 tab（或空格）。

3.2.1 变量

该语言的变量类型只有整型，其数值的大小范围，四则运算的符号和结果都和 C++ 的 int 相同, 且只包含四则运算 $+, -, \times, \div$。

数组即为若干个变量，可以通过下标的方式访问其中的变量。

变量和数组无需声明，全都为**全局变量**，且名称为不超过 3 个的小写英文字母构成 (名称不为 for,if)，数组和变量不会重名。数组只包含一维数组，要求下标为非负整数。数组的下标只能为一个变量或常量。

赋值语句有两种类型：

A=B

A=BCD

其中，A 为变量，B,D 为变量或常量，C 为一个四则运算符，表示方式为 (+,-,*,/)，运算的结果和 C++ 的运算结果相同，保证运算在 C++ 中有定义。

本题中的运算符均为双目运算符，即不会出现 a=-10,a=-b 等语法，所以所有常量均为非负整数

在本题中，变量的调用个数不超过 10000，数组长度不超过 1000, 数组个数不超过 100

例：

a=10

b[23]=99

c[b[23]]=a+10

3.2.2 if 语句

形式如下：

```
if Y
//do something
else
//do something
fi
```

其中 Y 为一个判断语句，它由 A，B，C 这 3 个部分构成。

A,C 为一个变量或常量，B 为一个比较符号。

比较符号包含 <，>，=

Y 为真时，执行 else 上方的语句，否则执行下方的语句。

if 语句还有另一种形式，即忽略 else

```
if Y
//do something
fi
```

Y 为真时，执行 fi 上方的语句，否则不执行。

//do something 部分保证包含至少一行代码。

3.2.3 for 语句

形式如下：

```
for(A;B;C){
//do something
}
```

其中 A 为一个赋值语句，B 为一个判断语句，C 为一个赋值语句。

for 循环的执行规则和 C++ 相同，即在进入循环时，执行 A 语句；在每次循环开始时，如果 B 语句为真，进入循环，否则离开循环；进入循环后，按顺序执行//do somehing 中的内容，接着执行 C 语句，然后再次开始循环。

//do something 部分保证包含至少一行代码。

3.2.4 读入输出

```
input a
output b
```

a 为一个变量,b 为一个变量或常量。

其中 input a 的效果等价于 C++ 的

cin»a;

output b 的效果等价于 C++ 的

cout«b«endl;

3.2.5 任务要求

你要做的就是模拟一个程序的运行。

程序一定符合代码规范，不出现未定义的语法和未定义的行为。

3.3 输入格式

第一行一个 n 表示程序的行数。

接下来 n 行表示程序

一行一个整数 T, 表示有 T 组数据，即你要执行 T 次程序。

接下来若干行表示给程序输入的数据。

保证程序会读完所有数据。

3.4 输出格式

输出若干行表示运行 T 次程序的输出。

3.5 样例 1 输入

```
4
input a
input b
a=a+b
output a
3
1 2
-1 2
0 1000
```

3.6 样例 1 输出

```
3
1
```

```
1000
```

3.7 样例 1 解释

该程序实现了 a+b 问题。

3.8 样例 2 输入

```
7
input a
input b
if a<b
    output a
else
    output b
fi
2
1 3
4 3
```

3.9 样例 2 输出

```
1
3
```

3.10 样例 3/样例 4

见下发文件

3.11 数据范围与约定

输入数据的文件大小不超过 1KB(也就是说数据很弱)。

数据保证程序执行的语句总数和调用数组次数的总数不超过 100000。

读入的程序只包含小写字母, 数字, 四则运算符号, 比较符号, "{}[]();" 字符 '$\backslash n$'(换行符),' $\backslash t$'(tab), 空格。

注意：Windows 的换行符包含'$\backslash r$'，这在输入文件中是不应该出现的。

提示：你可以通过读入了几次'$\backslash n$' 来判断当前行数。

测试点 1： 10 pts 不包含 if,for,input, 变量。

测试点 2： 10 pts 不包含 if,for,input, 数组。

测试点 3： 20 pts 不包含 if,for, 数组。

测试点 4： 17 pts 不包含 for, 数组。

测试点 5： 15 pts 不包含 if,for。

测试点 6： 15 pts 不包含 for。

测试点 7： 13 pts 没有特殊限制。